

Secure and Fast OTP Authentication Protocol for Wireless Networks

Student: M. Newlin Rajkumar

Advisor: Dr. Yu-Lun Huang

Department of Electrical and Control Engineering

National Chiao Tung University

Abstract

Wireless Networks provide more flexibility and connectivity to the users, to be connected with anyone, any time, any where. This feature makes the wireless networks to be wide open to threats and attacks by the intruders. To ensure a reliable and secure wireless environment, suitable authentication protocols have to be employed based on the network infrastructure. The scope of this thesis is to present a general approach of design and analysis, of an authentication protocol, which is robust and easy to implement in real time. Along with these aspects, this protocol is coupled with secure and fast authentication features and especially designed for large scale enterprise wireless networks. The protocol design is formalized with Protected One Time Password Server (POTP), which ensures secure One Time Password (OTP) authentication and an RADIUS Authentication, Authorization and Accounting (AAA) server which favors the user authentication. The Access Point (AP) involves in the subsequent authentication phase for the user requests.

Acknowledgements

At the foremost, I would like to articulate, my heartfelt gratitude and my sincere thanks to my Research Advisor Dr. Yu-Lun Huang, for guiding and advising all the way through this Research and Setting up a new perspective for me, in the arena of Wireless Networks and Security design. Throughout this Research, her guidance meant a lot to me, both academically and personally,

Special thanks to Dr. James Ho, Dr. Wu Yang and Dr. Wen-Guey Tzeng for rendering their consent and serving as Advisory Committee Members.

Next, I would like to thank all my lab mates, who assisted this Research, in preparing and proof reading my thesis.

Finally, I extend my gratefulness, to all my friends in NCTU, for their motivation and active support, right through my study.

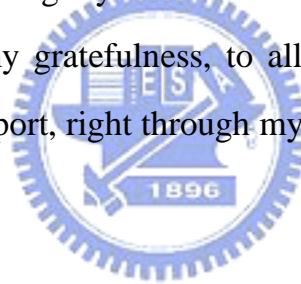


Table of Contents

Abstract	i
Acknowledgements	ii
Table of Contents	iii
Table of Contents	iii
List of Tables	v
List of Figures	vi
Chapter 1 Introduction.....	1
1.1 Background.....	1
1.2 Contribution	3
1.3 Synopsis	3
Chapter 2 Related Work.....	5
2.1 Authentication Protocols	5
2.2 RADIUS protocol	6
2.4 KERBEROS authentication protocol	12
2.5 OTP-based key-exchange technology	16
2.5.1 S/Key OTP authentication protocol	17
2.5.2 Protected One Time Password (POTP) scheme	18
2.6 Summary.....	19
Chapter 3 Proposed Secure and Fast OTP authentication protocol	20
3.1 Notations	20
3.2 Assumptions	21
3.3 Network Topology.....	21
3.4 Protocol Architecture	23
3.5 Authentication Message Flow.....	24
3.5.1 Authentication Message Flow Description	24
3.6 Secure and Fast OTP Authentication Protocol.....	25
3.6.1 Registration Phase	25
3.6.3 Password server Token Number Verification	28
3.6.4 Subsequent Authentication phase.....	29
3.7 Authentication Goals	29
3.8 Summary.....	30
Chapter 4 Security and Privacy Analysis.....	31
4.1 Security Analysis	31
4.2 Security Proof - using SVO Logic	34
4.3 Security Properties – Comparison table.....	34
4.4 Summary.....	36
Chapter 5 Performance Analysis.....	37

5.1 Storage Cost	39
5.2 Fitness function	40
5.2.1 Security fitness function (s)	40
5.2.2 Efficiency fitness function (e)	41
5.2.3 Results of Fitness functions	41
5.2.4 Result Analysis w.r.to Weight strategy	43
5.3 Summary	45
Chapter 6 Special Features of Our Protocol	46
Chapter 7 Conclusion and Future Work	47
7.1 Conclusion	47
7.2 Future Work	47
References	49
Appendix	51



List of Tables

Table 3-1 Notations	20
Table 4-1 Security Properties - Comparison	35
Table 5-1 Comparison of Computational & Communications Efficiency	38
Table 5-2 Comparison of storage cost.....	39
Table 5-3 Weighting Strategies for N = 6.....	40
Table 5-4 Fitness function weightings for the first search	43



List of Figures

Figure 1-1 Basic IEEE 802.1x Authentication structure.....	2
Figure 2-1 RADIUS proxying method	7
Figure 2-2 RADIUS protocol – Authentication message flow diagram.....	8
Figure 2-3 RADSEC proxying method.....	9
Figure 2-4 Functionalities of RADSEC protocol	10
Figure 2-5 Kerberos authentication protocol - working principle.....	12
Figure 2-6 Kerberos authentication protocol - message flow	14
Figure 2-6 S/Key OTP authentication protocol.....	17
Figure 3-1 A Network Domain module – Large scale Enterprise Wireless Network.....	22
Figure 3-2 Protocol Architecture	23
Figure 3-3 Authentication Message Flow	24
Figure 5-1 Basic message flow of Login Phase	44
Figure 5-2 A symmetric key protocol found in the first search.....	44



Chapter 1

Introduction

The development of wireless networks range from WPAN (Wireless Personal Area Network) implemented using Bluetooth, WLAN (Wireless Local Area Network) Using 802.11b, 802.11a,g,i and WWAN (Wireless Wide Area Network) accomplished by the operators of wireless networks using GPRS, CDMA, etc.. Though this wide network range helps the users to achieve high connectivity, they are prone to intruders, hackers and attackers everywhere; either the network is WWAN or WLAN. Thus, more care should be taken to ascertain network security, from the initial design configuration thru its implementation. To ensure network security, proper authentication mechanisms should be equipped and authentication protocols serve this purpose. Hence, these authentication protocols should be efficiently well designed, in performing fast authentication, along with their security enforcement strategy. This thesis is entitled "Secure and Fast OTP Authentication Protocol for Wireless Networks", which by itself, enlighten the goals and perspectives of this research, in dealing with the security issues of WLAN.

1.1 Background

Technical advancements in wireless networks, provide many security features in the mode of authentication protocols. The structural design of these protocols depends upon the Network infrastructure, implementation scenario, and deployment environment. Though the designs of these protocols differ in practice, the basic authentication principle remains the same for all. The basic authentication standard 802.1x is devised by IEEE, which forms root of all authentication standards.

Figure 1-1 given below, depicts the IEEE 802.1x Authentication structure, with all basic components required for authentication process. This process is a common scheme which is generally applicable to all ranges of wireless networks, but only during implementation, the

design of architecture may vary, depending upon the wireless environment and the mode of implementation.

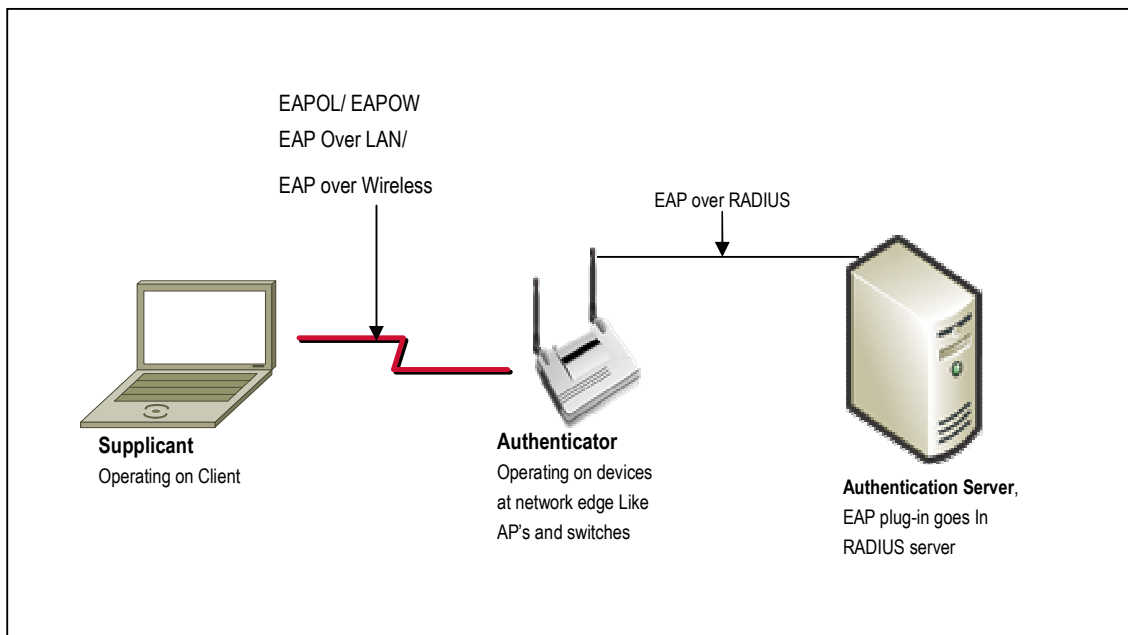


Figure 1-1 Basic IEEE 802.1x Authentication structure

In this authentication structure, to verify the credentials of the supplicant, the authenticator uses an authentication server. The authentication server checks the credentials of the supplicant on behalf of the authenticator, and then responds to the authenticator, indicating whether or not the supplicant is authorized, to access the authenticator's services.

During authentication, the Access Point (AP) forwards the credentials of the wireless connection attempt to the authentication server. The wireless AP uses the Remote Authentication Dial-In User Service (RADIUS) protocol to send the connection attempt parameters to a authentication server. The major drawback in this authentication scheme is “User’s Password”, which is sent to the server for authentication purpose. User provides his credentials – in the form of a user name and password – when making a request. The server verifies the user and grants access if the user credentials are matching, else it denies the user.

Although this scheme is easily implemented, it relies on the assumption that the connection

between the client and server computers is secure and can be trusted. Specifically, the credentials are sent across the network as plaintext and could be intercepted easily and subsequently used by eavesdroppers to impersonate the user. Password based authentication is inconvenient; users do not want to enter a password each time they access a network service when they are roaming.

Hence, password based authentication is not suitable for use on computer networks.

1.2 Contribution

To overcome this password based authentication issue, we have proposed a general approach of design and analysis, of an authentication protocol, which functions by using One Time Password (OTP) scheme thru Password Server (PS), along with RADIUS Authentication, Authorization and Accounting (AAA) server (AS). Through this thesis, we explain the proposed authentication protocol design, its functionality and its implementation scenario in a large scale enterprise network. We also provide a detailed view, on the security analysis and performance analysis of the proposed protocol. The security proof is realized by utilizing SVO logic, which is a well known standard, for proving the effectiveness of authentication protocols. Security features of our protocol are compared with that of some existing protocols and tabulated. The overall performance of our protocol is scrutinized thru efficiency analysis, which describes the communications efficiency, computational efficiency and storage cost. Finally, the application of fitness function, evaluates the overall performance of our protocol thru an analytical approach.

1.3 Synopsis

The remaining part of this thesis is categorized in the following manner. Chapter 2 reviews the related work, which provides the architectures of some of the existing authentication protocols and their pros and cons related to authentication scenario. In chapter 3, we explain our proposed authentication protocol, its architecture, message flow and functionality. Chapter 4 provides security and privacy analysis of our design. In chapter 5, we describe the performance analysis of our protocol, followed by its special features in chapter 6. In the final chapter, we present our

concluding concepts to enhance our design in the future, in expanded wireless networking environments.



Chapter 2

Related Work

2.1 Authentication Protocols

For over many years, a wide range of authentication protocols have been proposed by researchers and they still continue their research in getting new protocol forms with specific communication properties and added security features. In spite of the special features indulged within these authentication protocols, they still have some constraints in real time implementation, while facing security threats and attacks. It is true that any wireless infrastructure is open to wireless attacks. Some of the wireless attacks are similar to that of wired network environment; some are severe and some are fresh. The base behind all these attacks is the underlying communications channel, the airwave medium, which is wide open to intruders.

The loss of confidentiality and integrity and the threat of Denial of Service (DoS) attacks are risks, typically associated with wireless networks. Intruders may obtain access to systems and information, corrupt the data, consume network bandwidth, degrade network performance, launch attacks that prevent authorized users from accessing the network, or use the resources to launch attacks on other networks. Specifically to say, in the wireless authentication attacks, the impact is more. Intruders use these attacks to steal legitimate user identities and credentials to access otherwise private networks and services. Replay Attack, Eavesdropping attack, Man-in-the-Middle attack and User Impersonation attacks are the most common attacks.

Shared Key Guessing, PSK Cracking, Application Login Theft, Domain Login Cracking, VPN Login Cracking, 802.1X Password Guessing, 802.1X LEAP Cracking and, 802.1X EAP Downgrade, are also considered to be harmful attacks.

To attain a secure channel of communication, secure authentication is considered as the backbone and an essential factor in any computer networked environment. It proves pivotal, when network security has to be ensured. The necessity of secure authentication emerges in many

different forms, and authentication protocols are employed in numerous ways, based on their essential security properties and on the nature and complexity of the network environment.

Hence, these authentication protocols should prove their consistency and highly secured nature, along with high speed authentication capabilities. We list some of the authentication protocols widely in usage now and explain their design flow, along with their merits and demerits.

2.2 RADIUS protocol

RADIUS protocol (RFC 2865)[1][2][3] provides centralized authentication and access control services, to a variety of network access and application-layer devices. This protocol has been in use for many years and has become the de-facto standard for providing AAA services.

Being the basic standard for all authentication protocols, and most commonly used authentication protocol, it is inbuilt with a lot of advantageous features. RADIUS protocol is well-understood and implementations are widely available and widely supported. It is capable of supporting a variety of connection types and is no longer limited to human identities. RADIUS protocol is considered as a general-purpose distributed authentication protocol for network connections of all types. Currently, RADIUS protocol is the widely used protocol in network environments. It is commonly used for embedded network devices such as routers, modem servers, switches, etc. It is used for several reasons:

The embedded systems generally cannot deal with a large number of users with distinct authentication information. This requires more storage than many embedded systems possess. Since, RADIUS protocol facilitates centralized user administration, which is important for several of these applications; the role of this protocol proves vital. Many ISPs have tens of thousands, hundreds of thousands, or even millions of users. Users are added and deleted continuously throughout the day, and user authentication information changes constantly. Centralized administration of users in this setting is an operational requirement and this is provided by RADIUS. RADIUS consistently provides some level of protection against sniffing and active attacks. Other remote authentication protocols provide either intermittent protection, inadequate

protection or non-existent protection. RADIUS support is nearly omni-present. Other remote authentication protocols do not have consistent support from hardware vendors, whereas RADIUS is uniformly supported. Because the platforms on which RADIUS is implemented on are often embedded systems, there are limited opportunities to support additional protocols. Any changes to the RADIUS protocol would have to be at least minimally compatible, with pre-existing (unmodified) RADIUS clients and servers.

In spite of these merits, in the time since its release, it has revealed a number of serious shortcomings when used in some environments. RADIUS proxying was used to send conventional RADIUS requests across insecure networks which permit the users to roam on the networks and RADIUS authentication and accounting requests are sent, to the user's home RADIUS server, based on the user's Realm. Figure 2-1 given below depicts the RADIUS proxying method.

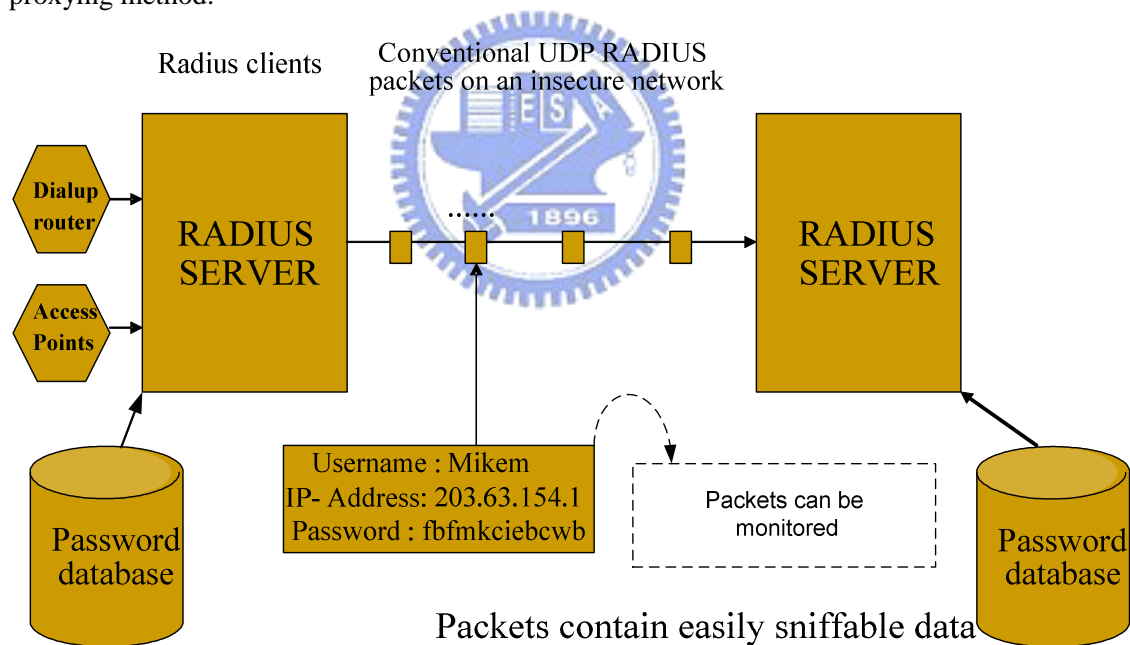


Figure 2-1 RADIUS proxying method

From figure 2-1 we realize that the data in conventional RADIUS access requests is mostly plaintext, including the user name, IP address, login times etc. The user's password is encrypted

with a shared secret, but using a fairly weak encryption algorithm. This means that eavesdroppers can monitor the packets in the conventional RADIUS requests. This is usually a problem when the RADIUS requests travel across the internet or any other insecure or shared network.

Also, the conventional RADIUS protocol does not always provide a reliable indication of whether the RADIUS server we are connected to is the one we expect, or that the client that sends a request, is really who it claims to be. This means that it is relatively easy to spoof RADIUS clients and servers when using conventional UDP based RADIUS proxying. This can also be used by attackers, to gain valuable information about an operator's network and users.

The authentication message flow of RADIUS protocol is presented in detail in figure 2-2, which explains that, when a user need to be authenticated, he sends his username and password to the Network Access Server (NAS). The user's password is encrypted by using MD5 algorithm. The username, NAS id and Port address, are sent to the RADIUS server. The server validates or discards the client, after password verification.

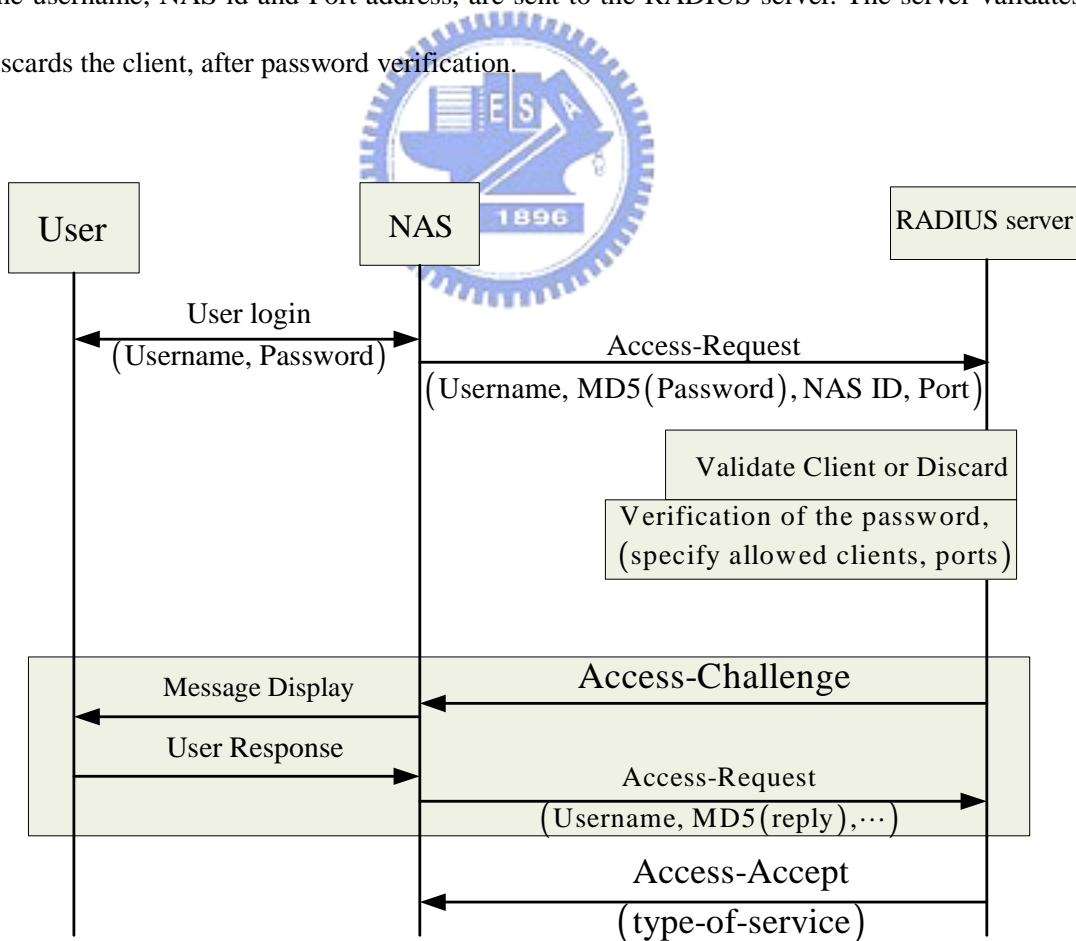


Figure 2-2 RADIUS protocol – Authentication message flow diagram

2.3 RADSEC protocol

RADSEC protocol is a secure, reliable protocol for proxying RADIUS requests. In other words RADSEC protocol provides secure and reliable RADIUS transport of RADIUS proxying requests [3][4][5]. It was designed by Open System Consultants, in response to some of the shortcomings of the conventional RADIUS protocol.

RADSEC is used to carry RADIUS traffic between 2 cooperating RADIUS servers, or between a RADIUS client and a RADIUS server. In either case, one end of the connection is the RADSEC client and the other is the RADSEC server. Figure 2-3 shows the RADSEC proxying

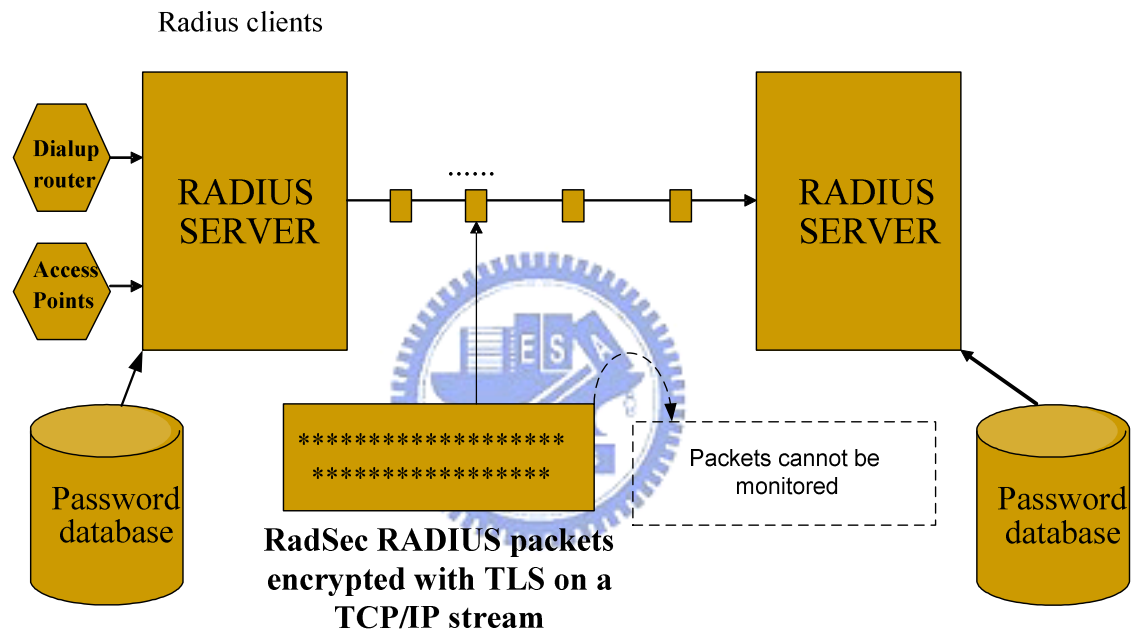


Figure 2-3 RADSEC proxying method

A RADSEC server can also be the client of another RADSEC server. From figure 2-3, we recognize that in the RADSEC protocol, the RADIUS proxy requests are sent in an encrypted form, and so, it is not possible to monitor the packets as like that of conventional RADIUS proxying. RADSEC compliant instances can connect to multiple other RADSEC instances. For some connections it can act as the client, and in others it will act as the server. RADSEC servers listen for connections on a well-known TCP/IP port. The default port number is 2083 (IANA official RADSEC port number).

The following figure 2-4, gives the functionalities of RADSEC protocol.

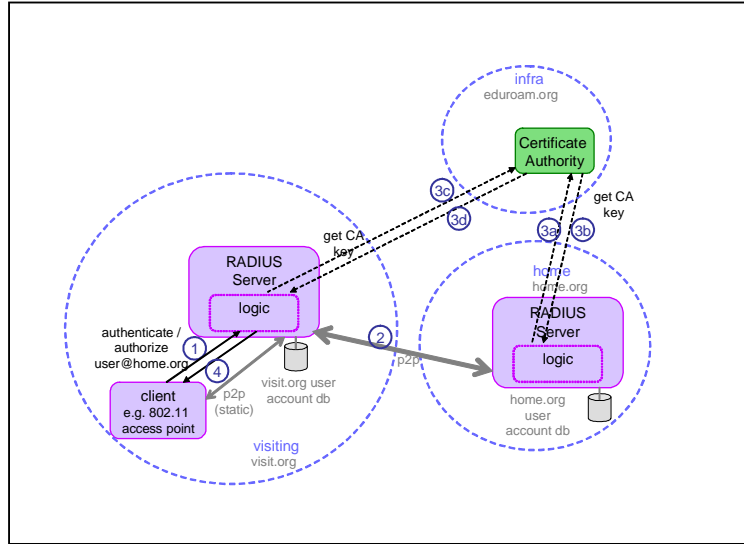


Figure 2-4 Functionalities of RADSEC protocol

From figure 2-4, we realize the functionalities of RADSEC protocol, which explains the following

Message 1: RADSEC clients initiate a RADSEC connection by establishing a TCP/IP connection to the address and port number of the RADSEC server. The address and port number are should be well known by clients who are willing to connect to a given server (i.e. the server administrator will notify intending client administrators of the address and port number to connect to). TCP/IP modules will be enabled for the stream, on platforms that support them.

Message 2: After a TCP connection is established, TLS handshaking commences if client and server are configured to use TLS. TLSV1 is used for this handshaking function.

Messages 3a, 3b and 3c, 3d: RADSEC servers present a PKI certificate with a CN (Common Name) identical to the DNS host name corresponding to the address where the server listens. The server might be configured to require a PKI certificate from the client. If a valid certificate is not presented, the TLS handshake will fail. Certificates may be from either public or private certificate authorities. If the TCP connection between client and server should fail or being disconnected, or the TLS handshake should fail for any reason, the client will attempt to reconnect

to the server at configurable intervals, default 5 seconds. TLS session resumption is not required or supported. This property is an added advantage in this protocol.

Message 4: Once a stream connection is established (and TLS handshake successfully completed if required), RADIUS requests are sent from the client to the server and replies are sent from the server to the client through the stream. RADIUS packets are encoded on the stream, in precisely the same format as standard RADIUS UDP packets, including the request type, identifier and packet length. There are no record separators, since the RADIUS packet length field is used to determine the record boundaries.

Like conventional RADIUS proxying, every RADSEC connection will have a shared secret associated with it, and known to the RADSEC Client and Server (regardless of whether or not TLS encryption will be used on the transport). The RADIUS authenticator and any user-password fields in the RADIUS packets will be encrypted using the connection's shared secret as conventional RADIUS UDP packets. This serves to provide at least the same level of protection as conventional RADIUS, when RADSEC is used without TLS encryption.

RADSEC clients use either the normal RADIUS packet identifier field or the proxy-state attribute to match RADIUS replies to requests. This state is used to avoid the limitations of the 8 bit RADIUS identifier. RADIUS replies may be sent back from the server in a different order to the received requests. There is no guarantee that reply will be sent in response to any particular request. Client may use the User-Name, Realm, or any other combination of RADIUS attributes to determine which RADSEC server, to send a RADIUS request. This permits RADIUS servers to implement Realm-based proxying using RADSEC to a user's home RADSEC / RADIUS server.

TLS encryption should be enabled and mutual RADSEC client/ server authentication required when using RADSEC protocol is over an insecure network. Though the RADSEC authentication protocol is said to be comparatively secure than the RADIUS protocol, it is still in the scratch, in terms of real-time applications. The implementation issues and complex nature, lead the protocol to be unpopular, comparatively with the RADIUS protocol.

2.4 KERBEROS authentication protocol

Kerberos [6][7] is a centralized secure authentication protocol that allows a process (a client) running on behalf of a principal (a user) to prove its identity to a verifier (an application server) without sending data across the network that might allow an attacker or the verifier to then impersonate the principal. This protocol provides integrity and confidentiality for data sent between the client and server, and facilitates a fast response to security attacks. The added advantage of this protocol, is its availability, as it is freely available and Open-source protocol . This protocol was developed in the mid-'80s as part of MIT's Project Athena.

Kerberos authentication protocol uses a series of encrypted messages to prove to a verifier that a client is running on behalf of a particular user. This protocol uses timestamps to reduce the number of messages needed for basic authentication and also Kerberos uses “ticket-granting” service, to support subsequent authentication, without re-entry of a principal's password. The encryption standard used for authentication is the Data Encryption Standard (DES).

The following figure 2-5, explains the working principle of Kerberos authentication protocol.

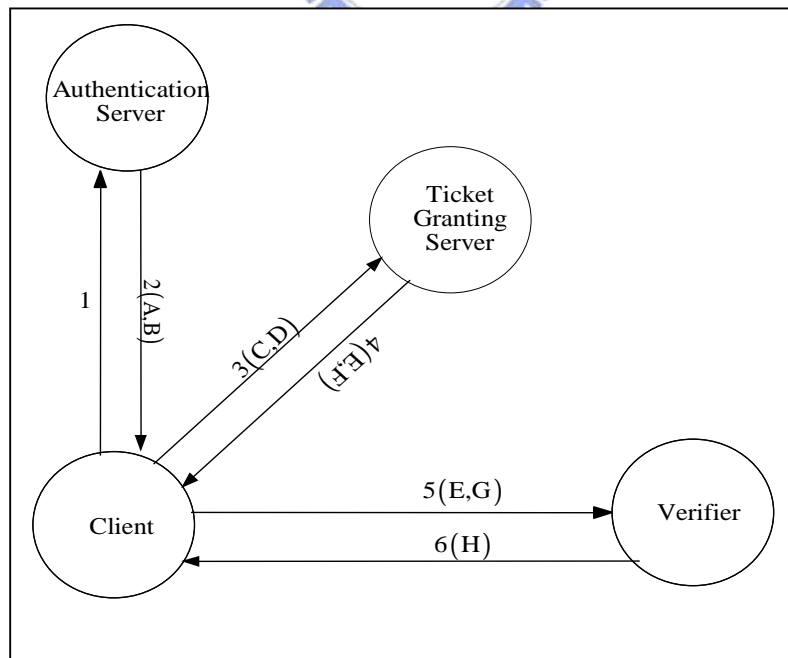


Figure 2-5 Kerberos authentication protocol - working principle

From figure 2-5. Arrows 1 and 2 are used only when the user first logs in to the system, Arrows 3 and 4 are used whenever a user authenticates to a new verifier, Arrow 5 is used each time the user authenticates itself, Arrow 6 is optional and used only when the user requires mutual-authentication.

To begin the authentication process, a user enters a username and password on the client. Then the client performs a one-way hash on the entered password, and this becomes the secret key of the client. Following this the client sends a clear-text message to the Kerberos Authentication Server (AS) requesting services on behalf of the user. Now the AS checks to see if the client is in its database. If it is, then it sends back the following two messages to the client: Message A: client/TGS (Ticket Granting Server) session key encrypted using the secret key of the user and Message B: Ticket-Granting Ticket (which includes the client ID, client network address, ticket validity period, and the client/TGS session key) encrypted using the secret key of the TGS.

Once the client receives messages A and B, it decrypts message A, to obtain the client/TGS session key. This session key is used for further communications with TGS. At this point, the client has enough information to authenticate itself to the TGS. When requesting services, the client sends the following two messages to the TGS: Message C: Composed of the Ticket-Granting Ticket from message B and the ID of the requested service and Message D: Authenticator (which is composed of the client ID and the timestamp), encrypted using the client/TGS session key.

Upon receiving messages C and D, the TGS decrypts message D (Authenticator) using the client/TGS session key and sends the following two messages to the client.

Message E: Client-to-server ticket (which includes the client ID, client network address, validity period and Client/server session key) encrypted using the verifier's secret key.

Message F: Client/server session key encrypted with the client/TGS session key. Upon receiving messages E and F from TGS, the client has enough information to authenticate itself to the Verifier. The client connects to the Verifier and sends the following two messages: Message E from the previous step (the client-to-server ticket, encrypted using verifier's secret key) and

Message G: a new Authenticator, which includes the client ID, timestamp and is encrypted using client/server session key. The Verifier decrypts the ticket using its own secret key and sends the following message to the client to confirm its true identity and willingness to serve the client.

Message H: the timestamp found in client's recent Authenticator plus 1, encrypted using the client/server session key.

The client decrypts the confirmation using its shared key with the server and checks whether the timestamp is correctly updated. If so, then the client can trust the server and can start issuing service requests to the server. The server provides the requested services to the client.

The detailed message flow of the protocol is given in the figure 2-6, elaborating each message field components. AS represents the Authentication server, C refers the client and V refers to the Verifier.

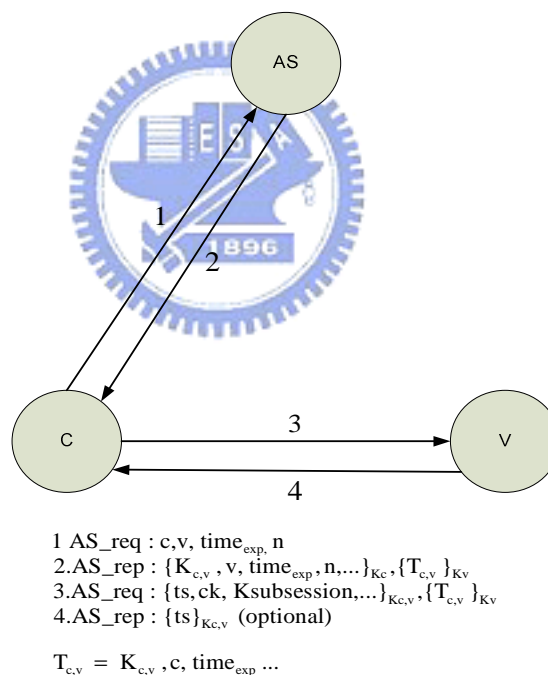


Figure 2-6 Kerberos authentication protocol - message flow

When a client wishes to create an association with a particular verifier, the client uses the authentication request and response, messages 1 and 2 from figure 2-6, to obtain a ticket and session key from the authentication server. In the request, the client sends the authentication server its claimed identity, the name of the verifier, a requested expiration time for the ticket, and

a random number, that will be used to match the authentication response, with the request.

In its response, the authentication server returns the session key, the assigned expiration time, the random number from the request, the name of the verifier, and other information from the ticket, all encrypted with the user's password, registered with the authentication server, together with a ticket containing similar information, and which is to be forwarded to the verifier as part of the application request. Together, the authentication request and response and the application request and response, comprise the basic Kerberos authentication protocol.

Messages 3 and 4 in figure 2-6, show the application request and response, which is the most basic exchange in the Kerberos protocol. In this message a client proves to a verifier that it knows the session key embedded in a Kerberos ticket. There are two parts to the application request: a ticket and an authenticator. The authenticator includes, among other fields: the current time, a checksum, and an optional encryption key, all encrypted with the session key from the accompanying ticket.

Upon receipt of the application request, the verifier decrypts the ticket, extracts the session key, and uses the session key to decrypt the authenticator. If the same key was used to encrypt the authenticator as used to decrypt it, the checksum will match and the verifier can assume the authenticator was generated by the principal named in the ticket and to whom the session key was issued. This is not by itself sufficient for authentication, since an attacker can intercept an authenticator and replay it later to impersonate the user. For this reason the verifier additionally checks the timestamp to make sure that the authenticator is fresh. If the timestamp is within a specified window (5 minutes) centered around the current time on the verifier, and if the timestamp has not been seen on other requests within that window, the verifier accepts the request as authentic.

Now the identity of the client has been verified by the server. If such mutual authentication is required, the server generates an application response by extracting the client's time from the authenticator, and returns it to the client, together with other information, all encrypted using the session key. The client requires a separate ticket and session key for each

verifier with which it communicates.

Though Kerberos authentication protocol is considered to be a suitable authentication protocol now, it has a few limitations which include ineffectiveness against password guessing attacks and requiring a trusted path through which passwords are entered. The other constraints are Kerberos server should be integrated with other parts of the system. It does not protect all messages sent between two computers; it only protects the messages from software that has been written or modified to use it.

2.5 OTP-based key-exchange technology

Based on the above related research works, the major problems concerning User Authentication arises, due to user password weakness, and when these weak passwords are applied during authentication, the result may lead to a heavy failure in real-time applications.

So, a solution to this password based authentication problem, is by implementing One Time Password (hereby referred as OTP)[8][9] based key-exchange technology. This scheme provides protection against capture of the user's password, capture of the server's password-database, dictionary attacks on the user's password and denial-of-service attacks.

In addition, an OTP-based key-exchange technology allows users to connect from an un-trusted terminal and still preserve the privacy of data transmitted ensuring confidentiality, authenticity, and integrity of the data and mutual authentication of the user and the server.

Moreover, OTP technology limit effectiveness of sniffing, OTP breaks up domino effect (use the same password for more than one system causes a domino effect, if a password is guessed) noted in recent hacks. Additional protection for unsecured, commercial environments (shared / home computers) provided by this technology, makes the user to choose, this new trend of password authentication in wireless networks.

2.5.1 S/Key OTP authentication protocol

The S/Key OTP authentication protocol in other words “Secure Key” authentication protocol, is closely based on a scheme devised by Lamport, [10][11][12], has been designed to provide users with 'one-time passwords', which can be used to control, user access to remote hosts. After the user authentication process is complete, i.e. after the OTP has been sent across the network, no protection is offered against subversion of the link by third parties.

The S/Key OTP scheme operates in the following way: The user holds a PC and possesses a 64-bit secret key. This key is derived from a 'pass-phrase' of arbitrary length, thus avoiding the need for it, to be stored by the user's hardware. The user and host share an implementation of a one-way function, which takes a 64-bit input and gives a 64-bit output.

The function is based on the MD4 hash-function, although it could be based on any other suitable function (e.g. SHA or RIPEMD-160). The key is used to generate a sequence of 'one-time passwords' in the following way and figure 2-6 elaborates the S/Key authentication scheme.

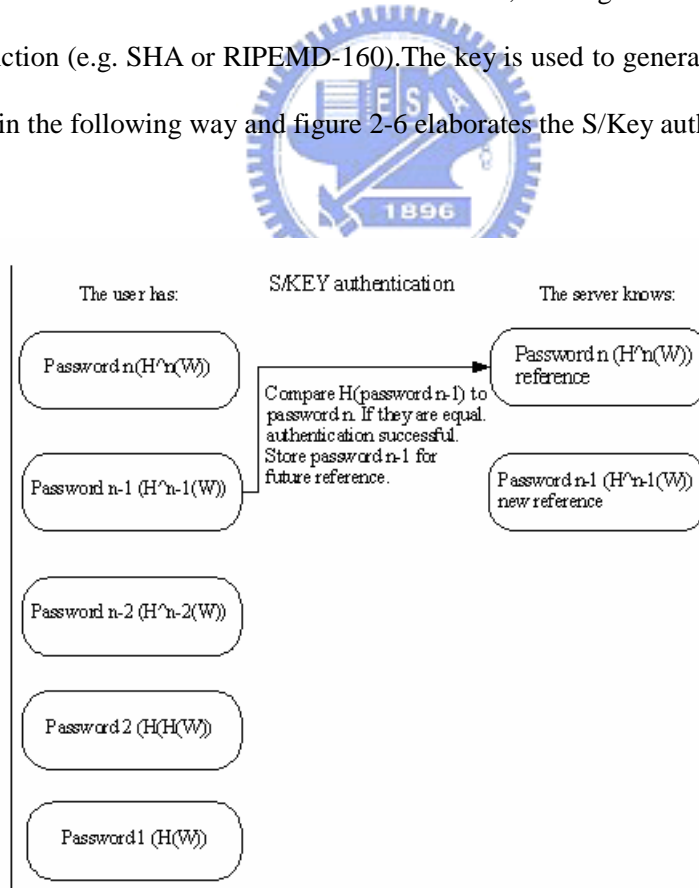


Figure 2-7 S/Key OTP authentication protocol

Each host which the user wishes to access is assigned a 64-bit seed value and a 'count' value, which will initially be set to some fixed value (e.g. 1000). For each user wishing to gain access, the host retains three components, a copy of the seed for that user, the current 'count' value for that user, and the previous one-time password for that user.

When a user wishes to gain access to the host, the following procedure is followed, the host decrements its stored counter for that user, and sends the value of this decremented counter, c say, to the user in conjunction with the seed value, D say. On receipt of D and c , the user takes D and bit-wise exclusive-ors it with its 64-bit key, to obtain a 64-bit value S .

The one-way function shared by the user and host is then recursively applied to S a total of c times to obtain the 64-bit one-time password P . The value P is then sent back to the host. On receipt of P , the host applies the one-way function to P once, and compares the result with its stored 'old password'. If the two values agree then the user is authenticated and the 'old' stored password is replaced with P . Otherwise the user is rejected.

Setting the system up, will require the user to enter his/her pass-phrase into the host, where the initial count can be chosen, and the initial password computed and stored. It is important to note that the host does not need to retain any secrets, since it only keeps the 'old' password, from which the new password cannot be derived.

The Limitations of S/Key scheme are, it does not protect a network eavesdropper from gaining access to private information, and does not provide protection against "inside" jobs or against active attacks where the potential intruder is able to intercept and modify the packet stream.

2.5.2 Protected One Time Password (POTP) scheme

This scheme is opted for providing proper OTP verification, which assumes the use of a shared secret key, or "seed", which is known both by the user and the POTP Server or Password Server (PS). The secret seed is stored on an OTP token that the user possesses, as well as in PS. In this method, the user provides his credentials to PS which verifies the OTP (which is sent by the user) with that of the OTP generated by it. If the OTP verification matches, then the OTP could be

considered to be originated from a recognized user. POTP scheme could also provide mutual authentication and protection against eavesdropping when needed.

2.6 Summary

In this chapter, we have described some authentication protocols and explained about their construction and working principle. RADIUS authentication protocol and RADSEC authentication protocol scale a slight difference between them, since the RADSEC is the successor of the RADIUS. RADIUS authentication protocol shows some limitations in packet exchanging while RADSEC authentication protocol is complex in construction and so unpopular still. Then, the Kerberos authentication protocol, which is a centralized secure authentication protocol, but the constraints include ineffectiveness in password guessing attacks. Finally we discussed about the OTP based Key exchange technology which provides protection against capture of the user's password, capture of the server's password-database, dictionary attacks on the user's password and denial-of-service attacks. The subsection in the OTP based Key Exchange Technology is the S/Key OTP authentication protocol which is a form of OTP authentication protocol, providing secure user password, but possessing some limitations like vulnerability to eavesdropping attacks or not provide protection against the modification of packet stream.

Thus all the authentication protocols mentioned in the related work, have some pros and cons, which indeed induced us to present our authentication protocol based on AS and PS, which is to be analyzed in detail and compared with all the above mentioned protocols in terms of security properties and performance evaluation, in the forth coming sections.

Chapter 3

Proposed Secure and Fast OTP authentication protocol

Owing the limitations of the security protocols given in detail, in our related work, we have framed a protocol, which could be able to overcome these limitations and could perform better compared to these protocols. This authentication protocol could be well suited and especially designed for providing a secure wireless networking environment for inter domain wireless networks. For experimental sake we consider a single module, in a large scale Enterprise wireless network environment. Our design is based on OTP technology, which provides security against password based authentication attacks, which forms the base of our protocol's architecture.

3.1 Notations

The following notations are used in our protocol design.

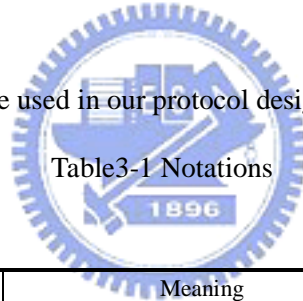


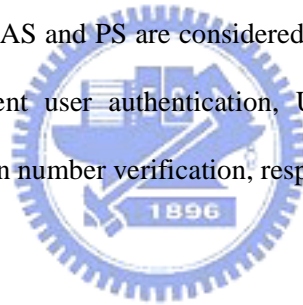
Table3-1 Notations

Notations	Meaning
U	User
AP	Access Point
AS	RADIUS AAA Server
PS	POTP Server (Password Server)
Seed	unique secret 128 bit random key, which is factory-encoded
PIN	Password which has to be memorized by the user
T	Token number
N_u	Nonce generated by user
N_s	Nonce generated by PS
$K_{AS,PS}$	Pre shared Key between AS and PS
$K_{AS,AP}$	Pre shared Key between AS and AP
K_{SS}	Session key generated by the AS and sent to AP
Sid	Session identifier (8 digit alpha numeric value, valid only for the session)
TS	Time stamp

3.2 Assumptions

We design the authentication protocol based on the following assumptions which are specifically mentioned below.

- I. This protocol is designed to provide secure and fast OTP authentication for large scale enterprise wireless network environment.
- II. This protocol is aimed in providing client authentication only.
- III. AS and PS are mutually trusting each other thru a pre shared key $K_{AS,PS}$
- IV. AS and AP are mutually trusting each other thru a pre shared key $K_{AS,AP}$
- V. The Session key K_{SS} issued by AS is highly efficient and effectively capable, so that it could be used by the user for the subsequent authentication process.
- VI. The Access point, AS and PS are considered to be highly proficient to perform their tasks of Subsequent user authentication, User credentials verification and OTP validation by Token number verification, respectively.



3.3 Network Topology

The network topology chosen to implement our protocol is a large scale enterprise environment. The adaptive nature of our protocol makes it easy to implement in the large scale wireless networks. To explain our protocol we consider a single network domain in a large scale enterprise wireless network and the structure in figure 3-1.

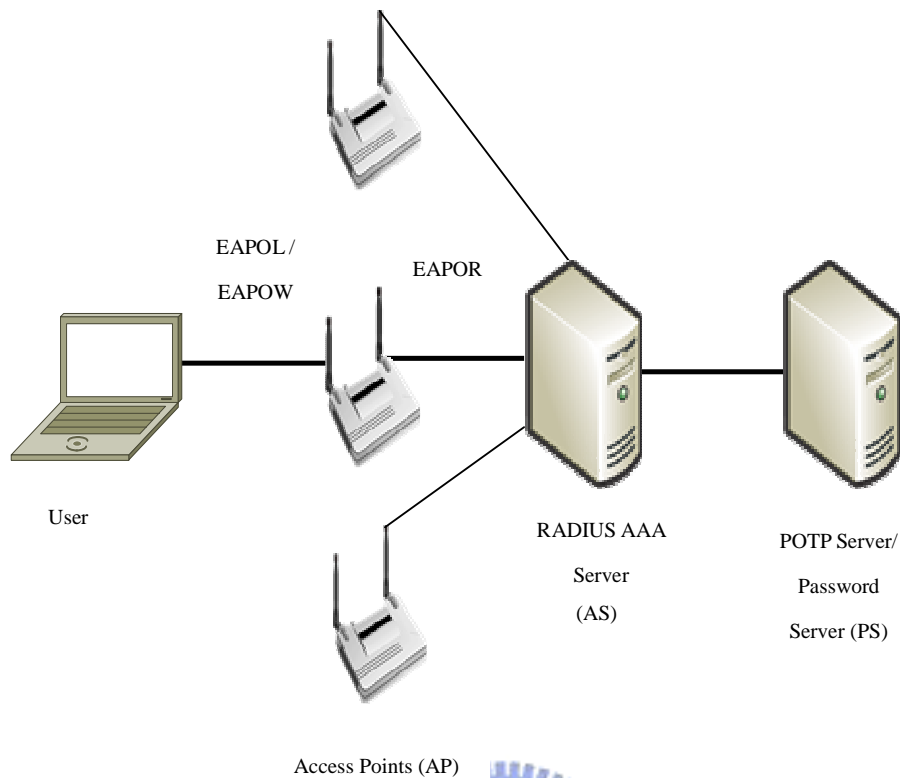


Figure 3-1 A Network Domain module – Large scale enterprise wireless network

In this structure the network components are, User, APs, AS and PS. In figure 3-1 three APs are considered for example. The User is connected the Access Point thru Extensible Authentication Protocol Over LAN (EAPOL) or Extensible Authentication Protocol Over Wireless (EAPOW). The AP and AS are connected thru Extensible Authentication Protocol Over RADIUS (EAPOR).

The User who is connected to the AP in the middle, is shown with darkened black line of connection and the other connections from AP to the AS are given in light Black color connection lines. When the user enters his credentials, they are sent to the AP; it uses the AS to verify the credentials of the user. AS checks and verifies the credentials of the user on behalf of the access point and forwards the user details to the PS, which verifies the user's one time password, and responds to the AS indicating whether or not the user's one time password is authorized and if so,

the PS initiates the AS to provide user authentication.

3.4 Protocol Architecture

This architecture is based on AS and PS. Since both these servers act together to perform user authentication and the link between both these servers are established thru the pre shared key $K_{AS,PS}$. The figure 3-2, given below gives the architecture of our proposed protocol.

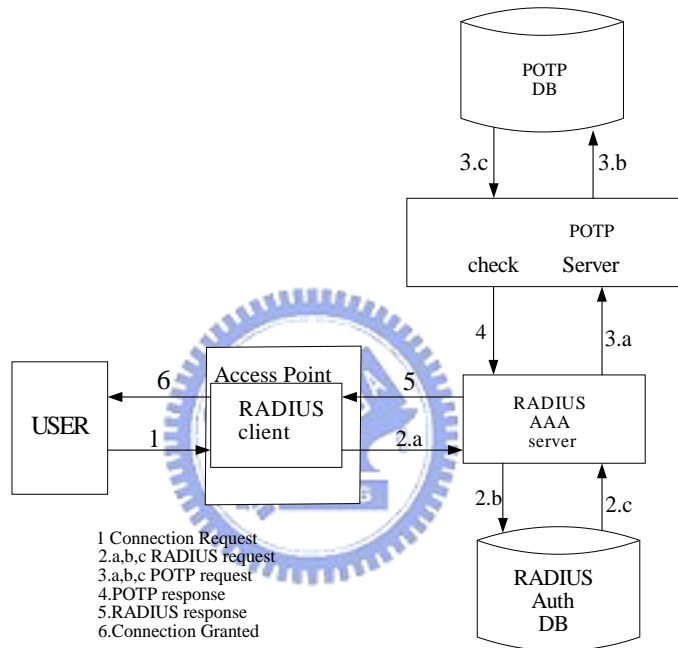


Figure 3-2 Protocol Architecture

In figure 3-2, both the PS and AS maintain their own databases which are known as authentication databases. These databases record all the user credentials before the authentication process is commenced. During authentication PS and AS utilize their corresponding authentication databases for user credential verification.

3.5 Authentication Message Flow

The Figure 3-3 given below is the authentication message flow of our protocol.

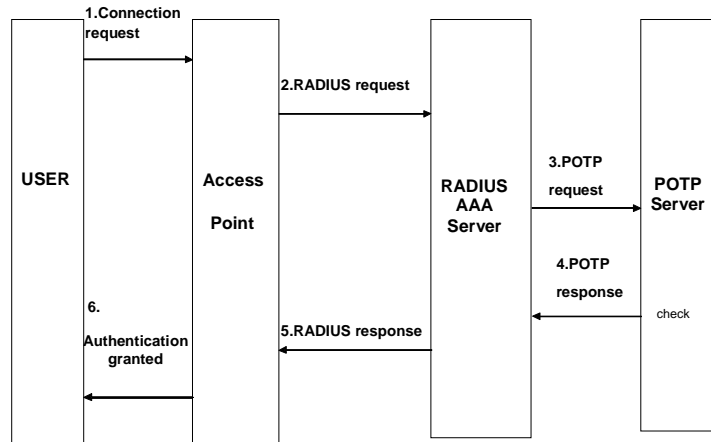


Figure 3-3 Authentication Message Flow

3.5.1 Authentication Message Flow Description

Initially the user enters the username, and at the password prompt, the user enters a PASSCODE (A two-part password, consisting of a memorized personal identification number (PIN) followed by the current token code displayed on the token).

The AP forwards this information to the AS for user credentials verification. When AS receives the User information, it checks its database regarding the user details and then it sends the user details to the PS.

The PS examines its database for the username and token number of the user's token. If the token number matches, the user is verified (detailed explanation is given in section 3.6.3) and the user's authentication permission is sent to the AS.

The AS receives the User verification message from the PS and it sends the Authentication Granted message to the AP.

The AP forwards the authentication success message to the user and now the user is authenticated. The detailed explanation of the protocol architecture is given in the next section.

3.6 Secure and Fast OTP Authentication Protocol

To provide a Secure and Fast authentication scenario, we have designed this protocol. We divide our proposed protocol into three phases: Registration phase, Login Phase and Subsequent Authentication phase. We explain these three phases in the following sections.

3.6.1 Registration Phase

(i) User and Password server: $U \leftrightarrow P : U, \text{PIN}, \text{seed}$

During this process the client registers himself with the PS, which means that the User details (Username, PIN, seed) are stored in the database of the PS. This process is done offline when the token generator is purchased. This process should be done with proper care and concern, because the whole authentication process is based on proper user credential registration with the AS and PIN, seed are stored in the PS.

(ii) User and Authentication server: $U \leftrightarrow AS : \text{User name}$

The Username and physical details like user registration date and time, type of user registration, etc regarding the user are stored in the AS. This process is done offline when the token generator is purchased.

(iii) Authentication server and Password server: $AS \leftrightarrow PS : \text{Pre shared key}$

AS and PS mutually trust each other, thru the pre shared key $K_{AS,PS}$ and this key is chosen to be highly secure and it is also 16 bit hexa decimal string. This key forms a relationship between the Authentication server and Access Point.

(iv) Authentication server and Access Point: $AS \leftrightarrow AP : \text{Pre shared key}$

$K_{AS,AP}$ is the pre shared key between AS and AP and this key is chosen to be highly secure and it is also 16 bit hexa decimal string. This key establishes a relationship between the AS and AP.

3.6.2 Login Phase

Message (i) : User \rightarrow Access Point

$$U \rightarrow AP: U, PS \{ H(PIN, T), H(U, T), N_u \}_T$$

The user attempts to login by entering his username, PS name, with a hash of PIN (memorized by the user) and token number (generated by the token generator), and a hash of username and token number and nonce, all encrypted by the token number. This encrypted message is sent to the AP thru wired or wireless channel.

Message (ii) : Access Point to Authentication server

$$AP \rightarrow AS: U, PS \{ H(PIN, T), H(U, T), N_u \}_T$$

The AP receives the encrypted message form the user and then forwards the received message to the AS

Message (iii) : Authentication server \rightarrow Password server

$$AS \rightarrow PS: \{ H(PIN, T), H(U, T), N_u \}_T$$

The AS receives the encrypted messages from the AP, identifies that it is a PS request message and then it checks the username in its database and the details. Then the message is forwarded to the PS.

The PS gets the message form the AS, and then using the token number it verifies the user database and confirms the user for authentication. The token number verification process of the PS is detailed in section 3.6.3,

Message (iv) : Password server \rightarrow Authentication server

$$PS \rightarrow AS: \{ H(U, T), T, N_s \}_{K_{AS,PS}}$$

After PS confirms that the token number is of the real user, then it includes the nonce N_s and encrypts the message with $K_{AS,PS}$ then sends the message to the AS to perform further process of authentication.

AS receives the message from the PS and decrypts using $K_{AS,PS}$ which is the pre-shared

between AS and PS. Then AS sends two parts (part 1 is the subset of part 2) of the same message to the AP.

Message (v): Authentication server \rightarrow Access Point

AS \rightarrow AP:

$$\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \}_{K_{AS,AP}}$$

Here, the part 1 of Message (v) is $\{ U, Sid, H(K_{SS}, N_u, N_s) \}$ which is to be stored in the AP for subsequent authentication purpose. K_{SS} is the session key included in the message which is used for the forth coming session management. The expiration time of the session key is solely decided and set by the AS, depending on the user's requirement. Sid is the session identifier, which is a unique number that a AS assigns to the user for the duration of that user's visit. This id consists two parts, in which the first part says the AS name who had given this id and the next part identifies the type of user account within the issuing domain and this id is used along with the session key for session management.

Part 2 of the Message(v) is $\{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$, which is the session ticket containing the username, session identifier, AS name, hash of the two nonce values along with the session key, and a time stamp (indicating the session expiry time and date). All these components are encrypted by the token number.

Both part1 and part2 of the messages are jointly encrypted using $K_{AS,AP}$ which is the Pre shared Key between AS and AP. The encrypted message is sent to the AP.

When AP receives the two part message from the AS, it first uses its Pre shared Key $K_{AS,AP}$, to decrypt the message. Then it stores the part1 of the message i.e $\{ U, Sid, H(K_{SS}, N_u, N_s) \}$ with itself and sends the part2 of the message $\{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$ (which is the session ticket) to the user.

Message (vi): Access Point \rightarrow User

$$AP \rightarrow U : \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$$

Now the user uses his token number to decrypt the session ticket. This session ticket is stored in the secure cache on the client in the memory (not in the disk), for subsequent authentication purpose. The user is authenticated and he is authorized to handle the resources within the system.

3.6.3 Password server Token Number Verification

This mechanism is based on counter based OTP authentication scheme. When a token is first initialized, the internal counter is set to zero. Each time an event occurs, as when the user requests a new password, the counter is incremented and the incremented value is used as the input value. This value is then encrypted with the seed using a Proprietary Encryption Algorithm (which is considered to be a secure encryption algorithm) and the result becomes the one-time dynamic password. Likewise, the user's account on the server also has a counter. It is initialized to zero when the account is created, and is incremented each time the user is authenticated.

During authentication, when the PS receives this message $\{ H(PIN,T), H(U,T), N_u \}_T$ from the AS, the PS first checks for the corresponding seed in its database, using its exhaustive search mechanism. If the seed is identified, the counter is incremented and the incremented value is used as the input value, which is encrypted with the seed of the server and the result is the token number.

Using this token number the server decrypts the message from the AS. If the token number is from the real user(token numbers are same) then the PS could do the decryption else it cannot .After decryption the PS checks the PIN value in the message with that of the PIN value in its database and then confirms that the token number is from the real user.

The counter based OTP authentication scheme is obviously much easier to have synchronization between user and PS, but if necessary, the server may maintain synchronization between the two counters by adjusting its counter to match the counter in the user's token.

3.6.4 Subsequent Authentication phase

User \rightarrow Access Point

U \rightarrow AP: $\{H(U, Sid, N_u)\}_{K_{ss}}$

For the next time if the user needs to login the system, he need not to login again with his complete user credentials, instead he can just use his session key K_{ss} to encrypt the hash value of username, Sid and nonce N_u and send to the AP, which takes the responsibility of the AS in performing authentication this time, thereby reducing the burden of authentication from the AS.

Access Point \rightarrow User

AP \rightarrow U: $\{H(U, N_u)\}_{K_{ss}}$

Now the user is authenticated again, by the AP.

3.7 Authentication Goals

Our protocol is designed on the basis of the following authentication goals. These goals are general and the goals achievement is subjected to the wireless network environment, type and efficiency of principals used and also in proper implementation of the protocol. Each message component used in our design obeys these goals and the security proof presented in Appendix. The goals are mentioned as G1, G2, G3, G4, G5 and G6.

G1. Ping authentication : P knows whether an interlocutor Q is alive

P believes Q says X

G2. Entity authentication: Q said something relevant to their present conversation

P believes (Q says $F(X, Y_p) \wedge fresh(Y_p)$)

G3 Secure key establishment: P believes that he has a good key to communicate with Q

P believes $P \xleftarrow{k} Q, P \xleftarrow{k} Q \equiv P \xleftarrow{k} Q \wedge P \text{ has } k$

G4. Key freshness: P believes a key to be fresh

P believes fresh(k)

G5. Mutual understanding of shared keys: P can establish that Q has sent a key as an unconfirmed secret between the two of them

$$P \text{ believes } Q \text{ says } (Q \xleftarrow{k} P)$$

G6. Key confirmation: P believes that Q proved to receive a previously shared secret key between the two of them

$$P \xleftarrow{k} Q \equiv Q \xleftarrow{k} P \wedge P \text{ has } k$$

3.8 Summary

In this section we explain the necessity of proposing our Secure and Fast OTP Authentication Protocol continued by presenting a detailed view, with a sample network topology of a large scale enterprise wireless environment. Then we explain the architecture of our protocol followed by a detailed description of the authentication message flow thru a diagram. Next we explained the execution phases of our protocol. Finally we present our authentication goals, which are to be proved with our protocol design, in the following sections.



Chapter 4

Security and Privacy Analysis

Our protocol is analyzed for its security features. Various types of attacks are evaluated and verified with our protocol to check for its persistence. We have done this security analysis based on [21]. Moreover a few references have been chosen from [19] and [20].

4.1 Security Analysis

In any protocol architecture, one of the most important factors to be considered is the security analysis portion. In this thesis we analyze our protocol design for its security temperament, by assuming common wireless attacks, which are well known and to be seriously handled attacks. Some of these attacks are mentioned below in the following and evaluated with our protocol.

(i). Replay attack prevention

This proposed protocol provides security against, passive attacks, based on replaying captured reusable user password. The token number (One Time Password) changes every time the user needs it, It is not possible for an attacker to maliciously or fraudulently repeated or replay the messages. Since the OTP expires after one time use, it cannot be reused.

(ii) Eavesdropping attack prevention

In our proposed protocol, during the state of data transfer in the first part of the login phase all the three messages are encrypted by using OTP itself. Hence, user – AP, AP – AS and AS – PS communication links are protected. In the return path of the login phase all the three messages are protected by one-way hashed function. The irreversible property of the one-way hashed function prevents an eavesdropper to get back the encryption data; even if the authentication messages are stolen.

(iii) Dictionary attack prevention

In our protocol, since there is only OTP scheme is employed, which changes every time, it's impossible for an attacker to try "every word in the dictionary" as a possible password, for an encrypted message. So dictionary attack could be prevented using our protocol.

(iv) Brute force attack prevention

Included with the OTP, the inducement of nonce in every session ensures that the data is different for every new session. Since the nonce value is fresh the message is also considered fresh, Along with the OTP, the nonce inclusion also plays a pivotal role in reducing the simplicity of Brute force attack. The other factor is the key size, which is large enough to increase further the simplicity of Brute force attack.

(v) 802.1X Identity theft attack prevention:

This attack, capturing user identities from clear text 802.1X Identity Response packets could be overcome when our proposed protocol is implemented. Since the only two clear texts available are the Username and the PS name. Though the attacker gets them it is of no use to him. Since user and the PS alone could handle the OTP and without knowing it, it is impossible for the user to crack the messages. Moreover, the encryption algorithm is strong enough to prove the secrecy,

(vi) Man-in-the-Middle attack prevention

Our design can surmount this attack well. If the intruder arrives in between user and AP i.e Message (i), or in between AP and AS i.e Message (ii), or in between AS and PS i.e Message (iii), he cannot break into the messages, because they are encrypted by OTP. The returning phase from PS to AS or in between AS to AP or in AP to user, if the intruder tries to hang in, he cannot get-in thru the messages as they are strongly encrypted by shared keys which are hard enough to break in. Hence, we mention that our protocol could surmount the man-in-the-middle attack well. Moreover, this feature is considered to be an important aspect of our protocol design and this attack is considered as one of the most common wireless attacks.

(vii) User impersonation attack prevention

Since the seed, PIN and the token numbers are known only to the user and the PS, the impersonation attack could be highly succumbed. For instance if an intruder disguises himself as the real user and enters his username, the PS will identify the illegal user, since the seed, PIN and the token number all should be matched only for an authorized user, else the user will be treated as an illegal user.

(viii) Ensuring Data confidentiality

In the Login phase from U \rightarrow P, when user sends the message to AP, AP to AS, AS to P all are encrypted by token number itself, which the OTP. Seed value is not sent during the message delivery. Next, when P sends the message to user, it does not carry any seed or PIN values, and the two shared keys $K_{AS,PS}$, $K_{AS,AP}$ are used for encryption, which are suitably long enough to ensure security. The encryption algorithm used in IDEA (International Data Encryption Algorithm) which is one of the strongest encryption standards in existence. The key length is 2^{128} (128 bits) which is hardly possible even for a brute force attacker to break in. So user data privacy is maintained and data confidentiality is achieved.

(ix) Providing data integrity

In our protocol, modification of data could be blocked. Since the messages are encrypted using OTP it is impossible to retrieve it and use it again to hack into the data. In case of any data loss due to DoS attacks, the PS is well equipped with back end data base along with data recovery software, which provides a good back-up for the data within the system.

(x) Enabling mutual authentication

In our design, though the user authentication is our main focus, both user and PS could mutually authenticate each other. The seed value and PIN, forms the key between the user and PS and the token Number (One Time Password) proves each other's identity. So in our protocol, mutual authentication is achieved between user and PS.

(xi) Forward Secrecy achievement

Our protocol can be suited to handle, the forward secrecy issue, as the disclosure of the long-term secret keying material that is used to derive the agreed keys, does not compromise the secrecy of agreed keys from earlier runs. In our protocol no key is derived from other key, and OTP itself is used as a key, which changes every time gives a guarantee that forward secrecy is maintained. For example, if an attacker retrieves any message encrypted by OTP is useless to him for the next time use, and if the attacker hacks any message encrypted by the shared keys, it is not possible for him to decrypt, since the keys are pre shared and considered to be with strong encryption standards.

4.2 Security Proof - using SVO Logic

The proposed protocol is verified for its correctness and security features by using SVO logic [16],[17],[18] which is a noted security proof venture for authentication protocols. This cryptographic protocol is also considered for its elegant features, which are scrutinized as well. The complete security proof is explained in Appendix for reference.

4.3 Security Properties – Comparison table

Table 4-1 shows a comparison of our proposed protocol with other existing security protocols on the basis of their security properties. Our protocol's security features are mentioned in the final column.

Table 4-1 Security Properties – Comparison

Protocol	RADIUS	S/KEY OTP	KERBEROS	Our Protocol
Replay Attack	X	√	√	√
Eavesdropping attack	X	√	⊠	√
Dictionary attack	X	√	X	√
Brute Force Attack	X	⊠	X	√
802.1X Identity Theft Attack	⊠	√	√	√
Man-in-the-middle attack	X	X	X	√
User Impersonation attack	X	⊠	⊠	√
Data confidentiality	⊠	√	√	√
Data integrity :	⊠	√	√	√
Mutual authentication	√	√	√	√
Forward Secrecy	X	⊠	X	√

Notation: √ Satisfied ⊠ Partially Satisfied × Not Satisfied

In the Table 4-1 we have mentioned the comparison of security properties of our protocol with that of other authentication protocols. In section 4.1 we explained all these above mentioned attacks and our protocol's ability to overcome those attacks. Here, for example, we consider impersonation attacks and replay attacks and prove their correctness thru SVO Logic. The detailed proof of these attacks is mentioned in the Appendix section elaborately. The details mentioned in the comparison chart may vary depending upon the network environment and implementation

factors.

4.4 Summary

This section presents an overall analysis of security through which we provide the possible attacks that could be surmounted well by our protocol. The security proof is collectively mentioned in the Appendix section. The security properties comparison table explains a comparison of different security features of our protocol with other authentication protocols. Hence our protocol is demonstrating its security fitness; we ensure that our protocol could be efficiently implemented in real time wireless networks.



Chapter 5

Performance Analysis

The overall performance of our Protocol is analyzed, to determine the computational efficiency and communications efficiency. Chen Hao, et.al [13], and Boyd and Mathuria [22], explain about the two types of efficiency, prior is the computational efficiency and the later is communications efficiency. Computational efficiency deals with the total number of computations that the principals require to engage in, for the whole protocol run.

Communications efficiency focuses on the number and length of messages that should be sent and received during the running of a protocol. Including this, we estimate the storage cost, which gives the details about the total number of keys and passwords stored in each principal. As a whole we focus on improving both the efficiency's and reducing the storage cost of the protocol execution.

As mentioned, the number of Hash operations, number of RNG operations, number of encryption and decryption operations and the total number of authentication steps required for the protocol execution determines the computational efficiency of the protocol. The encryption standards used in our protocol are symmetric-key based, which ensures the fast authentication nature of our protocol. The usage of limited number of symmetric-key encryption operations proves that, resources are limitedly used and protection of data, in a secured manner.

Since, the total number of authentication step is 6, which is another factor to describe that our protocol's execution speed is faster compared to other protocols that are taken into consideration and proves that the communications efficiency is fair. Thus, we describe our protocol is maintaining a reasonable computational efficiency and fair communications efficiency. Table 5.1 gives the comparison of computational efficiency and communications efficiency of our protocol for one session, with the related works. In the comparison table, we ignore mentioning RADSEC protocol because, it is involving certificate based authentication scheme. The

complexity of implementing certificate based schemes forms the next reason. The S/Key OTP authentication protocol is also not tabulated because; it is specifically designed for general purpose OTP authentication structures, but not for enterprise wireless network environments.

Table 5-1 Comparison of Computational & Communications Efficiency

Protocol	Entity	RADIUS	S/KEY OTP	KRBS	Our Protocol
No.of Hash Operations	User	-	N	1	1
	AP/NAS/ AS	1	-	-	-
	AAA/ TGS	1	-	-	1
	POTP/ KDC/ SKS	-	2	-	1
No.of RNG Operations	User	-	1	1	1
	POTP/ SKS	-	1	1	1
No.of Encryption operations	User	-	-	3	1
	AP/NAS/ AS	-	-	2	-
	AAA/ TGS	-	-	2	1
	POTP/ KDC	-	-	1	1
No.of Decryption operations	User	-	-	3	1
	AP/NAS	-	-	1	1
	AAA/ TGS	-	-	1	-
	POTP/ KDC	-	-	1	-
No. of Authentication steps		7	4	8	6

Components of Table 5-1

AP/NAS =Access Point/ Network Access Server (in RADIUS), AAA = RADIUS server, POTP = OTP Server, TGS = Ticket Granting Server, AS = Kerberos Authentication Server, KDC= Key Distribution Centre(verifier)

5.1 Storage Cost

This cost describes usage of total number of keys and passwords that are stored in each principal which are essential for the protocol execution. Since we have achieved fair and reasonable Computation efficiency and Communications efficiency, we consider the storage cost also to be moderate and quite equaling the normal standards. Table 5-2 gives the comparison of storage cost of our protocol for one session, with the related works.

Table 5-2 Comparison of storage cost

Protocol	Entity	RADIUS	S/KEY OTP	KERBEROS	Our Protocol
Password stored and used	User	1	1	1	1
	AAA/ TGS	1	-	-	-
	POTP/ KDC/ SKS	-	-	-	1
Keys stored and used	User	-	-	2	3
	AAA/ TGS	1	-	1	-
	POTP/ KDC/ SKS	-	-	3	2

Components of Table 5-2

AP/NAS =Access Point/ Network Access Server (RADIUS), AAA = RADIUS server, POTP = OTP Server, TGS = Ticket Granting Server, KDC= Key Distribution Centre(verifier),SKS = S/Key server

Table 5-2, gives an account of the storage cost for each protocols and though our protocol's storage cost level seem to be comparatively more. In concerning with the higher security standards, this feeble incrementation of storage cost level, could be ignored in real time applications.

5.2 Fitness function

To determine the Security fitness and Efficiency fitness we include the fitness function. This function evaluates the Efficiency Criterion which determines how efficient the protocol is. The detailed study of fitness function is elaborated in [13], which explains the methods of implementing the fitness function and analyzing with the protocol execution. Based on this study we verify the fitness function of our Protocol. [21] and [22] presents the related study with the evaluation of fitness function.

The fitness function is denoted by $f(P1)$ which could be calculated by concatenating the security fitness function, s and an efficiency fitness function, e .

$$f(P1) = s(P1) + e(P1)$$

5.2.1 Security fitness function (s)

Security fitness function $s(P1)$ is defined as follows:

$$s(P1) = \sum_{i=1}^N (\sigma + \delta(i)) \times G(P1, i)$$

Here N is the maximum number of messages we allow in any protocol; $G(P1, i)$ is the number of new required security goals that message i of $P1$ achieves; σ is, a large constant that weights security much more heavily than efficiency and the $\delta(i)$ gives are weights among the individual messages. These weights $\delta(i)$ were chosen to represent the two strategies for finding protocols. They are early credit (EC) for achieving goals early in the protocol and uniform credit (UC). The values of EC varies accordingly with $\delta(i)$, but UC remains constant and maintains uniformity. Table 5-3 given below explains the weighting strategy for our protocol with total number messages $N=6$.

Table 5-3 Weighting Strategies for $N = 6$

Strategy	Weight					
	$\delta(1)$	$\delta(2)$	$\delta(3)$	$\delta(4)$	$\delta(5)$	$\delta(6)$
EC	320	160	80	40	20	10
UC	160	160	160	160	160	160

5.2.2 Efficiency fitness function (e)

The function $e(P1)$ can be considered as the sum of fitness functions, and we estimate this function as,

$$e(P1) = m(P1) + c(P1) + r(P1)$$

Including this, the fitness function $m(P1)$ punishes protocols with many messages.

$$m(P1) = \mu \times M(P1)$$

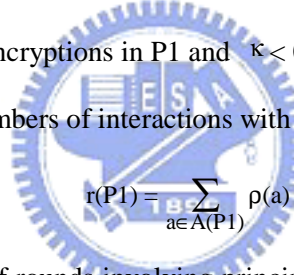
where $M(P1)$ is the highest index of a message of $P1$ that contributes new goals. We assign the weight for $\mu < 0$ for this reason.

Function $c(P1)$ punishes protocols with more encryption.

$$c(P1) = \kappa \times C(P1)$$

where $C(P1)$ is the number of encryptions in $P1$ and $\kappa < 0$ is the weight we give to this

The function $r(P1)$ punishes numbers of interactions with particular principals.


$$r(P1) = \sum_{a \in A(P1)} \rho(a) \times R(P1,a)$$

where $R(P1, a)$ is the number of rounds involving principal a in $P1$, $A(P1)$ is the set of principals in $P1$ and $\rho(a) < 0$ are weights which allow us to declare that interactions with some principals (for example, the server) are worse than others.

5.2.3 Results of Fitness functions

In this section we formalize the results of direct application of the techniques mentioned above and derive three-party symmetric key distribution protocols.

(i) Assumptions

The three parties involved in our protocol are User (U), Authentication server (AS) and Password server (PS). Access Point (AP) is considered for the subsequent authentication purpose only. All the basic assumptions are mentioned in Appendix, Security Proof section.

(ii) Goals

The Goals mentioned previously in section.3.7 could be explained in the basis of relationship between, user and PS, AS and PS, relationship between AS and AP. The further discussion of goals is given below in the Results analysis section (section 5.2.4).

(i) User and Password server

User has PIN, seed

Password server has PIN, seed

(ii) Authentication server and Password server

AS has $K_{AS,PS}$ AS believes $AS \xleftrightarrow{K_{AS,PS}} P$

P has $K_{AS,PS}$ P believes $AS \xleftrightarrow{K_{AS,PS}} P$

AS believes P has $K_{AS,PS}$

P believes AS has $K_{AS,PS}$

(iii) Authentication server and Access point.

AS has $K_{AS,AP}$ AS believes $AS \xleftrightarrow{K_{AS,AP}} AP$

AP has $K_{AS,AP}$ AP believes $AS \xleftrightarrow{K_{AS,AP}} AP$

AP believes AS has $K_{AS,AP}$

(i) User and Password server

User has PIN, seed

Password server has PIN, seed

(ii) Authentication server and Password server (P)

AS has $K_{AS,PS}$ AS believes $AS \xleftrightarrow{K_{AS,PS}} P$

P has $K_{AS,PS}$ P believes $AS \xleftrightarrow{K_{AS,PS}} P$



As believes P has $K_{AS,PS}$

P believes As has $K_{AS,PS}$

(iii) Authentication server and Access point.

AS has $K_{AS,AP}$ AS believes $AS \xleftrightarrow{K_{AS,AP}} AP$

AP has $K_{AS,AP}$ AP believes $AS \xleftrightarrow{K_{AS,AP}} AP$

AP believes AS has $K_{AS,AP}$

5.2.4 Result Analysis w.r.to Weight strategy

(i) First search:

We consider the total number of messages $N = 6$. The weights we used for the fitness function are given in Table 5-3. We utilize the EC strategy to find whether the protocol is able to satisfy security requirements quickly.



Table 5-4 Fitness function weightings for the first search

σ	4000	Correctness is more important than efficiency
$\delta(i)$	EC	Referring Table 5
μ	-300	
κ	-200	
$\rho(U)$	-100	User
$\rho(AP)$	-50	Access Point
$\rho(AS)$	-50	Authentication server
$\rho(PS)$	-50	Password server

We give the basic message flow of Login Phase in Figure 5-1

Message (i) : $U \rightarrow AP$: $U, PS \{ H(PIN, T), H(U, T), N_u \}_T$

Message (ii) : $AP \rightarrow AS$: $U, PS \{ H(PIN, T), H(U, T), N_u \}_T$

Message (iii) : $AS \rightarrow PS$: $\{ H(PIN, T), H(U, T), N_u \}_T$

Message (iv) : $P \rightarrow AS : \{ H(U,T), T, N_s \}_{K_{AS,PS}}$

Message (v) : $AS \rightarrow AP :$

$\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \}_{K_{AS,AP}}$

Message (vi) : $AP \rightarrow U : \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$

Figure 5-1 Basic message flow of Login Phase

We present one of the symmetric key protocols formulated by the weight strategy method in the figure 5-2, given below. Only the core security factors in our protocol are presented. Hence we have removed components from the description that do not contribute to the goals and also the, redundant components, which are included twice or more in one message have also been removed to formulate a new protocol.

Message (i) : $U \rightarrow AP : U, P \{ H(PIN, T), H(T), N_u \}_T$

Message (ii) : $AP \rightarrow AS : U, P \{ H(PIN, T), H(T), N_u \}_T$

Message (iii) : $AS \rightarrow P : \{ H(PIN, T), H(T), N_u \}_T$

Message (iv) : $P \rightarrow AS : \{ H(U, T), T, N_s \}_{K_{AS,PS}}$

Message (v) : $AS \rightarrow AP :$

$\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \}_{K_{AS,AP}}$

Message (vi) : $AP \rightarrow U : \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$

Figure 5-2 A symmetric key protocol found in the first search

In figure 5-2 we have removed Username (U) in M1 and further removed in M2 and M3 as well. Moreover, the search limits could be extended to achieve more shrink version of the protocol. The same principle could be followed in forming new symmetric key protocols with further fewer server interactions, by increasing the values of $\rho(U)$, $\rho(AP)$, $\rho(AS)$ and $\rho(P)$

5.3 Summary

This section presents a detailed analysis of performance factor of our protocol. First we present an analysis of protocol efficiency in terms of communications efficiency and computational efficiency. We give comparison of both of these efficiencies with that of the other authentication protocols. Along with this we provide a comparison of storage cost of our protocol with other protocols and in both these analysis our protocol seems to be comparatively good and it is identified thru the comparison table. Finally, we conclude this section with an analytical proof of fitness function which provides the goodness of our protocol to shrink its message components and the obtained result message is presented in Figure 5-2.



Chapter 6

Special Features of Our Protocol

(i) Robustness

Our Protocol is comparatively robust in nature. Since many types of attacks are handled well by our design, the essential robustness is embedded within the structure. Moreover, the strength of the protocol lies on OTP alone, which provides vigor to our protocol during the execution phase.

(ii) Extensibility

We believe that our protocol is extensible in nature and adaptive for any further enhancements to be made within its structure. Hence, this future is an asset to any wireless network infrastructure.

(iii) Fast Authentication

As mentioned in [13], “Reducing one message from a five-message protocol represents a 20% reduction in the number of messages and possibly a similar amount of reduction of the overall running time of the protocol”. Compared to the protocols mentioned in our related work, our protocol proves fast authentication since the number of authentication steps are 6.

(iv) Secure Authentication

Along with fast authentication, our protocol provides high security during execution. Since the usage of one time password helps in high password security, which is the prime motto behind our design and also many wireless attacks could be handled with ease and high efficiency.

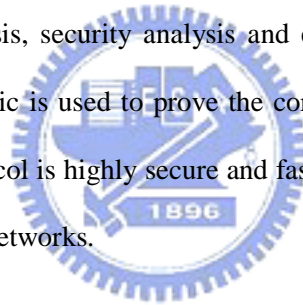


Chapter 7

Conclusion and Future Work

7.1 Conclusion

In this paper, we have presented a robust, efficient, secure and fast OTP authentication protocol which could effectively handle some of the most common wireless attacks. Our protocol is easy to implement intra domain for which it is designed and in the future the design strategy could be enhanced to cross realm wireless architecture as well. In the proposed protocol the total number of authentication steps is 6, which means that fast authentication is ensured and thus makes our protocol efficient when compared with the related research works, whose authentication steps are more. The OTP scheme used in the protocol helps to maintain high security. The theoretical analysis, security analysis and efficiency analysis are done to test the overall protocol flow. SVO logic is used to prove the correctness of our design and the security proof determines that our protocol is highly secure and fast in nature with OTP authentication, for large scale enterprise wireless networks.



7.2 Future Work

Our protocol could be extended further in future, to be suited for cross realm roaming and authentication.

Currently, our protocol is designed for large scale enterprise wireless networks for inter domain environment. It is possible to implement our protocol to a cross realm environment also. Since easy to initialize and our design will be well suited for this type of architecture. Its Fast authentication nature could facilitate the seamless roaming property in the cross domain roaming, which is considered to be one of the key issues in the wireless networks today. Hence, even when the network environment extends further, each network domain could be connected with another network domain with, seamless and secure authentication wireless feature, thru our protocol.

Furthermore, in cross realm architecture, multiple authentication servers will be employed. Hence the security issues have to be more diligently taken into account. The suggested structure is organized thru implementing an highly efficient password server in the middle of each cross realm architecture which could effectively handle the entire authentication processes, held all along the cross realm architecture. Our protocol could be well suited for the cross domain structure. The implementation scenario could be complex in real time and when included with some additional security features related to the cross domain networks, our protocol could prove its ability.



References

- [1] Rigney, C., Willens, S., Rubins, A., Simpson, W., Remote Authentication Dial in User Service (RADIUS). IETF RFC 2865, June 2000.
- [2] Rigney, C., Willats, W., P. Calhoun., "RADIUS Extensions", IETF RFC 2869, June 2000.
- [3] The Open Group, "RADSEC, a secure, reliable, RADIUS protocol", whitepaper. May 2005.
<http://www.open.com.au/radiator/radsec-whitepaper.pdf>
- [4] P. Calhoun, J. Loughney, E. Guttman, G. Zorn, and J. Arkko, "Diameter Base Protocol", IETF RFC 3588, September 2003.
- [5] S. Winter, M. McCauley, S. Venaas, RADSEC Version 2 – "A Secure and Reliable Transport for the RADIUS Protocol", draft-winter-radsec-00, June 2007.
- [6] B. C. Neuman, Theodore Ts'ó, "Kerberos : An Authentication service for computer networks", IEEE Communications Magazine, Volume 32, Number 9, pages 33-38, September 1994.
- [7] Kwang-Hyun Baek, Sean W. Smith, David Kotz "A Survey of WPA and 802.11i RSN Authentication Protocols" Technical Report TR2004-524, Dartmouth College Computer Science, November 2004.
- [8] N. Haller, P. Nesser, M. Straw, A One-Time Password System, February 1998, IETF, RFC 2289.
- [9] C. Metz, OTP Extended Responses, 1997, IETF, RFC2243.
- [10] N. Haller. The S/KEY one-time password system. Bellcore, February 1995. IETF RFC 1760.
- [11] L. Lamport. Password authentication with insecure communication. Communications of the ACM, 24:770-772, 1981.
- [12] Chris J. Mitchell and Liqun Chen, Comments on the S/KEY user authentication scheme, ACTS project AC095 (ASPECT).
- [13] Chen Hao, John A. Clark, Jeremy L. Jacob, Synthesizing Efficient and Effective Security Protocols, ARSPA 2004 Preliminary Version

- [14] Kenji Imamoto, Kouichi Sakurai, Design and Analysis of Diffie-Hellman-Based Key Exchange Using One-time ID by SVO Logic, *Electronic Notes in Theoretical Computer Science*, 79-94, 2005.
- [15] Kenji Imamoto, Kouichi Sakurai, A Design of Diffie-Hellman Based Key Exchange Using One-time ID in Pre-shared Key Model. *IEEE, 18th International Conference of Advanced Information Networking and Application (AINA'04.)*.
- [16] Paul F. Syverson and Paul C. van Oorschot. A Unified Cryptographic Protocol Logic. Submitted for Publication. Parts of this paper appeared in preliminary form in [SVO94].
- [17] Paul F. Syverson and Paul C. van Oorschot. On Unifying Some Cryptographic Protocol Logics. In *Proceedings of the 1994 IEEE Computer Society Symposium on Research in Security and Privacy*, pages 14-28. IEEE Computer Society Press, Los Alamitos, California, 1994.
- [18] Paul F. Syverson. Knowledge, Belief, and Semantics in the Analysis of Cryptographic Protocols. *Journal of Computer Security*, 1(3):317-334, 1992.
- [19] D. McDonald and R. Atkinson, "One-Time Passwords In Everything (OPIE): Experiences with Building and Using Stronger Authentication," *Proceedings of the Fifth USENIX UNIX Security Symposium*, 1995.
- [20] N. Haller, "The S/KEY One-Time Password System, " *Proceedings of the Internet Society Symposium on Network and Distributed System Security*, pp.151-158, 1994.
- [21] John A. Clark and Jeremy L. Jacob "Protocols are programs too: the meta-heuristic search for security protocols" *Science Direct, Information and Software Technology* 43 (Nov 2001), pp. 891-904.
- [22] John A. Clark and Jeremy L. Jacob, Search for a solution: Engineering tradeoffs and the evolution of provably secure protocols, in: *Proceedings of 2000 IEEE Symposium on Security and Privacy (2000)*, pp. 82-95.

Appendix

Security Proof by using SVO Logic

All the notations used in our protocol are already defined in Table 1, and other symbols and rules can be referred in [13][14][15].

Our protocol consists of Registration Phase, Login Phase and Subsequent Authentication phase.

Since the Registration Phase is done offline, we provide security analysis and proof for Login Phase and Subsequent Authentication phase only.

Preliminaries of SVO Logic

In SVO logic we categorize its sections in the following

- i. SVO notations
- ii. SVO axioms
- iii. Authentication goals



SVO Notations

Notations	Explanations
P believes X.	The principal P may act as though X is true
P received X	:The principal P received a message containing X to P , who can read and repeat X .
P said X	The principal P at some time sent a message including X .
P says X	P must have said X since the beginning of current epoch.
P has X	Initially available to P , Received by P , Freshly generated by P, Constructible by P from the above.
P controls X	P has jurisdiction over X. The principal P is an Authority on X and should be trusted on this matter.

$\text{fresh}(X)$	X has not been sent in any message prior to the current protocol run.
$P \xleftrightarrow{k} Q$	P and Q may use the shared key k to communicate. k will never be discovered by any principal but P, Q, or a principal trusted by P or Q.
$\{M\}_k$	Encryption of message, M, using key k. Encrypted messages are uniquely readable and verifiable as such by holders of right keys.
$\langle X \rangle_{*P}$	Used for messages that P doesn't know or recognize X
$P \xleftrightarrow{k} Q \equiv P \xleftrightarrow{k} Q \wedge P \text{ has } k$	

SVO Axioms

Belief Axioms

1. $(P \text{ believes } \varphi \wedge P \text{ believes } (\varphi \rightarrow \psi)) \rightarrow P \text{ believes } \psi$
2. $P \text{ believes } \varphi \rightarrow P \text{ believes } (P \text{ believes } \varphi)$

Receiving Axioms

3. $(P \text{ received } \{X\}_k \wedge P \text{ has } k) \rightarrow P \text{ received } X$

Possession Axioms

4. $P \text{ received } X \rightarrow P \text{ has } X$
5. $P \text{ has } (X_1, \dots, X_n) \rightarrow P \text{ has } X_i \text{ for } i = 1, \dots, n$

Freshness Axioms

6. $\text{fresh}(X_i) \rightarrow \text{fresh}(X_1, \dots, X_n) \text{ for } i = 1, \dots, n$
- If a message is fresh, a message including the message is also fresh

Jurisdiction and Nonce-Verification Axioms

7. $P \text{ said } (P \text{ controls } \varphi \wedge P \text{ says } \varphi) \rightarrow \varphi$

Symmetric Goodness Axiom

8. $P \xleftrightarrow{k} Q \equiv Q \xleftrightarrow{k} P$

Saying Axioms

9. $P \text{ said } (X_1, \dots, X_n) \rightarrow P \text{ said } X_i \wedge P \text{ has } X_i \text{ for } i = 1, \dots, n$

Source Association Axioms

10. $(P \xleftrightarrow{k} Q \wedge R \text{ received } \{X \text{ from } Q\}_k) \rightarrow (Q \text{ said } X \wedge Q \text{ has } X)$

Procedure of Security Analysis by SVO Logic

This process commence with a list of assumptions used in the Logic. [13][14][15]

Login Phase

Initial state assumptions

Detailed assumptions about initial status

User

A_1 : U has (U,AS,PS)

A_2 : U has (Seed,PIN)

A_3 : U believes U $\xleftarrow{\text{Seed,PIN}}$ PS

A_4 : U has (N_u)

A_5 : U believes fresh (N_u)

A_6 : U believes $\varphi(U)$

A_7 : U believes $\varphi(N_u)$

A_8 : U believes PS controls PS $\xleftarrow{K_{AS,PS}}$ AS

Authentication server

A_9 : U has (U,AP,Sid, N_u , K_{SS})

A_{10} : U believes U $\xleftarrow{K_{SS}}$ PS

A_{11} : U has K_{SS} and U believes fresh(K_{SS})

A_{12} : AS has $K_{AS,AP}$

A_{13} : AS believes fresh($K_{AS,AP}$)

A_{14} : AS believes $\varphi(AS)$

A_{15} : AS believes PS controls PS $\xleftarrow{K_{AS,PS}}$ AS

Access Point

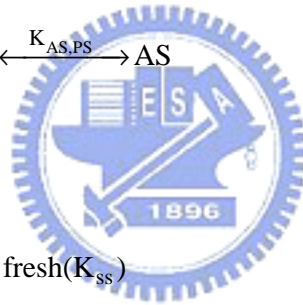
A_{16} : AP has (AP,AS)

A_{17} : AP has $K_{AS,AP}$

A_{18} : AP believes fresh($K_{AS,AP}$)

A_{19} : AP believes $\varphi(AP)$

A_{20} : AP believes AS controls AS $\xleftarrow{K_{AS,AP}}$ AP



Password server

- A_{21} : PS has $(U, AS, PS) \wedge PS \xleftarrow{\text{Seed, PIN}} U$
 A_{22} : PS has $K_{AS,PS}$
 A_{23} : PS believes fresh($K_{AS,PS}$)
 A_{24} : PS believes $PS \xleftarrow{K_{AS,PS}} AS$
 A_{25} : PS has N_s
 A_{26} : PS believes fresh(N_s)
 A_{27} : PS believes fresh($\varnothing(PS)$)
 A_{28} : PS believes AS controls $AS \xleftarrow{K_{AS,PS}} AP$

Received Message assumptions

Detailed assumptions about messages each party receives the following...

- Message (i) : $U \rightarrow AP$: $U, PS \{ H(PIN, T), H(U, T), N_u \}_T$
Message (ii) : $AP \rightarrow AS$: $U, PS \{ H(PIN, T), H(U, T), N_u \}_T$
Message (iii) : $AS \rightarrow PS$: $\{ H(PIN, T), H(U, T), N_u \}_T$
Message (iv) : $PS \rightarrow AS$: $\{ H(U, T), T, N_s \}_{K_{AS,PS}}$
Message (v) : $AS \rightarrow AP$: $\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \}_{K_{AS,AP}}$
Message (vi) : $AP \rightarrow U$: $\{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$

After the protocol completes faithfully each party have received the following....

- A_{29} : AP received $U, PS \{ H(PIN, T), H(U, T), N_u \}_T$
 A_{30} : AS received $U, PS \{ H(PIN, T), H(U, T), N_u \}_T$
 A_{31} : PS received $\{ H(PIN, T), H(U, T), N_u \}_T$
 A_{32} : AS received $\{ H(U, T), T, N_s \}_{K_{AS,PS}}$
 A_{33} : AP received $\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \}_{K_{AS,AP}}$
 A_{34} : U received $\{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$

Comprehension assumptions

Detailed assumptions about each party's comprehension of received messages based on $A_{29} - A_{34}$

A_{35} : AP believes AP received $\langle U \rangle_{*AP}, PS, \langle \{ H(PIN, T), H(U, T), N_u \}_T \rangle_{*AP}$

A_{36} : AS believes AS received $U, PS, \langle \{ H(PIN, T), H(U, T), N_u \}_T \rangle_{*AS}$

A_{37} : PS believes PS received $\langle \{ H(PIN, T), H(U, T), N_u \}_T \rangle$

A_{38} : AS believes AS received $\{ \{ H(U, T), T, N_s \}_{K_{AS, PS}} \}$

A_{39} : AP believes AP received $\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \}_{K_{AS, AP}}$

A_{40} : U received $\{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$

Interpretation assumptions:

Detailed assumptions about how each party interprets after receiving the messages

A_{41} : U believes AP received $U, PS \{ H(PIN, T), H(U, T), N_u \}_T$

A_{42} : AP believes AS received $\langle U \rangle_{*AP}, \langle PS \rangle_{*AP}, \langle \{ H(PIN, T), H(U, T), N_u \}_T \rangle_{*AP}$

A_{43} : AS believes PS received $\langle \{ H(PIN, T), H(U, T), N_u \}_T \rangle_{*AS}$

A_{44} : PS believes AS received $\{ H(U, T), T, N_s \}_{K_{AS, PS}}$

A_{45} : AS believes AP received $\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \}_{K_{AS, AP}}$

A_{46} : AP believes U received $\{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$

Derivation w,r,t Password server

Here we analyze how messages sent to each component satisfy our authentication goals.

A_3 : U believes $U \xleftarrow{\text{Seed, PIN}} PS$

A_{47} : PS believes AS says $\langle \{ H(PIN, T), H(U, T), N_u \}_T \rangle_{*AS}$

When PS receives this message it does the token number verification. If the token numbers are verified successfully, the user is confirmed for his reality, PS has the following beliefs....

A_{48} : PS believes $PS \xleftarrow{K_{AS, PS}} AS$

A_{49} : PS believes PS has $K_{AS, PS}$

A_{50} : PS believes fresh ($K_{AS, PS}$)

A_{51} : PS believes AS has $K_{AS, PS}$

Based on these beliefs the following goals w.r.to PS for AS on behalf of User is mentioned

below :

Goals G1 and G2 are analyzed thru A_{47} and A_{50}

Goal G3 is analyzed thru A_{48}, A_{49} and A_{51}

Goal G4 is analyzed thru A_{50}

Goal G5 is analyzed thru A_{48} and A_{51}

Goal G6 is analyzed thru A_{48}

From these goals PS confirms the AS, that the user is verified and authenticated .Also confirms that, the user actually participates the session (**Prevent impersonation**), confirms freshness of messages (**Detect replay attacks**).

Derivation w,r,t Authentication server

The AS gets the confirmation user message from the PS, AS has the following beliefs,

A_{52} : AS believes PS says $\{ H(U,T), T, N_s \}_{K_{AS,PS}}$

A_{53} : AS believes $PS \xleftarrow{K_{AS,PS}} AS$

A_{54} : AS believes PS has $K_{AS,PS}$

A_{55} : AS believes fresh ($K_{AS,PS}$)

A_{56} : AS believes AS has $K_{AS,PS}$

A_{57} : AS believes PS says fresh(N_u)



Based on these beliefs the following goals w.r.to AS for AP on behalf of user is mentioned

below

Goals G1 and G2 are analyzed thru A_{52} and A_{55}

Goal G3 is analyzed thru and A_{53}, A_{54} and A_{56}

Goal G4 is analyzed thru A_{57}

Goal G5 is analyzed thru A_{53} and A_{56}

Goal G6 is analyzed thru A_{53}

These goals prove that, AS confirms the AP that the user is verified and authenticated. Also confirms that, the user actually participates the session (**Prevent impersonation**) confirms freshness of messages (**Detect replay attacks**). AS confirms the User and sends a message to the

AP by encrypting it with the Pre shared key $K_{AS,AP}$.

Derivation w,r,t Access Point

The AP gets the user confirmation message from the AS and it has the following beliefs,

A_{58} : AP believes AS says $\{ \{ U, Sid, H(K_{SS}, N_u, N_s) \}, \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T \} K_{AS,AP}$

A_{59} : AP believes $AS \xleftarrow{K_{AS,AP}} AP$

A_{60} : AP believes AS has $K_{AS,AP}$

A_{61} : AP believes AP has $K_{AS,AP}$

A_{62} : AP believes fresh ($K_{AS,AP}$)

A_{63} : AP believes fresh (K_{SS})

A_{64} : AP believes AS says fresh (N_u)

A_{65} : AP believes AS says fresh (N_s)

Based on these beliefs the following goals w.r.to AP on behalf of user is analyzed below

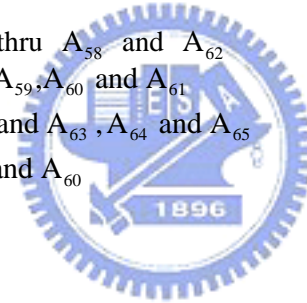
Goals G1 and G2 are analyzed thru A_{58} and A_{62}

Goal G3 is analyzed thru A_{59}, A_{60} and A_{61}

Goal G4 is analyzed thru A_{62} and A_{63}, A_{64} and A_{65}

Goal G5 is analyzed thru A_{59} and A_{60}

Goal G6 is analyzed thru A_{59}



From these goals it is meant that, AP identifies that the user is verified and authenticated .by the AS, hence AP finally confirms that the user actually participates the session, **(Prevent impersonation)** confirms freshness of messages **(Detect replay attacks)**

The AP sends the authentication granted message to the user, which means that the user is authenticated and authorized to take part in the current session with the system.

Derivation w,r,t User

The user gets the authentication granted message from the AP.

$AP \rightarrow U : \{ U, Sid, AS, H(K_{SS}, N_u, N_s), TS \}_T$

User uses his token number to decrypt the message and confirms that he has been

authenticated by the real PS

$$A_{66}: U \text{ believes PS has } (U, AS, PS) \wedge PS \xleftrightarrow{\text{Seed, PIN}} U$$

User already has the following belief:

$$\text{From, } A_3: U \text{ believes } U \xleftrightarrow{\text{Seed, PIN}} PS$$

So from these statements, it is confirmed that, both the user and the PS mutually trust each other and mutually authenticate each other.

Subsequent authentication phase :

$$U \rightarrow AP: \{H(U, Sid, N_u)\}_{K_{ss}}$$

$$AP \rightarrow U: \{H(U, N_u)\}_{K_{ss}}$$

Initial state assumptions

User

$$A_{67}: U \text{ has } \{U, AP, Sid, N_u, TS\}$$

$$A_{68}: U \text{ believes } U \xleftrightarrow{K_{ss}} AP$$

$$A_{69}: U \text{ has } K_{ss}$$

$$A_{70}: U \text{ believes fresh}(K_{ss})$$

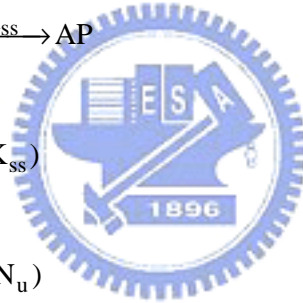
$$A_{71}: U \text{ has } (N_u)$$

$$A_{72}: U \text{ believes fresh } (N_u)$$

$$A_{73}: U \text{ believes } \varphi(U)$$

$$A_{74}: U \text{ believes } \varphi(N_u)$$

$$A_{75}: U \text{ believes AP controls } AP \xleftrightarrow{K_{ss}} U$$



^z
Access Point

- A_{76} : AP has { U, AP, Sid, K_{ss} }
- A_{77} : AP believes $AP \xleftarrow{K_{ss}} U$
- A_{78} : AP has K_{ss}
- A_{79} : AP believes fresh (K_{ss})
- A_{80} : AP believes $\varphi(AP)$
- A_{81} : AP believes $\varphi(N_u)$
- A_{82} : AP believes U says fresh (N_u)
- A_{83} : AP believes $AP \xleftarrow{K_{ss}} U$

Received Message assumptions

1. $U \rightarrow AP : \{H(U, Sid, N_u)\}_{K_{ss}}$
2. $AP \rightarrow U : \{H(U, N_u)\}_{K_{ss}}$

After the protocol completes faithfully each party have received the following

- A_{84} : AP received $H(U, Sid, N_u)_{K_{ss}}$
- A_{85} : U received $\{H(U, N_u)\}_{K_{ss}}$

Comprehension assumptions

- A_{86} : AP believes AP received $H(U, Sid, N_u)_{K_{ss}}$
- A_{87} : U believes U received $\{H(U, N_u)\}_{K_{ss}}$

Interpretation assumptions

- A_{88} : U believes AP received $H(U, Sid, N_u)_{K_{ss}}$
- A_{89} : AP believes U received $\{H(U, N_u)\}_{K_{ss}}$
- A_{90} : U believes AP has K_{ss}

Derivation w,r,t Access Point

User requests the AP for subsequent authentication by giving message encrypted using its session key K_{ss}

- A_{91} : AP believes U says $H(U, Sid, N_u)_{K_{ss}}$
- A_{92} : AP has { U, AP, Sid, K_{ss} }
- A_{93} : AP believes $AP \xleftarrow{K_{ss}} U$
- A_{94} : AP has K_{ss}
- A_{95} : AP believes fresh (K_{ss})
- A_{96} : AP believes U says fresh (N_u)
- A_{97} : AP believes AP controls $AP \xleftarrow{K_{ss}} U$

Based on these beliefs the following goals w.r.to AP on behalf of user is analyzed below :

Goals G1 and G2 are analyzed thru A_{91} and A_{95}

Goal G3 is analyzed thru and A_{93}, A_{94} and A_{97}

Goal G4 is analyzed thru A_{95}

Goal G5 is analyzed thru A_{93} and A_{94}

Goal G6 is analyzed thru A_{93}

From these goals it is clear that, the AP identifies that the User is the real user using the session key K_{ss} . So, the AP confirms the User to be a participant of the session, (**Prevent impersonation**) and confirm freshness of messages (**Detect replay attacks**).

Derivation w,r,to User

When the AP receives the user request message for subsequent authentication it first identifies the User using the session key K_{ss} , decrypts the message using its own session key.

When the user is verified the AP sends the authentication granted message to the user, which means that the user is subsequently authenticated to the current session with the system.

Now the user is subsequently authenticated, by the AP.

A_{98} : U believes AP says $H(U, N_u) \}_{K_{ss}}$

A_{99} : U has $\{U, AP, Sid, N_u, TS\}$

A_{100} : U believes $U \xleftarrow{K_{ss}} AP$

A_{101} : U has K_{ss}

A_{102} : U believes fresh(K_{ss})

A_{103} : U has (N_u)

A_{104} : U believes fresh (N_u)

A_{105} : U believes AP controls $AP \xleftarrow{K_{ss}} U$

Based on these beliefs the following goals w.r.to User on behalf of AP is analyzed below

Goals G1 and G2 are analyzed thru A_{98} and A_{102}

Goal G3 is analyzed thru and A_{100}, A_{101} and A_{105}

Goal G4 is analyzed thru A_{102}

Goal G5 is analyzed thru A_{100} and A_{101}

Goal G6 is analyzed thru A_{100} and A_{105}

From these goals it is meant that, the user is subsequently authenticated by the AP and the AP is real to authenticate the user. So the user and the AP could mutually trust each other by the means of the session key K_{ss} and it is confirmed thru the above mentioned goals.

