



SECURITY MODEL IN INTERNET RELAY CHAT

MS. THARNKAMOL PENPOPPICHANAN

Submitted in Partial Fulfillment of the
Requirements for the Degree of

Master of Science

In Information Technology
Assumption University

December, 2001

Security Model in Internet Relay Chat

By

Ms. Tharnkamol Penpoppichanan



**Submitted in Partial Fulfillment of
The Requirements for the Degree of
Master of Science
In Information Technology
Assumption University**

December, 2001

The Faculty of Science and Technology


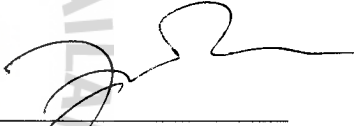
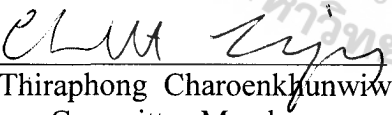

Thesis Approval

Thesis Title Security in Internet Relay Chat

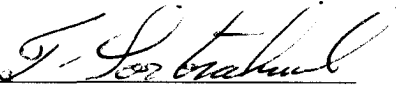

By Ms. Tharnkamol Penpoppichanan
Thesis Advisor Asst.Prof.Dr. Thotsapon Sortrakul
Academic Year 1/2001

The Department of Information Technology, Faculty of Science and Technology of Assumption University has approved this final report of the **twelve** credits course. **IT7000 Master Thesis**, submitted in partial fulfillment of the requirements for the degree of Master of Science in Information Technology.

Approval Committee:


(Asst.Prof.Dr. Thotsapon Sortrakul)
Advisor
(Dr. Jirapun Daengdej)
Committee Member
(Dr. Thiraphong Charoenkunwiwat)
Committee Member
(Asst.Prof.Dr. Surapong Auwatanamongkol)
Representative of Ministry of
University Affairs

Faculty Approval:


(Asst.Prof.Dr. Thotsapon Sortrakul)
Director
(Asst.Prof.Dr. Pratit Santiprabhob)
Dean

ACKNOWLEDGEMENTS

I would like to express my most sincere appreciation and special thanks to Dr. Thotsapon Sortrakul, my advisor, for the good advice, valuable suggestions, and guidance through this thesis's period.

I would like to convey my special thanks my thesis's committee members, Dr. Thiraphong Charoenkhunwiwat and Dr. Jirapun Daengdej, for their valuable comments and suggestions.

Finally, I would like to express special thanks to my mom, dad, K.Kritsada Ruayruengrung, and all friends who encouraged and supported me throughout my thesis.

Ms. Tharnkamol Penpoppichanan

December, 2001



ABSTRACT

This thesis emphasizes on solving the security problems in IRC, by proposing the alternative security model on IRC that is implemented by four main techniques:

- Strong authentication based on cryptography (RSA algorithm 2048 bits key)
- Strong encryption (Triple DES algorithm 168 bits key)
- Use random function
- Use one way hashing function (SHA-1 algorithm 128 bits key)

After implementing these techniques, IRC will have good authentication process to verify people before establishing session between client-server, have encryption process to encrypt message before sending through the Internet, and have preventing playback messages.

The experiment shows it does not affect the performance of the chat system, so an alternative model has advantages for private communication, and for applying for use in business organization which require chatting via the public network.

TABLE OF CONTENTS

	Page
ABSTRACT	i
ACKNOWLEDGEMENT	ii
LIST OF FIGURES	vi
LIST OF TABLES	viii
CHAPTER 1 INTRODUCTION	1
1.1 Background of thesis	1
1.2 Objective and scope of thesis	3
CHAPTER 2 IRC AND SECURITY	5
2.1 What is IRC	5
2.2 IRC Network	6
2.3 IRC Components	6
2.3.1 Servers	6
2.3.2 Clients	7
2.3.3 Channels	7
2.4 Type of communication	8
2.4.1 One to one communication	9
2.4.2 One to many communication	9
2.5 How IRC works	12
2.6 Data Flow Control Characteristic of IRC Communication	14
2.6.1 Network protocol	14
2.6.2 Message Delivery	14

2.6.3 Connection “Liveness”	15
2.6.4 Establishing a server to client connection	15
2.6.5 Establishing a server to server connection	15
2.6.6 Server exchange information when connecting	16
2.6.7 Terminating server-client connection	17
2.6.8 Terminating server-server connection	17
2.7 Goals of IRC	17
2.8 Good point of IRC	17
2.9 Limitation of IRC	18
2.10 Security Terminology	19
2.11 Security goals	20
2.12 Threats to the security of computing system	20
2.13 Encryption	20
2.13.1 Encryption Components	21
2.13.2 Security and Encryption	22
2.14 Encryption Techniques	22
2.14.1 Symmetric Encryption	22
2.14.2 Asymmetric Encryption	23
2.15 DES cryptosystem	24
2.16 Triple DES cryptosystem	26
2.17 RSA cryptosystem	29
2.18 One-way Encryption (One-way Hash Function)	30
2.19 SHA and SHA-1	31

CHAPTER 3	BUSINESS IRC MODEL (bIRC)	32
3.1	Overview of bIRC model	33
3.2	Pre-define at Server's site	33
3.3	Pre-define at Client's site	35
3.4	Client-Server Authentication process of the model	36
3.4.1	The first process "Client verify Server's Public key that is buried in client's Database valid or not"	36
3.4.2	The second process "Client identify itself to the server by using User ID., password"	37
3.4.3	The third process "Client verify server authorized or not by checking random data (C1)"	38
3.4.4	The fourth process "Server verify client authorized or not by checking random data (C2)"	39
3.5	Communication in channel	44
3.6	Sample screen for communication in channel of the model	46
CHAPTER 4	COMPARISON, ANALYSIS AND EVALUATION	49
4.1	Security Comparison and Analysis	49
4.2	Experimental and Evaluation	53
CHAPTER 5	CONCLUSION AND RECOMMENDATION	56
5.1	Conclusions	56
5.2	Recommendation	57
CHAPTER 6	BIBLIOGRAPHY	58

LIST OF FIGURES

Figure 2-1 : IRC network	6
Figure 2-2 : One-to-one communication	9
Figure 2-3 : One-to-many communication (To a list)	9
Figure 2-4 : One-to-many communication (To a group/channel)	10
Figure 2-5 : One-to-many communication (To a host/server mask)	11
Figure 2-6 : One-to-many communication (To all)	11
Figure 2-7 : Client establish connection to server	12
Figure 2-8 : Symmetric Encryption	23
Figure 2-9 : Asymmetric Encryption	24
Figure 2-10 : DES Encryption Algorithm	25
Figure 2-11 : Triple DES Encryption Algorithm	28
Figure 3-1 : Step of pre-defining model at server's site	34
Figure 3-2 : Step of pre-defining model at client's site	35
Figure 3-3 : (First process) Client verify server's public key valid or not	36
Figure 3-4 : Client's encryption and Server's decryption process	37
Figure 3-5 : Check user ID. and digested password	37
Figure 3-6 : Server reject, and terminate connection	38
Figure 3-7 : Server's encryption and Client's decryption process	38
Figure 3-8 : Compare random data (C1) for authenticate server	39
Figure 3-9 : Compare random data (C2) for authenticate client	40

Figure 3-10 : All authentication process when client request to establish session to IRC server	41
Figure 3-11 : Requesting to join in channel process	45
Figure 3-12 : Flow of message in the channel	46
Figure 3-13 : Example screen for chatting	46
Figure 3-14 : Example screen when client join to Host for chatting	47
Figure 3-15 : Example screen when client are chatting in channel	47
Figure 3-16 : Establish connection (Authentication and communication process)	48
Figure 4-1 : Security level classified by security techniques	50



LIST OF TABLES

Table 4-1 : Security feature of other well known chat versus bIRC model	49
Table 4-2 : Experimental result conclusion table	54



CHAPTER 1

INTRODUCTION

1.1 Background of the thesis

In the past, the familiar communication system was the local public telephone system and long distance telephone system, which enabled you to contact with everyone in every part of the world, but this communication technology is rather expensive. With high technology in chip produced at lower price, computer prices became lower. This composed with networking technology becoming more powerful and growing very rapidly resulted in the new technology called the Internet.

The Internet is a collection of computers connected to each other by using network equipment. They make a public network where everyone can contact each other. In recent years, the Internet has experienced a tremendous growth in number of users and popularity, so nowadays the Internet is more and more becoming a necessity. People use Internet as an everyday-communication tool among their business partners, family and friends everywhere in the world. The Internet is lower priced than other communication systems, uses TCP/IP protocol to communicate with each other, and it uses TCP/IP protocol, it provides many communication services such as E-mail, FTP, Chat, Instant Messaging (IM), etc.

The most common way of communication on the Internet is E-mail, and although it can approximate the feel of conversation at times, they lack the true spontaneity of live interaction. E-mail is a messaging system that is not a real-time communication tool on the Internet, but Chat is a real-time interaction communication tool on the Internet. So chat has become a very popular communication tool on the Internet, because its cheap, quick, and real-time.

There are many chat programs on the Internet, such as SNC (Secure Network Chat), Zephyr, AnotherNet Chat, mIRC, Pirch, PharoWeb Chat, etc.

Most of them are user-friendly and real-time Internet tools that allow users to find other people, add them to a contact list, see their online/offline status at any given time, and communicate with them in several different ways and they depend on the same protocol, Internet Relay Chat (IRC).

IRC has been developed since 1989, and it was first formally documented in May 1993 by RFC 1459. This protocol has kept evolving, updated by RFC 2810 (describing the architecture of current IRC protocol), RFC 2811 (specifies how channels, their characteristics and properties that managed by IRC servers), RFC 2812 (defines the client protocol), and RFC 2813 (develops IRC server protocol for better scalability).

IRC is an example of the communication systems in which both one-to-one and one-to-many communication or as we study in the class mono-casting and broadcasting models are implemented. A typical setup involves a single process (the server) forming a central point for clients (or other servers) to connect to, performing the required message delivery/multiplexing and other functions.

IRC is networked over much of North America, Europe, and Asia. When you are talking in IRC, everything you type can instantly be transmitted around the world to other users who are connected at the time. They can then type something and respond to your messages.

Because the Internet is an open system, so it seems too hard to identify people, companies and computers communicating on the Internet are valid or not. Furthermore, the very communication path is inherently insecure - all communications are potentially open for an eavesdropper to read and modify as they

pass between the communicating endpoints, if an attacker is in the right place at the right time, he can:

- Read your message, thus snooping on your conversation.
- Modify your message, thus subverting your conversation.
- Send message to you or your correspondent, thus impersonating either party.

The conventional IRC does not have a good security issue to eliminate its limitation, and it can be concluded that IRC is not suitable for private or secret personal conversation, and it cannot be applied for use within the business organization that requires secure conference via the public network (Internet).

As mentioned above, a strong authentication method and encryption method based on cryptography are required.

1.2 Objective and scope of thesis

Today, in the Information Technology age, information security has become a major concern for everyone around the world. Businesses and people appreciate the importance of their communication privacy and consider information as a valuable asset.

So this thesis focuses on weak points of IRC which is lacking good security model to provide confidentiality and integrity of messages to the users on the public network.

This thesis aims to design the extended security model for IRC protocol to eliminate limitation of IRC, making secure conferencing in the Internet.

The extended security model is designed by

1.) Adding the authentication process to verify a legitimate person before establishing connection between client and server, by using Asymmetric key or public key that is generated by RSA algorithm (2048 bits key).

2.) Adding an encryption process to encrypt message before sending a message travelling through network with a symmetric key or secret key that is generated by Triple DES algorithm (168 bits key).

3.) Using random function to create random number to authenticate each other between client and server, and prevent playback message.

4.) Using a one-way hashing function (SHA-1 algorithm 128 bits key) to create message digest for message integrity (server's public key integrity).

By implement authentication and encryption, the IRC will provide confidentiality and integrity for data. So an unauthorized party cannot intercept, modify, or fabricate data in the network, furthermore, it can be developed for use in business as well, using existing networks to make secure chat networks for their business with low capital.

CHAPTER 2

IRC AND SECURITY

2.1 What is IRC

IRC, Internet Relay Chat was originally designed by Jarkko Oikarinen in Finland in 1988. It is a real-time conversational system that is similar to the talk command which is available on many machines in the Internet. The IRC protocol has been designed over a number of years for use with text based conferencing. It has been developed on systems using the TCP/IP network protocol, although there is no requirement that this remain the only sphere in which it operates. Through the use of the client-server model, IRC is well-suited to running on many machines in a distributed fashion. Some of the more popular chat clients are mIRC and Pirc for Windows.

IRC is an example of the communication systems in which both one-to-one and one-to-many communication or as we study in the class mono-casting and broadcasting models are implemented. A typical setup involves a single process (the server) forming a central point for clients (or other servers) to connect to, performing the required message delivery/multiplexing and other functions.

IRC is networked over much of North America, Europe, and Asia. When you are talking in IRC, everything you type can instantly be transmitted around the world to other users who are connected at the time. They can then type something and respond to your messages.

Topics of discussion on IRC are varied. Technical and political discussions are popular, especially when world events are in progress. IRC is also a way to expand your horizons, as people from many countries and cultures are on, 24 hours a

day. Most conversations are in English, but there are always channels in German, Japanese, Finnish, Russian and occasionally other languages.

2.2 IRC Network

IRC is impressive enough on a standalone server, but its magic lies in the fact that IRC servers can be linked together into IRC networks, with each public channel accessible to any user connected to any server on that network.

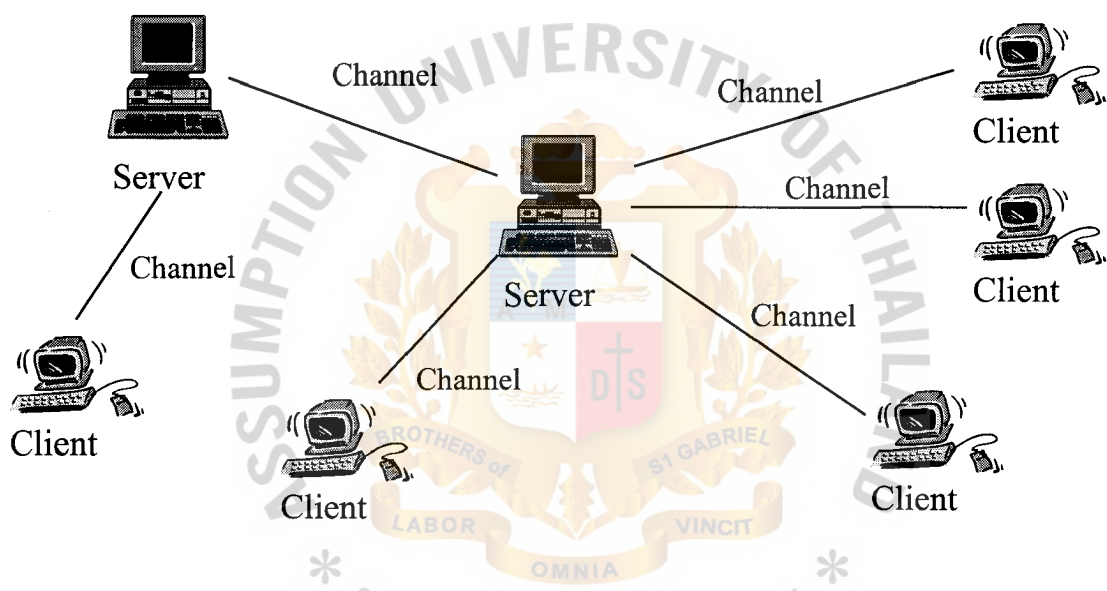


Figure 2-1 : IRC network

2.3 IRC Components

2.3.1. Servers

The server forms the backbone of IRC, providing a point to which clients may connect for talking to each other, and a point for other servers to connect to, forming an IRC network. The only network configuration allowed for IRC servers is that of a spanning tree where each server acts as a central node for the rest of the net it sees.

2.3.2 Clients

A client is anything connecting to a server that is not another server. Each client is distinguished from other clients by a unique nickname having a maximum length of nine (9) characters. Usually users choose a constant nickname, as our experience has taught us, although generally each time connecting to a server one can use a new nickname. We connect to the server in the following way:

irc Nick /SERVER irc.tau.ac.il

this means that a client who chooses the nickname Nick connects to the IRC-server of Tel-Aviv University.

In addition to the nickname, all servers must have the following information about all clients:

- * The real name of the host that the client is running on.
- * The username of the client on that host.
- * The server to which the client is connected.

2.3.3. Channels

A channel is a named for group of one or more clients who will all receive messages addressed to that channel. The channel is created implicitly when the first client joins it, and the channel ceases to exist when the last client leaves it. While channel exists, any client can refer to the channel using the name of the channel.

To create a new channel or become part of an existing channel, a user is required to JOIN the channel. If the channel doesn't exist prior to joining, the channel is created. As part of the protocol, a user may be a part of several channels at once, but a limit of ten channels is recommended.

The JOIN command is used by client to start listening to a specific channel. Whether or not a client is allowed to join a channel is checked only by the server the client is connected to; all other servers automatically add the user to the channel when it is received from other servers.

Before JOINing a channel a user might be interested to know WHO is already listening to this channel. Then he (or she) uses the WHO command for example like this:

/who #israel

The printed information will include the nickname, the real name of the host, the username, the server of each user connected to this channel

Once a user has joined a channel, they receive notice about all commands their server receives which affect the channel. The JOIN command needs to be broadcast to all servers so that each server knows where to find the users who are on the channel.

If a JOIN is successful, the user is then sent the channel's topic and the list of users who are on the channel .

2.4 Types of communication

IRC is an example of the communication systems in which both one-to-one and one-to-many communication or as we study in the class mono-casting and broadcasting models are implemented. A typical setup involves a single process (the server) forming a central point for clients (or other servers) to connect to, performing the required message delivery/multiplexing and other functions.

2.4.1 One-to-one communication.

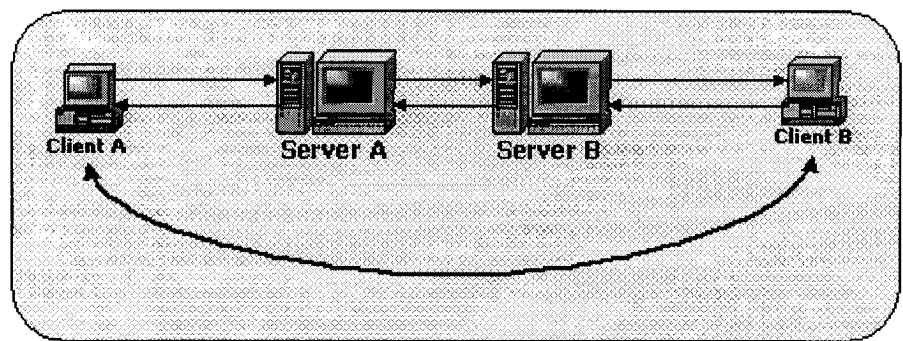


Figure 2-2 : One-to-one communication

Communication on a one-to-one basis is usually only performed by clients, since most server-server traffic is not the result of servers talking only to each other. To provide a secure means for clients to talk to each other, it is required that all servers be able to send a message in exactly one direction along the spanning tree in order to reach any client. The path of a message being delivered is the shortest path between any two points on the spanning tree.

2.4.2 One-to-many communication.

The main goal of IRC is to provide a forum which allows easy and efficient conferencing (one to many conversations). IRC offers several means to achieve this, each serving its own purpose.

- **To a list.**

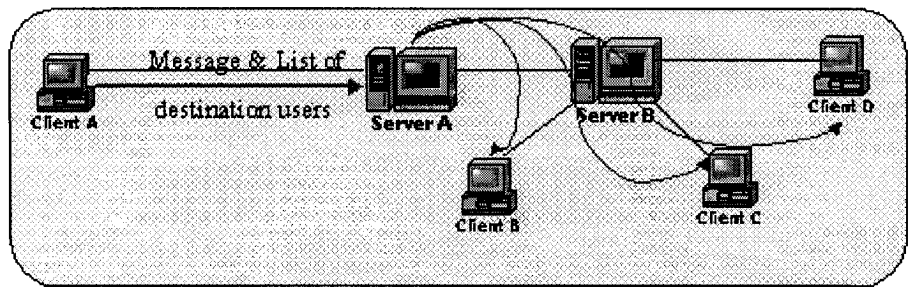


Figure 2-3 : One-to-many communication (To a list)

The least efficient style of one-to-many conversation is through clients talking to a 'list' of users. This is how it's done:

The client gives a list of destinations to which the message is to be delivered and the server breaks it up and dispatches a separate copy of the message to each given destination. This isn't as efficient as using a group since the destination list is broken up and the dispatch sent without checking to make sure duplicates aren't sent down each path.

- **To a group (channel)**

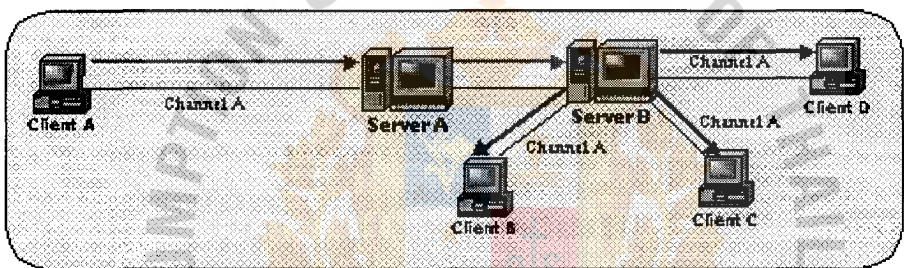


Figure 2-4 : One-to-many communication (To a group/channel)

In IRC, the channel has a role equivalent to that of the multicast group; their existence is dynamic (coming and going as people join and leave channels) and the actual conversation carried out on a channel is only sent to servers which are supporting users on a given channel. If there are multiple users on a server in the same channel, the message text is sent only once to that server and then sent to each client on the channel. This action is then repeated for each client-server combination until the original message has fanned out and reached each member of the channel.

- One to a host/server mask.

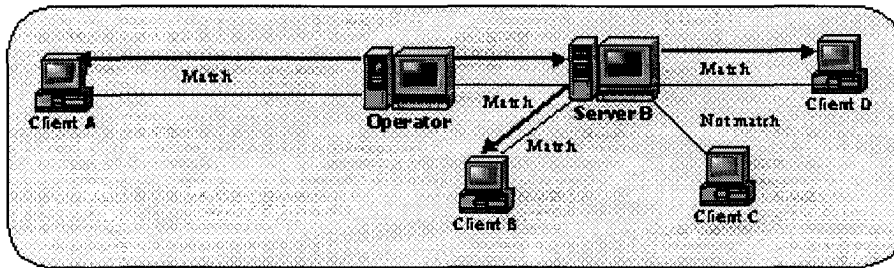


Figure 2-5 : One-to-many communication (To a host/server mask)

To provide IRC operators with some mechanism to send messages to a large body of related users, host and server mask messages are provided. These messages are sent to users whose host or server information match that of the mask. The messages are only sent to locations where users are, in a fashion similar to that of channels.

- One-to-all.

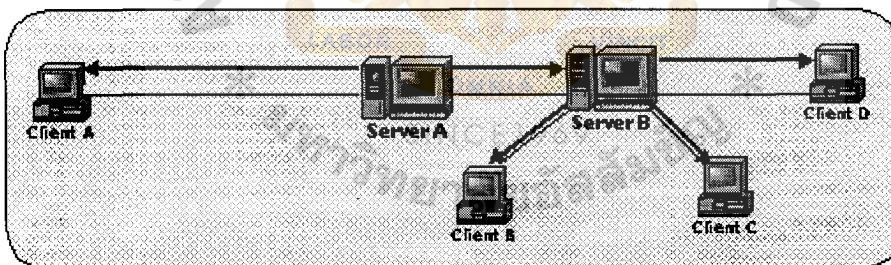


Figure 2-6 : One-to-many communication (To all)

The one-to-all type of message is better described as a broadcast message, sent to all clients or servers or both. On a large network of users and servers, a single message can result in a lot of traffic being sent over the network in an effort to reach all of the desired destinations. For some messages, there is no option but to broadcast

it to all servers so that the state information held by each server is reasonably consistent between servers.

2.5 How IRC works

IRC users meet in channels. The "official" definition of a channel, according to the technical description of the IRC protocol in RFC 1459, is "a named group of one or more clients which will all receive messages addressed to that channel." The definition says nothing about the content of the shared messages, but by convention, messages on a given channel are confined to one topic area.

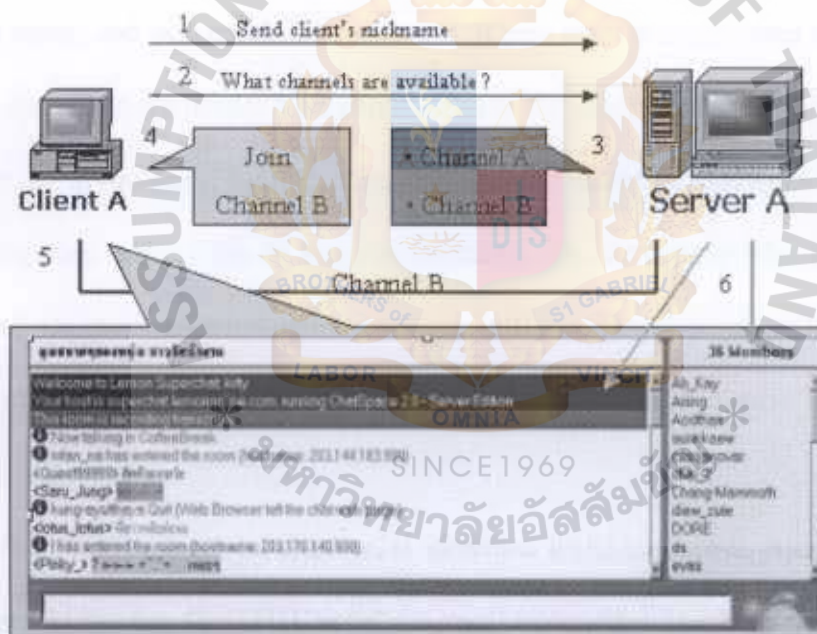


Figure 2-7 : Client establish connection to server

When you connect your IRC client to a server, you can see what channels are available by typing `/LIST`. Once you find one in which you'd like to participate--let's say the `#NEWBIES` channel for this example (by convention, channel names begin with the pound sign)--you simply type `/JOIN #NEWBIES`.

Windows-based client software will, at this point, open a new window, showing who is currently on the #NEWBIES channel. You can then begin communicating. If the #NEWBIES channel does not currently exist, the /JOIN #NEWBIES command will create it, and it will be accessible to anyone using the /LIST command (although the IRC protocol does allow new channels to be set as private, secret, or invite-only).

If you create a channel, you're given the status of channel operator, or op or chop. Your nickname is prefaced by the @ character (for instance, @Xena), and as channel owner, you have special powers over that channel. If it's an invite-only channel, you can invite users to join you. You can also change the description of the channel's topic, and you can kick a user out. If you want a customized channel, you can alter the channel's mode: You can moderate the channel; specify that it's secret, private, or by invite-only; limit the number of users; establish password-only access; and put together a ban filter to keep undesirables out. Obviously, this is significant power, and too many channel operators abuse it. But with thousands of IRC channels from which to choose, the easiest way to deal with power-mad channel ops is simply to go somewhere else.

When you connect to a server, it identifies itself and sends your IRC clients the message of the day (MOTD), usually a small welcoming message supplied by the server administrator. It then gives you a count of the number of users currently connected to the server, as well as the number of IRC servers on the network visible to that server. At this point, the server sends a message to all servers on the network stating your nickname and the fact that you've connected. When you send a message, the server decides what to do with it. If you're the only person in a channel, the message goes to the server and nowhere else. If you send a message to a specific

user, it will go to the server and the server will route it to the user either directly or through an IRC network. If the message is a general one for the entire channel, the server sends it directly to clients connected to that server, and indirectly to everyone else currently in that channel by relaying it through other servers.

2.6 Data Flow Control Characteristic of IRC communication

2.6.1 Network protocol: TCP

IRC has been implemented on top of TCP since TCP supplies a reliable network protocol which is well suited to this scale of conferencing. The use of multicast IP is an alternative, but it is not widely available or supported at the present time.

2.6.2 Message delivery

It is common to find network links saturated or hosts to which you are sending data unable to send data. Although Unix typically handles this through the TCP window and internal buffers, the server often has large amounts of data to send (especially when a new server-server link forms) and the small buffers provided in the kernel are not enough for the outgoing queue. To alleviate this problem, a "send queue" is used as a FIFO queue for data to be sent. A typical "send queue" may grow to 200 Kbytes on a large IRC network with a slow network connection when a new server connects.

When polling its connections, a server will first read and parse all incoming data, queuing any data to be sent out. When all available input is processed, the queued data is sent. This reduces the number of write() system calls and helps TCP make bigger packets.

2.6.3 Connection 'Liveness'

To detect when a connection has died or become unresponsive, the server must ping each of its connections that it doesn't get a response from in a given amount of time.

If a connection doesn't respond in time, its connection is closed using the appropriate procedures. A connection is also dropped if its sendq grows beyond the maximum allowed, because it is better to close a slow connection than have a server process block.

2.6.4 Establishing a server to client connection

Upon connecting to an IRC server, a client is sent the MOTD (if present) as well as the current user/server count (as per the LUSER command). The server is also required to give an unambiguous message to the client which states its name and version as well as any other introductory messages which may be deemed appropriate.

After dealing with this, the server must then send out the new user's nickname and other information as supplied by itself (USER command) and as the server could discover (from DNS/authentication servers). The server must send this information out with NICK first followed by USER.

2.6.5 Establishing a server-server connection

The process of establishing of a server-to-server connection is fraught with danger since there are many possible areas where problems can occur - the least of which are race conditions.

After a server has received a connection following by a PASS/SERVER pair which were recognized as being valid, the server should then reply with its own PASS/SERVER information for that connection as well as all of the other state information it knows about as described below.

When the initiating server receives a PASS/SERVER pair, it too then checks that the server responding is authenticated properly before accepting the connection to be that server.

2.6.6 Server exchange of state information when connecting

The order of state information being exchanged between servers is essential.

The required order is as follows:

- all known other servers;
- all known user information;
- all known channel information.

Information regarding servers is sent via extra SERVER messages. By passing the state information about servers first, any collisions with servers that already exist occur before nickname collisions due to a second server introducing a particular nickname. Due to the IRC network only being able to exist as an acyclic graph, it may be possible that the network has already reconnected in another location, the place where the collision occurs indicating where the net needs to split.

2.6.7 Terminating server-client connections

When a client connection closes, a QUIT message is generated on behalf of the client by the server to which the client connected. No other message is to be generated or used.

2.6.8 Terminating server-server connections

If a server-server connection is closed, either via a remotely generated SQUIT or 'natural' causes, the rest of the connected IRC network must have its information updated with by the server which detected the closure. The server then sends a list of SQUITs (one for each server behind that connection) and a list of QUITs (again, one for each client behind that connection).

2.7 Goals of IRC

- The main goal of IRC is to provide a forum which allows easy and efficient conferencing.
- In addition, IRC provide real-time communication at a low price, where people can communicate everywhere and every time they want.

2.8 Good point of IRC

- IRC is a teleconferencing system, which using client-server model that is well suited to running on many machines in a distributed fashion.
- IRC is a text-based conferencing that consume a little bandwidth, so it usually does not have any problem about networking.
- IRC offers the most immediate way to communicate over the Internet.

- IRC offers no restriction to the number of people who can participate in a given discussion.
- IRC server uses spanning tree algorithm to eliminate loops in the network.
- The path of a message being delivered is the shortest path between any two points on the spanning tree.
- IRC server can detect and close the connection, when the connection has died or become unresponsive or exceed maximum time allowed.
- IRC implement on top of TCP, since TCP supplies a reliable network protocol which is well suited to this scale of conferencing.
- IRC has a large buffer to keep “send queue”, typical “send queue” which may grow to 200 Kbytes.

2.9 Limitation of IRC

- IRC has ineffective authentication to control access, because one of the main way it uses to control access to channel is to use username and hostname of the user connections.
- Most IRC networks (especially public networks) do not have anything like authentication to provide little guarantee about the accuracy of the username and hostname for a particular client connection.
- Another way to control access is to use a channel key, but since this key is sent in plaintext, it is vulnerable to traditional man in the middle attacks.
- IRC offer very little or absolutely no privacy, because it has no encryption algorithm to encrypt message before transferring between client and server.
- IRC cannot be applied to use within the organization that requires confidentially and integrity of the data.

2.10 Security Terminology

- **Encryption** is a process of encoding a message so that its meaning is not obvious.
- **Decryption** is the reverse process: transforming an encrypted message back into its normal form.
- A system for encryption and decryption is called a **cryptosystem**.
- The original form of a message is known as **plaintext**, and the encrypted form is called **ciphertext**.
- Both a cryptographer and a cryptanalyst attempt to translate coded material to its original form, normally a **cryptographer** works on behalf of legitimate sender or receiver, whereas a **cryptanalyst** works on behalf of an unauthorized interceptor.
- **Brute-force attack** is attempting to read the message by using all possible keys one by one until the messages is decrypted.
- **Man-in-the-middle attack** is the attack which an Internet hacker who may interfere with the initial public key exchange, by intercepting the very first message to a new correspondent and substituting genuine public key with a bogus one.
- **Spoofing** is a person who pretends to be someone else.
- **Confidentially** or **privacy** are preventing unauthorized disclosure.
- **Integrity** is preventing unauthorized modification.
- **Availability** is preventing denial of authorized access.
- **Authentication** is the process of verifying the user's identity.

2.11 Security Goals

- **Confidentially** means that the assets of a computing system are accessible only by authorized parties. Confidentially is sometimes called **secrecy** or **privacy**.
- **Integrity** means that assets can be modified only by authorized parties or only in authorized ways.
- **Availability** means that assets are accessible to authorized parties. An authorized party should not be prevented from accessing objects to which he, she, or it has legitimate access.

2.12 Threats to the security of a computing system

- **Interruption** – In an interruption, an asset of the system becomes lost, unavailable, or unusable.
- **Interception** – Some unauthorized party has gained access to an asset.
- **Modification** – Unauthorized party not only accesses, but tampers with an asset.
- **Fabrication** – Unauthorized party might fabricate counterfeit objects on a computing system.

The most powerful tool for prevent exploitation of the vulnerabilities of computer system is “Encryption”

2.13 Encryption

The most powerful tool in providing computer security is coding. By transforming data so that it is unintelligible to the outside observer, security professionals can virtually nullify the value of an interception and the possibility of a modification or a fabrication.

Encryption provides confidentiality for data. Additionally, encryption can be used to achieve integrity because data that cannot be read generally also cannot be changed in a meaningful manner. Furthermore, encryption is the basis of some protocols, which are agreed-upon sequences of actions to accomplish some task. Some protocols ensure availability of resources. Thus, encryption is at the heart of methods for ensuring all three goals of computer security.

2.13.1 Encryption Components : consists of 2 components

- 1.) An algorithm
 - 2.) A key
- **Encryption Algorithms** is a series of steps that mathematically transforms plain-text or other readable information into unintelligible cipher text. Cipher text - Data that has been encrypted. Cipher text is unreadable until it has been converted into plain text (decrypted) with a key. Decryption - The inverse mathematical transformation, which transforms the encrypted cipher text back into something readable.
 - **Encryption Keys** is a bit string consisting of x number of bits. A 40 bit key is a string consisting of 40 bits an encryption algorithm can use one of a large number of possible keys the number of possible keys each algorithm can support depends on the number of bits in the key. The longer the key, the more the possible number of keys.

2.13.2 Security and Encryption

Encryption algorithms are considered secure if the security depends on only one factor key length security which does not depend on secrecy, inaccessibility, or anything else, only on the key length, then the only possible attack against the algorithm is a brute force attack. All key combinations must be tried in order to find the correct key. The length of the key determines the possible number of keys available for selection and the longer the key length, the longer it takes to discover which key will actually decrypt, so, specifying a long enough key length makes a brute-force attack non-feasible.

2.14 Encryption Techniques

- There are two kinds of encryption techniques.

2.14.1 Symmetric encryption – Symmetric cryptography, otherwise known as secret key cryptography. Encryption and decryption use the same key K .

It has been in use for thousands of years in forms ranging from simple substitution ciphers to more complex constructions. However, developments in mathematics and the growth of computing power have made it possible to create ciphers that are effectively unbreakable. Symmetric systems are generally very fast but are vulnerable so that the key used to encrypt must be shared with whomever needs to decrypt the message.

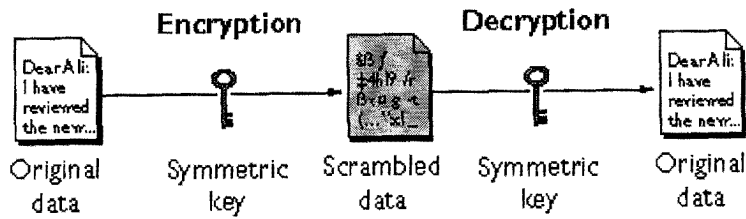


Figure 2-8 : Symmetric Encryption

2.14.2 Asymmetric encryption – The security afforded by asymmetric cryptosystem depends on mathematical problems that are difficult to solve, such as factoring large integers into primes. Public key systems use two keys, private and public key. Two keys are used and work together in such a way that plain text encrypted with the one key can only be decrypted with the other. One of these keys will typically be kept private by one individual, so there's almost no need to share keys, thus avoiding the risk of compromising security. The second key, the so-called public key, needs to be made as widely-known as possible.

The encryption/decryption process can work in both directions. That is, I can encrypt something using your public key and so ensure that only you can read it using your private key. I can also encrypt something using my private key and although this is a pointless exercise from the point of view of information hiding -- as anyone with access to my public key can decrypt the message -- it is the basis of a system of authentication, confirming that the message could only have originated from someone with access to my private key, presumably me.

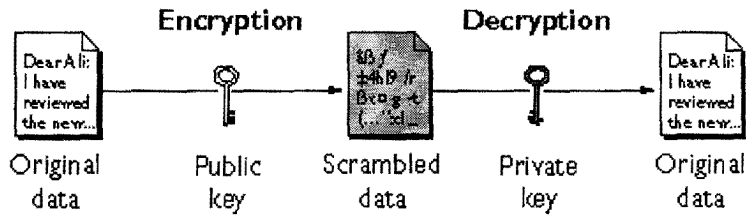


Figure 2-9 : Asymmetric Encryption

2.15 DES cryptosystem

Data Encryption Standard (DES) is the formal description, derived from work done by IBM and adopted officially by the US government in 1977. It is probably the most widely used secret key system, particularly in securing financial data, and was originally developed to be embedded in hardware, such as an ATM.

DES Encryption algorithms shown in Figure 2-10 are rather complex, however there are two basic steps: confusion and diffusion.

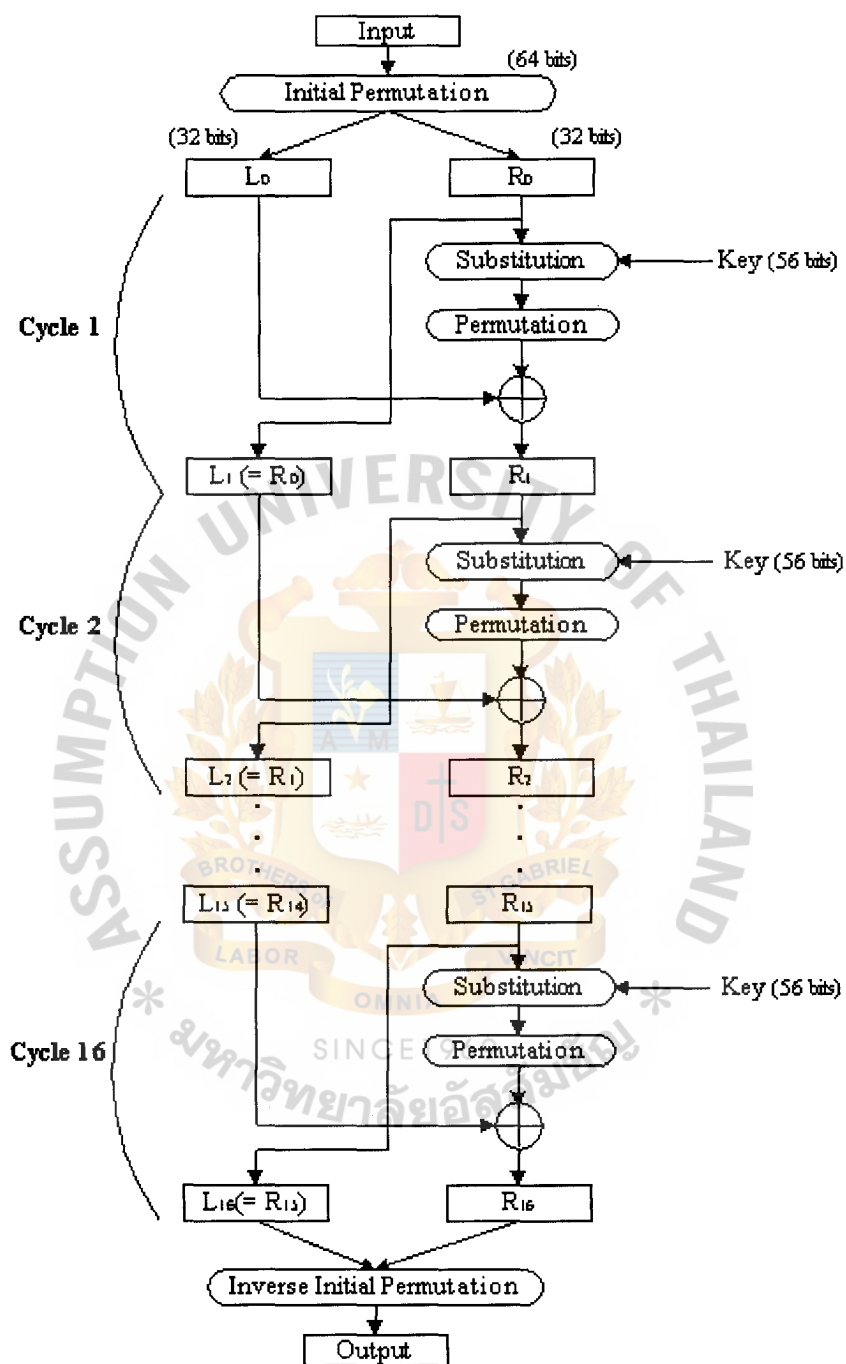


Figure 2-10 : DES Encryption Algorithm

DES is a block cipher, which works in blocks of 64 bits. (DES uses a 56-bit key with the additional eight parity bits to bring the block size up to 64 bits). It's an iterated block cipher using what's known as Feistel techniques where the text block being encrypted is split into two halves. The round function is applied to one half using a subkey and that output is then XORed with the other half; the two halves are then swapped and the process continues except that the last round is not swapped. To complete the algorithm, this process is repeated 16 times (DES uses 16 rounds).

2.16 Triple DES cryptosystem

A common variant on DES is **triple-DES**, a mechanism that encrypts the material three times using three different, unrelated keys; this generally provides considerably more security.

Triple DES is a minor variation of this standard. It is three times slower than regular DES but can be billions of times more secure if used properly. Triple DES enjoys much wider use than DES because DES is so easy to break with today's rapidly advancing technology. In 1998 the Electronic Frontier Foundation, using a specially developed computer called the DES Cracker, managed to break DES in less than 3 days. And this was done for under \$250,000. The encryption chip that powered the DES Cracker was capable of processing 88 billion keys per second. In addition, it has been shown that for a cost of one million dollars, a dedicated hardware device can be built that can search all possible DES keys in about 3.5 hours. This just serves to illustrate that any organization with moderate resources can break through DES with very little effort these days. No sane security expert would consider using DES to protect data.

Triple DES was the answer to many of the shortcomings of DES. Since it is based on the DES algorithm, it is very easy to modify existing software to use Triple DES. It also has the advantage of proven reliability and a longer key length that eliminates many of the shortcut attacks that can be used to reduce the amount of time it takes to break DES. However, even this more powerful version of DES may not be strong enough to protect data for very much longer. The DES algorithm itself has become obsolete and is in need of replacement. For the foreseeable future, Triple DES is an excellent and reliable choice for the security needs of highly sensitive information.

(Triple DES is simply another mode of DES operation. It takes three 64-bit keys, for an overall key length of 192 bits. In Stealth, you simply type in the entire 192-bit (24 character) key rather than entering each of the three keys individually. The Triple DES DLL then breaks the user provided key into three subkeys, padding the keys if necessary so they are each 64 bits long. The procedure for encryption is exactly the same as regular DES, but it is repeated three times. Hence the name Triple DES. The data is encrypted with the first key, decrypted with the second key, and finally encrypted again with the third key.) (Triple DES encryption algorithm shown in Figure 2-11)

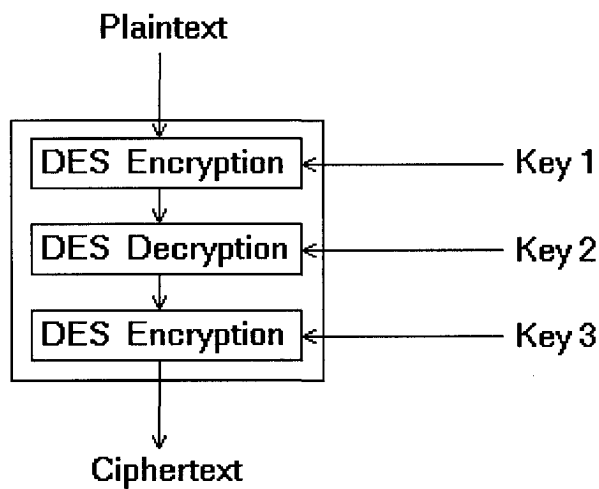


Figure 2-11 : Triple DES Encryption Algorithm

Unfortunately, there are some weak keys that one should be aware of: if all three keys, the first and second keys, or the second and third keys are the same, then the encryption procedure is essentially the same as standard DES.

Note that although the input key for DES is 64 bits long, the actual key used by DES is only 56 bits in length. The least significant (right-most) bit in each byte is a parity bit, and should be set so that there are always an odd number of 1s in every byte. These parity bits are ignored, so only the seven most significant bits of each byte are used, resulting in a key length of 56 bits. This means that the effective key strength for Triple DES is actually 168 bits because each of the three keys contains 8 parity bits that are not used during the encryption process.

2.17 RSA Cryptosystem

In 1977, three researchers at MIT -- Ron Rivest, Adi Shamir, and Leonard Adelman -- developed a practical method that initials the most widely-used public key cryptosystem, this became known as RSA. It was patented in the US in 1983, duly adopted as a standard, and has always been widely available outside the US in implementations developed locally even though, until recently, its export was restricted.

RSA Algorithm

As with other such systems, RSA uses large prime numbers to construct the key pairs. Each pair shares the product of two primes, the modulus, but each also has a specific exponent. RSA Laboratories explains the working of the RSA cryptosystem as follows:

- 1.) Take two large primes, p and q
- 2.) Compute their product $n = pq$; n is the modulus.
- 3.) Let $m = (p-1)(q-1)$
- 4.) Choose a number, e , less than n and relatively prime to m ,
which means that e and m have no common factors except 1.
- 5.) Find another number d , $de \bmod m = 1$
- 6.) The values e and d are called the public and private exponents, respectively.
- 7.) The **public key** is the pair (n, e)
- 8.) The **private key** is (n, d)
- 9.) Encryption algorithm is $c = pe \bmod n$
- 10.) Decryption algorithm is $p = cd \bmod n$

Knowledge of the public key offers a route to obtaining the private key but this depends on factoring the modulus into its component primes. This is difficult and can be made essentially impossible by choosing keys of adequate length. The measurement needed is the length of the modulus; RSA Laboratories currently recommends key sizes of 1024 bits for general corporate use and double that, 2048 bits, for extremely valuable material. For ordinary use, key lengths of 768 bits are adequate as these cannot readily be broken using current techniques. As always, the cost of securing material needs to be considered in conjunction with the material's value and whether the costs of breaking the protection might be excessive. RSA Laboratories mentions a recent study of RSA key-size security based on factoring techniques available in 1995. This study suggested that a 512-bit key might be factored for less than \$1 million in eight months of effort. In fact, a particular RSA 512-bit number, known as RSA-155, was factored in seven months during 1999 as part of the regular RSA Security challenge.

2.18 One-way Encryption (One-way Hash Function)

Encryption with one-way algorithms (also known as hashes), one-way hash functions squeeze message reduces a large amount of data to a smaller result, called a *digest* or *check value*. A hash function H is said to be *one-way* because it is hard to invert. In such a way that no information about the message can be revealed from the digest, moreover one cannot produce a message for a given digest. Hash functions have several uses:

Data integrity: insure that the received data was not corrupted. The sender appends the digest to the message; the receiver recalculates the digest and compares

the result to the received digest. Any difference means that the message was corrupted.

Message compression as part of digital signature protocol.

Keyed Hash Message Authentication Code (HMAC): incorporates a secret key into existing hash algorithm. Only if the sender and the receiver have the same key, they will calculate the same digest.

There are many well-known hash functions used in cryptography. These include the message digest hash functions MD2, MD4, and MD5, used for hashing digital signatures into a shorter value called a message-digest, and a commonly employed hash is SHA-1 (secure hashing algorithm) that generates 160 bit hashes.

2.19 SHA-1

The Secure Hash Algorithm (SHA-1) was developed by the National Institute of Standards and Technology, a division of the US Department of Commerce, it is published in 1994.

SHA-1 generates a message digest of 160 bits and is considered secure. It is widely believed in the cryptographic community that SHA-1 is a collision-resistant, one-way function. Indeed no cryptanalyst has yet announced the discovery of collision in SHA-1.

CHAPTER 3

BUSINESS IRC MODEL (bIRC)

The conventional IRC does not have effective security to verify legitimate person, and protect data from an eavesdropper, to read and modify as they pass between the insecure communication path. So the conventional IRC should be extended an effective security model to it. Using a strong authentication and encryption method based on cryptography will make IRC suitable for private personal conversation, and can be applied to use within the business organization that required secrete conference via the public network (Internet).

This thesis focuses on designing a security model by adding three cryptography algorithms, the first one is RSA algorithm that is used for client-server authentication, the second is SHA-1 that is used for integrity of data (password's integrity, server public key's integrity), and the third is 3DES that is used to encrypt messages. Additional uses are random functions to generate random data for client-server authentication and protection play back messages, designs secure public key distribution to protect man-in-the-middle attack. Another part except authentication and encryption part will remain the same as the conventional IRC.

The bIRC model has many processes, pre-define at server's site, pre-define at client's site, client-server authentication process, and encryption during communication process.

3.1 Overview of bIRC model

This model is a client-server model, has both one-to-one and one-to-many communication types, has a client-server authentication part to verify each other before establishing connection, and all messages travelling in channel are cipher-text.

Cryptography method and other additional features supporting the model

- **One-way hashing function (SHA-1 algorithm 128 bits key)** will be used in part of integrity of server's public key that is buried in the client's program, and used for integrity password in server's database.
- **Asymmetric key scheme (RSA algorithm 2048 bits key)** is used in the part of request to establish connection between client and server. This scheme is used to encrypt and decrypt necessary information (such as User ID, password, client's public key, random number, etc.) that is passed along the communication network before creating a session key for client-server authentication.
- **Random number** which is generated from random function that is used for client-server authentication (if the correct random number is sent back to generator, so the receiver is authenticity), and used to prevent play back messages.
- **Symmetric key scheme (Triple DES 168 bits key)** is used to generate session key and channel key that is used to encrypt messages. Session key is created randomly every time when clients establish connection to server. Channel key is created randomly and generated every 24 hours.

3.2 Pre-definition of model at server's site

The following step is needed to pre-define at server's site only one time when setting up an IRC server. (See flow of pre-defined step of server at Figure 3-1)

- Server generates one key pair, server's public key and server's private key.
- Server stores server's public key into server's database.
- Server use server's public key to generate digestion of server's public key by using one-way hashing function
- Bury server's public key and digest server's public key into client's chat program.
- Server creates channel, uses Triple DES algorithm to create channel's key, and saves it in server's database. Channel's key will be generated randomly by using Triple DES algorithm every 24 hours.

The following step is needed to pre-define at server's site only one time when there is a new user (client).

- Server creates user ID and password of authorized user
- Server uses one-way hashing function to digest password, and save user ID and digested password into server's database

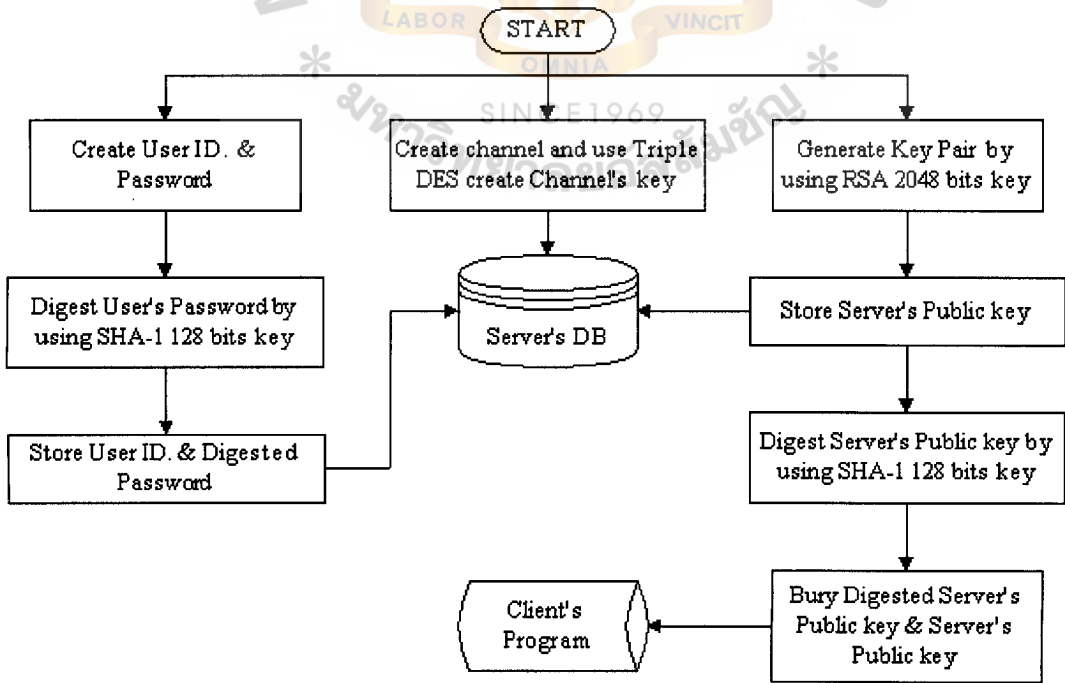


Figure 3-1 : Step of pre-defining model at server's site

3.3 Pre-definition of model at client’s site

The following step is needed to pre-define at client’s site only one time when installing client’s chat program. (See flow of pre-define step of client at Figure 3-2)

- Client installs the client's chat program and fills the information : Name, Surname, Address, Telephone number, Birthday, User ID for logon and password
- Client uses the above information to generate one key pair, client's public key and client's private key
- Client encrypts server's digital signature by using client's public key

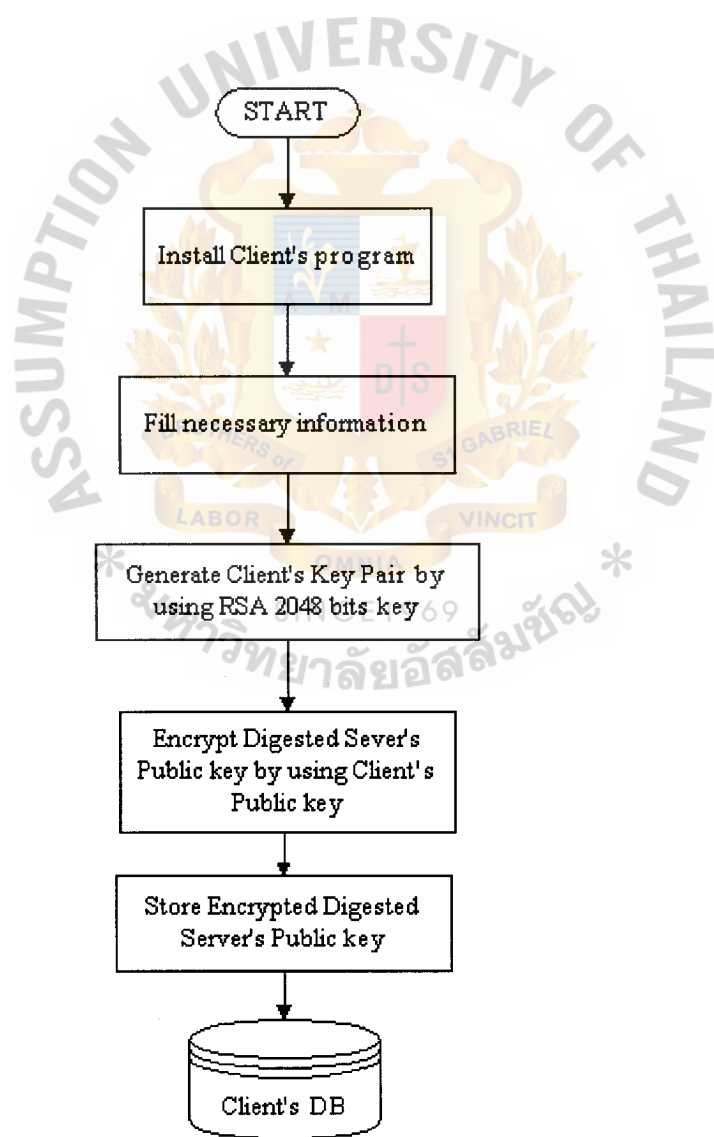


Figure 3-2 : Step of pre-defining model at client’s site

3.4 Client-Server Authentication process of the model

The following four processes are designed to authenticate each other between client and server before establish connection

3.4.1 The first process "Client verifies Server's Public key that is buried in client's Database valid or not" (Shown in Figure 3-3)

- Client decrypts encrypted digest server's public key by using client's private key.
- Duplicate server's public key, digest one of server's public key by using SHA-1, receive digested server's public key.
- Compare the digested server's public key that is generated by client with the old one, if it does not match, server's public key is invalid, send error message to user and terminate program. If it matches, server's public key is valid, continue the second process.

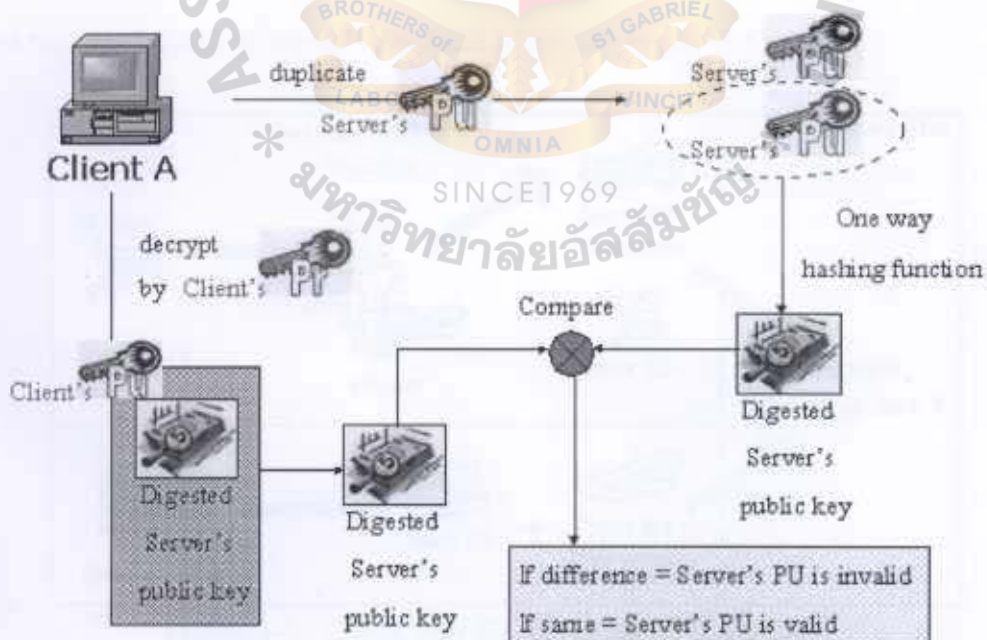


Figure 3-3 : (First process) Client verify server's public key valid or not

3.4.2 The second process “Client identifies itself to the server by using User ID., password”

- Client will generate Random data (C1) and store it in client’s memory. Client encrypts User ID, Password, Public key, Random data (C1) by using server’s public key, and send it to server. Server uses server’s private key to decrypt. This process is shown in Figure 3-4.

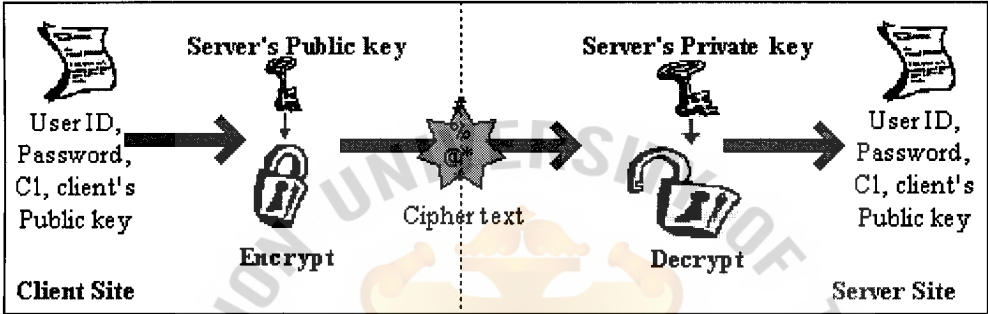


Figure 3-4 : Client’s encryption and Server’s decryption process

- Server will digest password by using SHA-1 to digested password, and compare with the pre-defined in server’s database (shown in Figure 3-5).

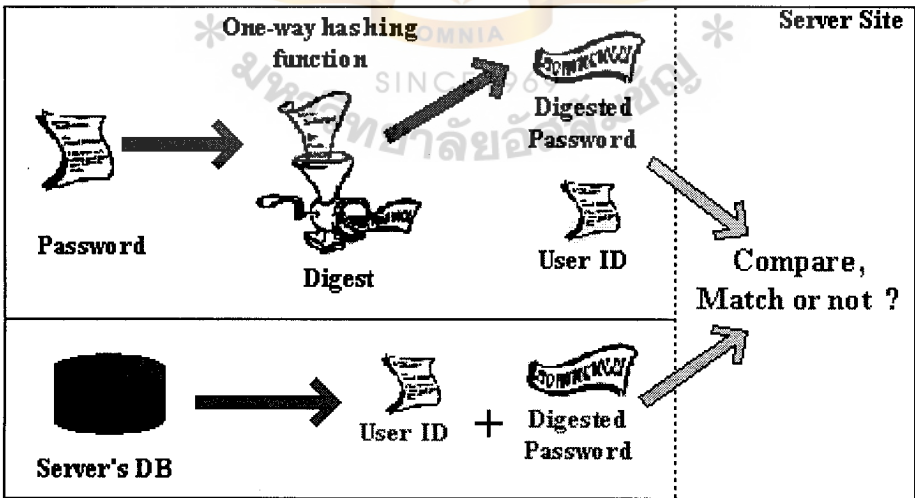


Figure 3-5 : Check user ID. and digested password

- If the user ID and/or digested password doesn't match with the pre-defined in the server's database, server will reject that client. Figure 3-6 is illustration of server rejects client.

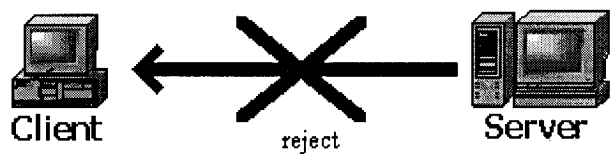


Figure 3-6 : Server rejects, and terminates connection.

- If this user ID and digested password matches with the pre-defined in the server's database, server will generate new Random data (C2), store Client's public key and random data (C1) and random data (C2) into server's memory.

3.4.3 The third process *"Client verifies server authorized or not by checking random data (C1)"*

- Server encrypts random data (C1,C2) by using client's public key, and sends it to client. Client decrypts it by using client's private key shown in Figure 3-7.

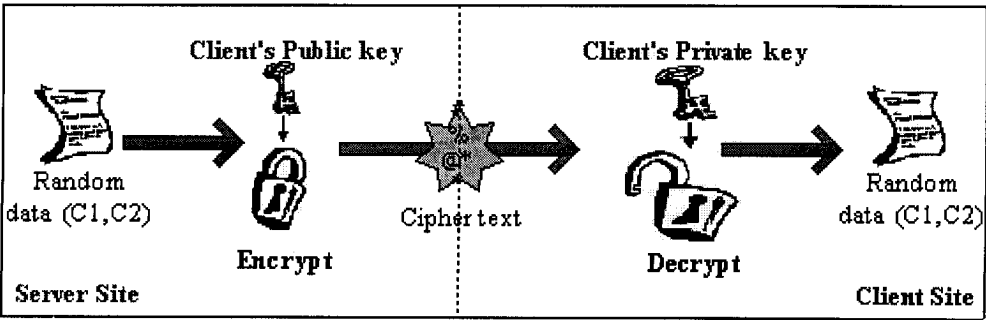


Figure 3-7 : Server's encryption and Client's decryption process

- Client checks random data (C1) that is received from server, if it is same as the old one that client generated or not. Figure 3-8 is illustration of comparison of random data (C2) for authenticate server.

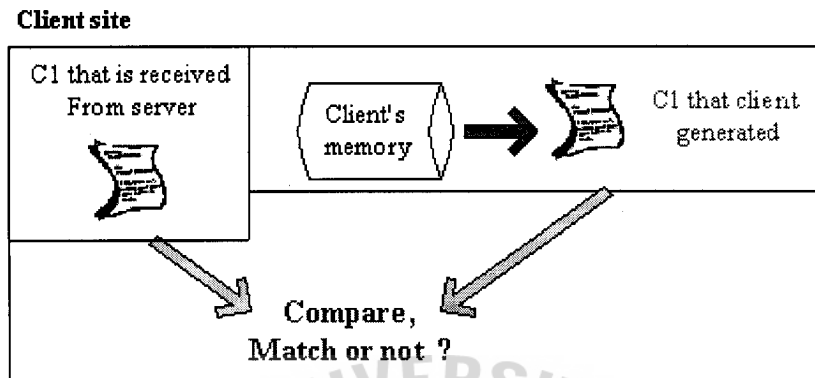


Figure 3-8 : Compare random data (C1) to authenticate server

- If it does not match, server is unauthorized and client terminates connection.
- If match server is authorized, client generates random data (pre-master secret (pre-MS)) by using random function, store random data (C2, pre-MS) in memory.
- Client uses random data (C1, C2, pre-MS) and Triple-DES algorithm to generate session key, stores session key into memory, and continue the fourth process.

3.4.4 The fourth process “Server verifies client authorized or not by checking random data (C2)”

- Client encrypts random data (C2, pre-MS) by using server’s public key, and sends it to server. Server decrypts it by using server’s private key.
- Server checks random data (C2) that is received from client, if it is same as the old one that server generated or not. Figure 3-9 is illustration of comparison random data (C2) to authenticate client.

Server site

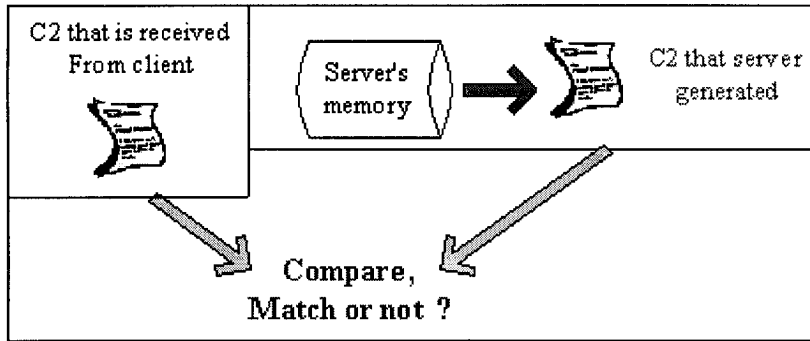


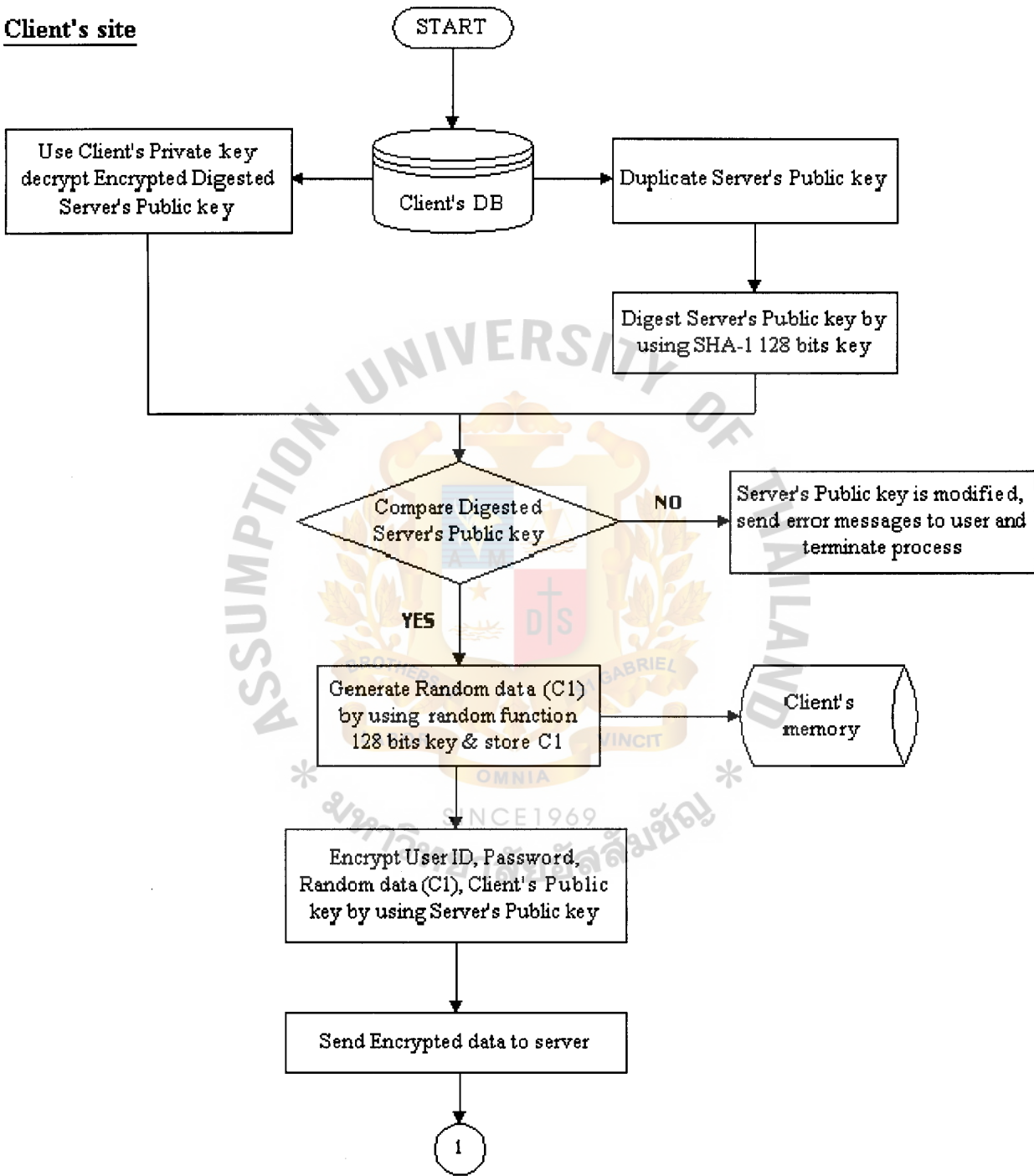
Figure 3-9 : Compare random data (C2) for authenticate client

- If it does not match, client is unauthorized and server terminates connection.
- If match client is authorized, server stores random data (pre-MS) in memory.
- Server uses random data (C1, C2, pre-MS) and Triple-DES algorithm to generate session key, and stores session key into memory.

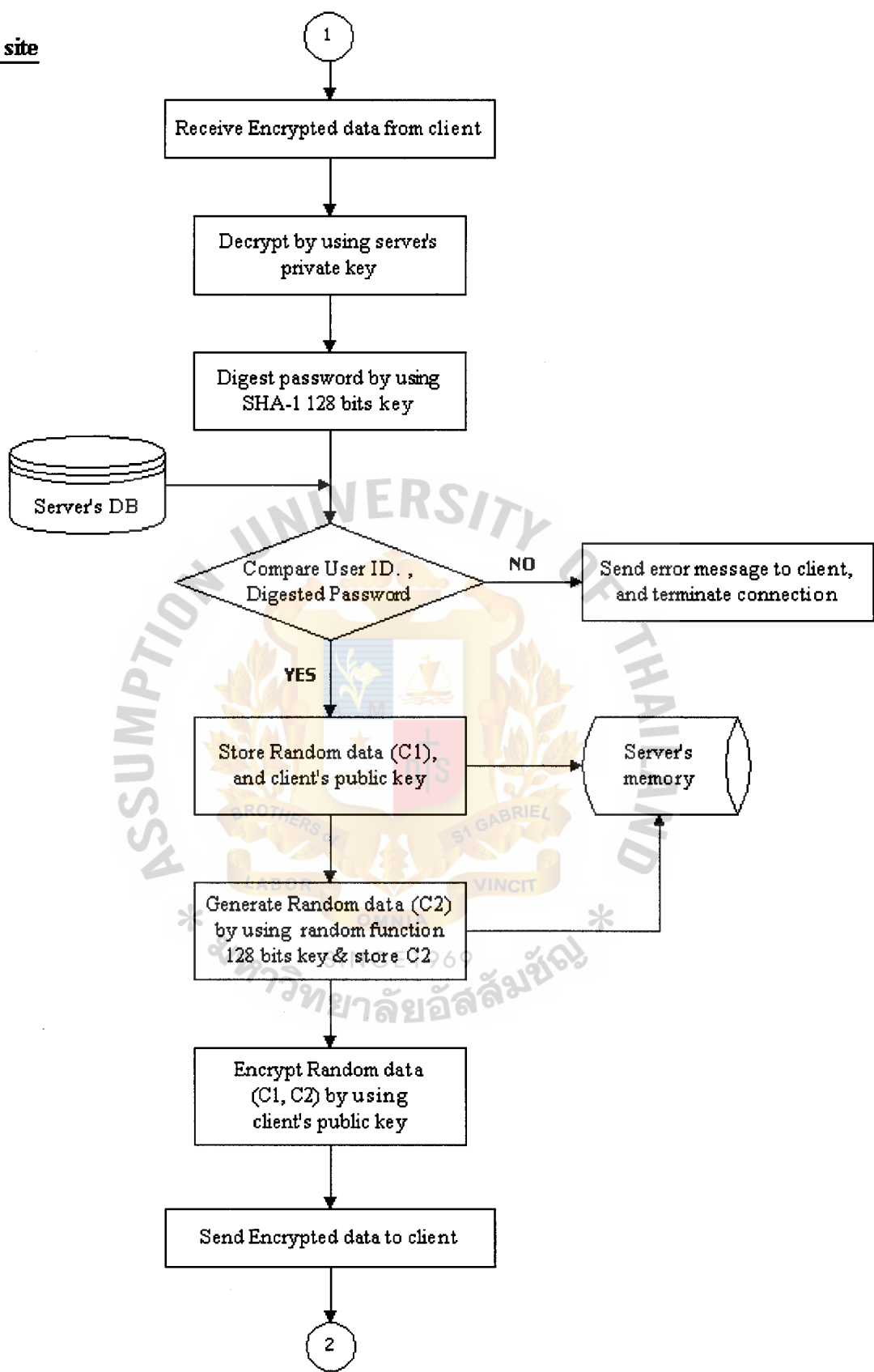


The Figure 3-10 shows all authentication processes when a client requests to establish session to IRC server.

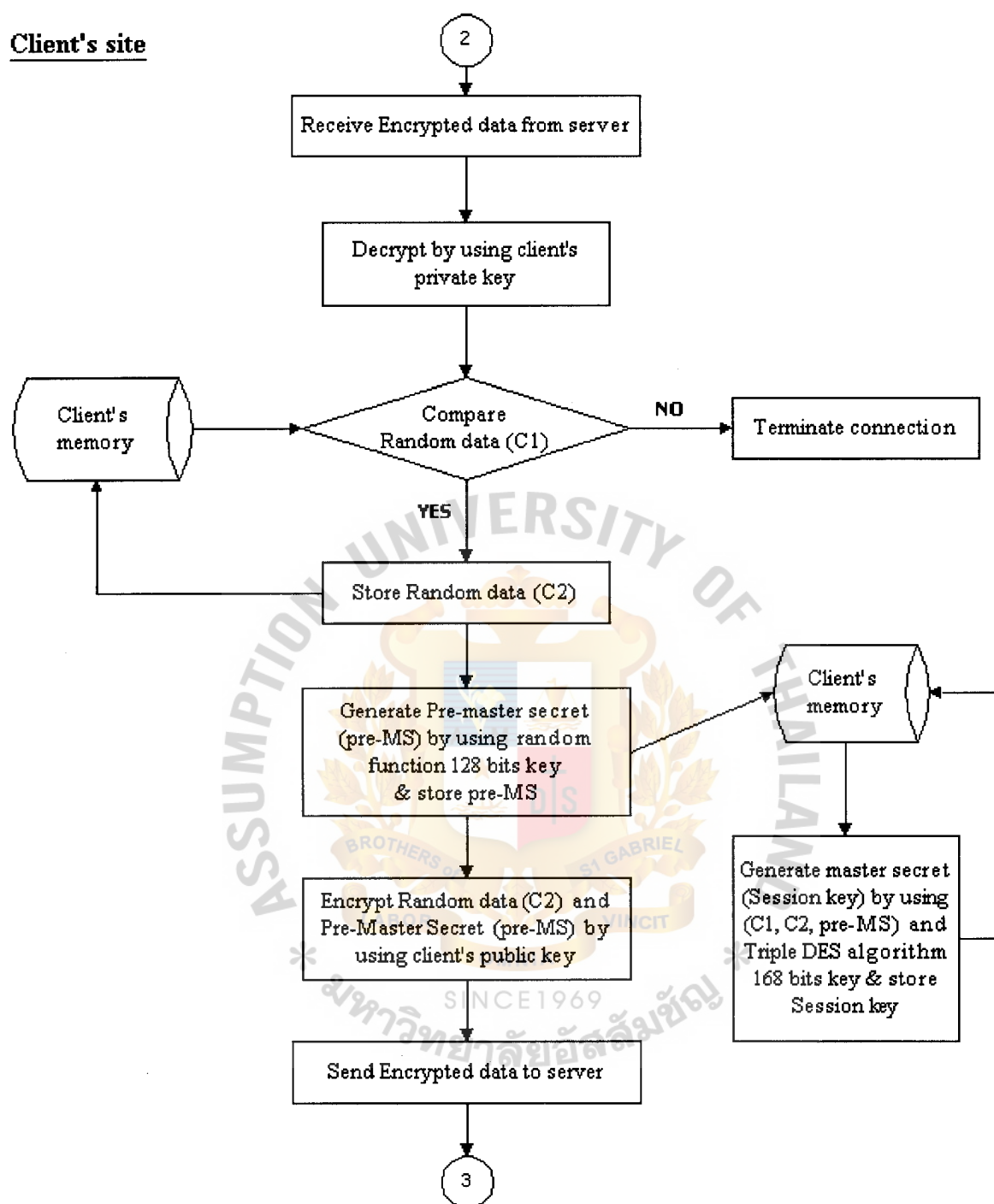
Client's site



Server's site



Client's site



Server's site

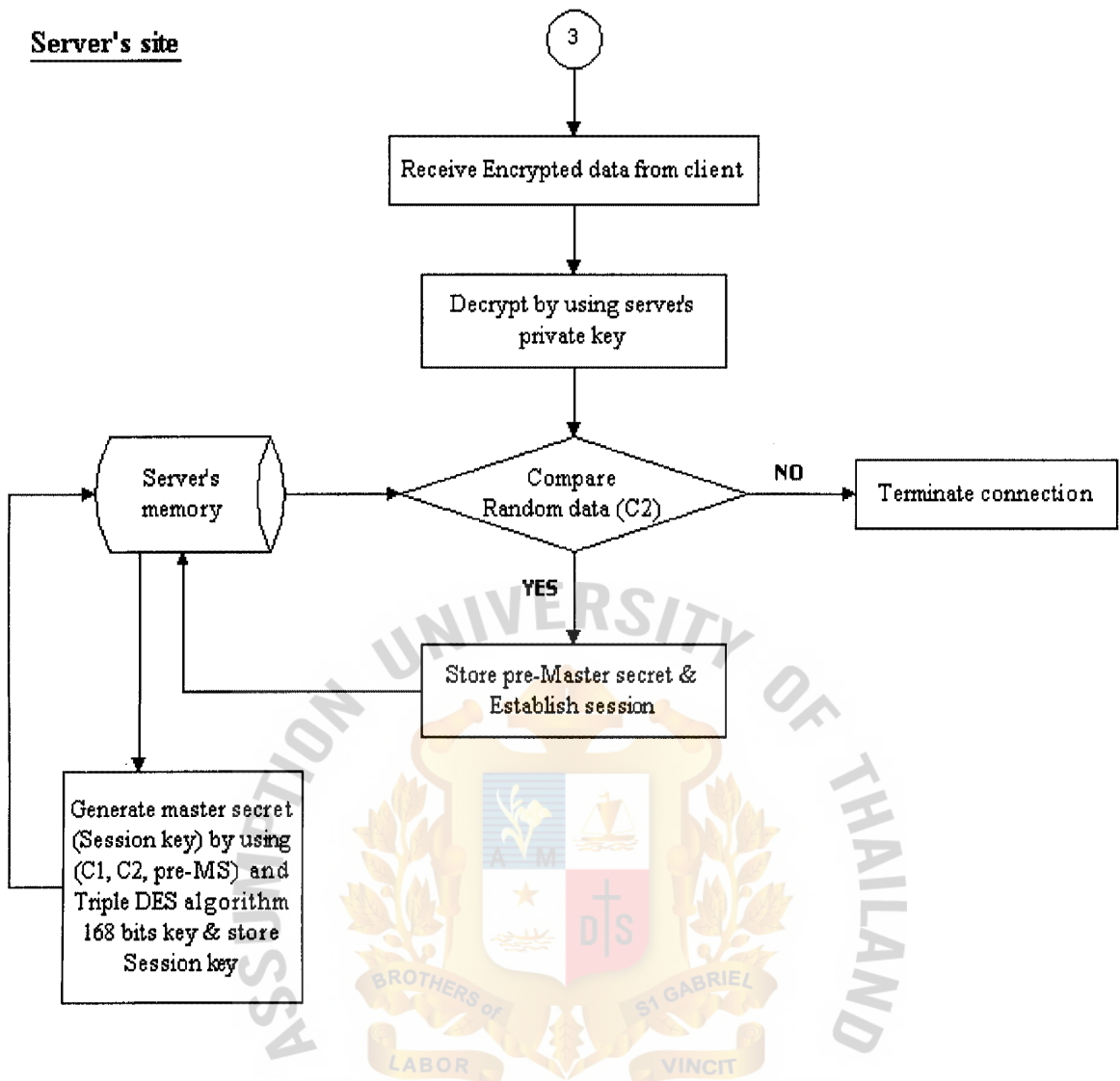


Figure 3-10 : All authentication process when client request to establish session to IRC server

3.5 Communication in channel

After client establishes connection to the server, client begins talking in channel by requesting to join into the available channel at server. Server will send channel key to client to use to encrypt and decrypt message in that channel.

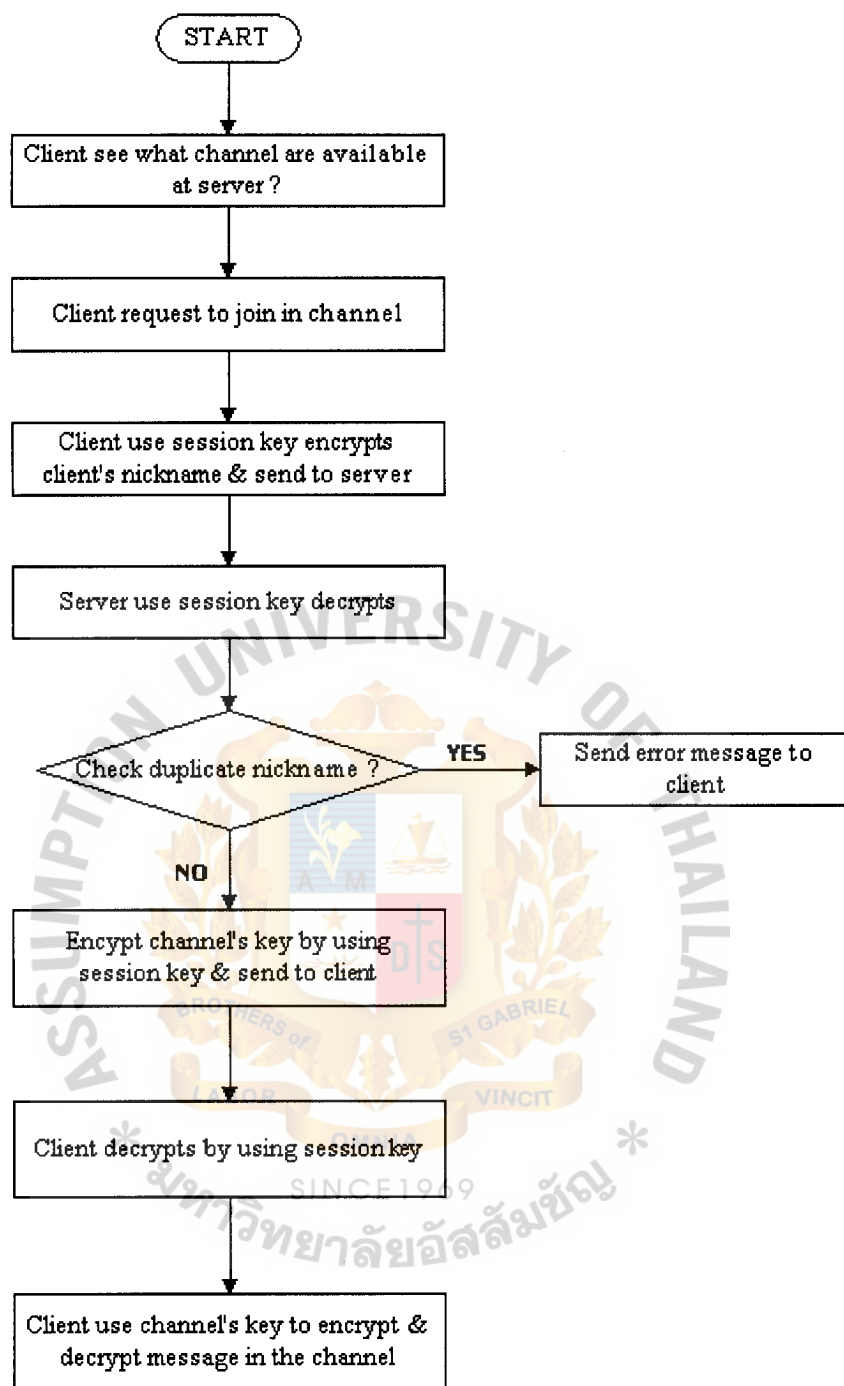


Figure 3-11 : Requesting to join in channel process

After client receives channel key from server, client can start talking in channel by typing a message, and the message is automatically encrypted by channel key, and forwarded to server, server forwards the encrypted message to every client in the

same channel, then every client in that channel receives the message, and automatically decrypts message, and clients can read message

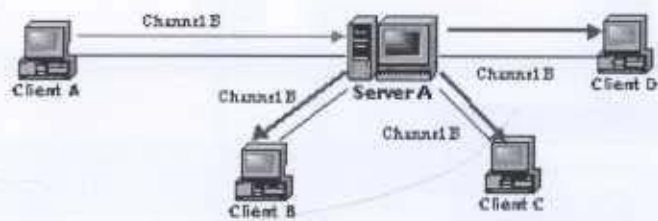


Figure 3-12 : Flow of message in the channel

All messages in channel are in encrypted format (cipher-text). Encryption and decryption are performed by clients.

3.6 Sample screen to communicate in channel of the model

The following pictures are the examples of chat screen, and examples of communication in channel of this model

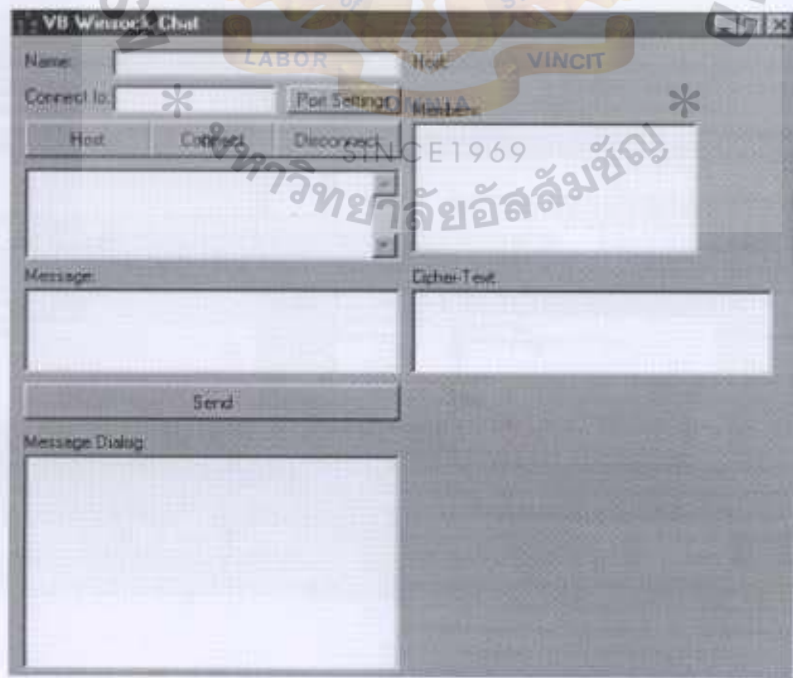


Figure 3-13 : Example of screen for chatting

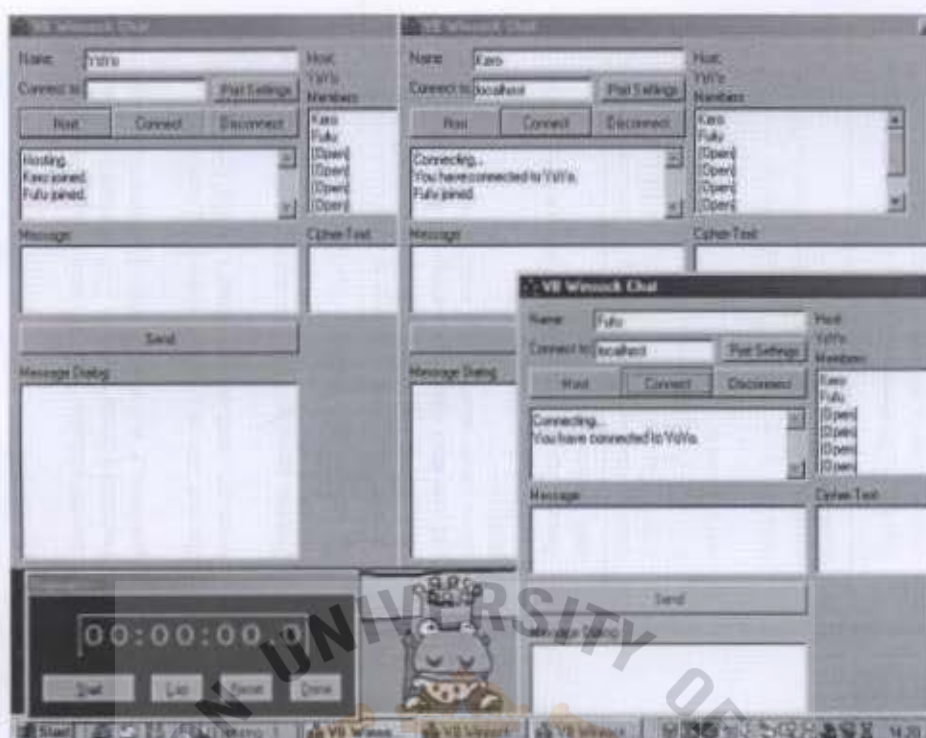


Figure 3-14: Example of screen when client joins to Host(server) for chatting

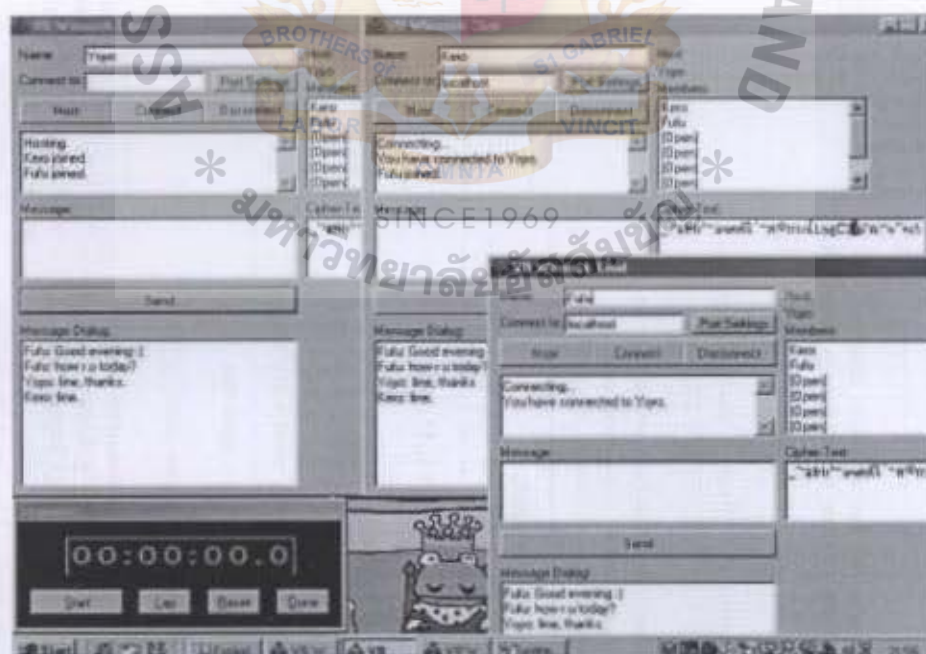


Figure 3-15 : Example of screen when clients are chatting in channel



CLIENT

Client duplicates server's public key, uses one-way hashing function to digest one of server's public key to digested server's public key

Compare digested server's public key with server's digital signature, if not match, show error message, if match, continue process

Client generates a "random data" (C1)

C1 is stored in memory for later use

Client decrypts by using Client's private key, checks random data (C1), if it is same as the old one that client generated, if not match, show error message, if match, continue process

Client keeps random data (C2) into memory

Client generates a new "random data" or "Pre-Master Secret (Pre-MS)"

Client derives Session key or Master Secret (MS) (for encrypt message) combining both C1, C2, Pre-MS using the same function as server uses

Session Key = Fx (C1, C2, Pre-MS)
keeps Session key or Master Secret (MS) into memory

Client see what channel are available at server, and requests to join the available channel

Client decrypts by using session key, and receives channel key

Client uses channel key to encrypt and decrypt message in the channel

Data Flow and Information Exchanged

Client sends Encrypted (C1, Client's public key, User ID, password) by Server's public key

Server sends Encrypted (C1, C2) by Client's public key

Client sends Encrypted (C2, Pre-MS) by Server's public key

Client sends Encrypted (nickname) by MS

Server sends Encrypted (channel key) by session key

Encrypted message in channel



SERVER

Server decrypts by using server's private key

Server uses one-way hashing function to digest password.

Server checks User ID and compares digested password from server's DB, if match continue process, if not match, reject connection request

Server stores random data (C1) into the memory, and stores client's public key in DB

Server generates a "random data" (C2)

C2 is stored in memory for later use

Server decrypts by using server's private key, checks random data (C2), if it is same as the old one that client generated, if not match, show error message, if match, continue process

Server keeps Pre-Master Secret (Pre-MS) into memory

Server derives Session key or Master Secret (MS) (for encrypt message) combining both C1, C2, Pre-MS using the same function as server uses

Session Key = Fx (C1, C2, Pre-MS)
keeps Session key or Master Secret (MS) into memory

Server uses session key to decrypt message, check client's nickname duplicate or not, if duplicate, send error message, if not duplicate, continue process

Figure 3-16 : Establish connection (Authentication and Communication process)

CHAPTER 4

COMPARISON, ANALYSIS AND EVALUATION

4.1 Security Comparison and Analysis

There are many chat programs on the Internet, the popular chat programs are SNC (Secure Network Chat), Zephyr, mIRC, Pirch. Most of them provide different security levels to users, some programs do not provide any security feature to users, but some programs provide some security features to users.

The following table shows the different security techniques that are used in the conventional chats and in the alternative security model (Business IRC or bIRC model).

Chat Program	Authentication	Encryption	Remark
1 IRC / mIRC	Optional (Based on password)	None	Password is sent in clear text
2 AnotherNet / PIRCH	None	None	-
3 SNC	None	Have (Blowfish 448 bits key)	-
4 Zephyr	Have (Kerberos - DES) (Symmetric key)	Have (Kerberos - DES) (Symmetric key)	-
5 bIRC model	Have (RSA) (Use Asymmetric key & password)	Have (Triple-DES) (Use Symmetric key)	Have SHA-1 for integrity

Table 4-1 : Security feature of other well known chats versus bIRC model

Table 4-1 shows that, each chat programs have different security techniques, and each technique has a difference security level, then the following figure shows the security level in theory of each technique.

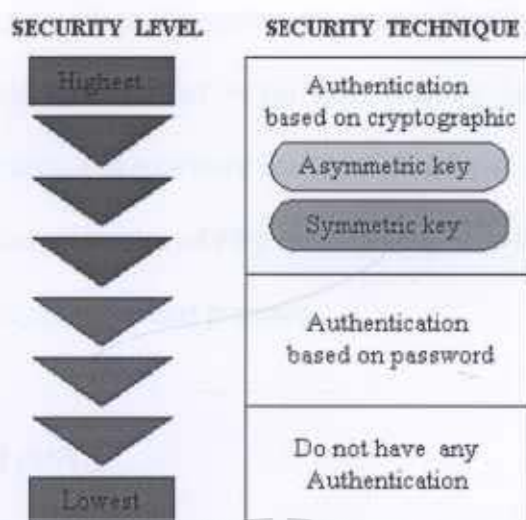


Figure 4-1 : Security Level classified by security technique

In theory, a system which has authentication based on password is more secure than nothing, and authentication based on cryptographic is more secure than authentication based on password. Asymmetric key cryptography has a higher level of security than symmetric key cryptography.

This shows that most of the chat programs on the Internet always provide very little or absolutely no security in them, but the bIRC model will increase more security, so the following is the conclusion of security analysis of each conventional Chat and bIRC.

Internet Relay Chat (IRC), mIRC, AnotherNet and PIRCH

They afford no message encryption, minimal authentication (identification is sometimes used, which is almost worse than nothing at all, because it uses password for authentication, but password is sent in clear-text).

They are perfect examples of why security should be designed into network protocols rather than relying on social pressure.

Secure Network Chat (SNC)

All the client-server data communications protocol is encrypted with Blowfish, the most reliable contemporary algorithm, and with 448 bit key, but it does not have authentication process, so it is especially designed to work with medium and large local area networks (LAN), home and local office, where verifying a user is not important.

Zephyr

Zephyr is a chat system originally developed by MIT Athena as a system message service, now used as a chat system at MIT and other locations.

It is tied to the Kerberos authentication system, providing relatively secure but complicated authentication. Zephyr is very complex to set up if you don't run Kerberos, as Kerberos itself is a major effort to configure. It is also the least actively developed and maintained of the various chat system, and is only minimally cross-platform. Kerberos is designed to provide authentication by using symmetric key cryptography (DES algorithm), but now DES algorithm can be broken within very little time, and low cost, so no security expert would consider using DES to protect data.

bIRC

The bIRC model will increase security because bIRC model provides the authentication process based on asymmetric key (RSA algorithm), and also has authentication based on password, to verify a legitimate person before establishing connection between client and server, and it is well designed to eliminate man-in-the-middle attack problem.

It uses random function to create random number to authenticate each other between client-server, and prevent playback message.

The model's encryption process uses symmetric key (Triple DES) to encrypt message before sending a message travelling through network (To preventing unauthorized parties disclosure the messages).

It uses one-way hashing function (SHA-1) to create message digest for message integrity (To protect password in server's Database, because password cannot be revealed from the digested password).

The bIRC model can be applied for use within the business organization to create private conferencing via insecure public network. Most people and business companies can identify if people, companies, and computers are valid or not, and can protect the essential data from the unauthorized (attacker).

It can be conclude that, in theory, bIRC model has a higher security level than other conventional chat systems from combining asymmetric key cryptography in authentication part with symmetric key cryptography in communication process.

According to this fact, the proper way to evaluate the bIRC model is to measure the efficiency of the system which is implementing this model. Because chat is a real-time conversational system, so implementing bIRC model should not affect

the efficiency of its real-time property of chat system, then it should find the factor which may be affected to increase time of system. The additional process of communication in channel which may degrade the efficiency of system is encryption and decryption messages, then it should design the experimental process to evaluate the additional time that is spent during communication process between clients in the channel.

4.2 Experimental and Evaluation

The interesting point of the experiment is the time that a client takes for encryption and decryption message, because it is mostly time that users are involved when talking in the chat system. Another interesting point is whether size of messages after encryption (cipher-text) will increase or not, because cipher-text will be sent to travel into the network.

This experiment is designed to find the summary of encryption and decryption time of Triple-DES algorithm 192 bit key that used in the bIRC model when there are many different sizes of messages. Tested messages 400, 800, 1600 and 3200 characters (bytes) (or 5 lines, 10 lines, 20 lines and 40 lines).

The following is the testing environment:

- CPU = AMD 600 MHz.
- RAM = 64 Mb.
- OS = Windows 98
- Encryption algorithm = Triple-DES 192 bit key
- Encryption file type = DLL file

- Secret key = tp242021yooh1234tharnkam
- Timer = Timer function in VB
- Timer scale = 0.00 seconds

Every case will be tested fifty times. Finding the result by deletion maximum value, summation the remainder values, dividend it by forty-nine. (Find Mean value)
 (Remark: Maximum value is time for loading DLL file for encryption and decryption messages, it occurs when using encryption and decryption function in the first time after opening chat program. The maximum value is around 0.03-0.10 seconds.)

The experiment results conclusion table

Length of message (characters/bytes)	Cipher-Text (bytes)	Encryption Time (seconds)	Decryption Time (seconds)	Total Time (seconds)
400	400	less than 0.01	less than 0.01	less than 0.02
800	800	less than 0.01	less than 0.01	less than 0.02
1600	1600	less than 0.01	less than 0.01	less than 0.02
3200	3200	less than 0.01	less than 0.01	less than 0.02

Table 4-2 : Experiment results conclusion table

The experimental results and analysis...

- The transaction time which a client uses for encrypt message and decrypt message

400 characters (5 lines)	=	less than 0.02	seconds
800 characters (10 lines)	=	less than 0.02	seconds
1600 characters (20 lines)	=	less than 0.02	seconds
3200 characters (40 lines)	=	less than 0.02	seconds

- The different sizes of messages have the effect on the transaction time that a client used in encryption-decryption process, larger sized of messages take slower encryption-decryption time than the smaller one. In fact, the conversation that you usually use is a short message, so its length is usually not longer than 400 characters or 5 lines. The encryption-decryption time is less than 0.02 seconds.

- The transaction time of encryption-decryption is very little when compared with the time that a user uses when thinking and typing message, so users cannot feel the difference from other IRC models that do not implement this security model.

- From the experiment, size of messages after encryption is same as the original size, so this encryption algorithm does not increase size of messages.

- The encryption algorithm will not degrade the speed of the chat system, so the additional model does not have any effect on the performance of this chat system.

CHAPTER 5

CONCLUSION AND RECOMMENDATION

5.1 Conclusions

Nowadays Chat is a popular communication tool on the Internet, because it provides real-time conference for users around the world at a lower cost than other communication tools, but it has some limitations about security. So it cannot be applied for use within the business organization or private conferencing. Most people and business companies need more security methods to identify if people, companies, and computers are valid or not, and also need some security methods to protect the essential data from the unauthorized (attacker) when they use chatting system via public network.

The bIRC model is extended for supporting authentication in distributed systems by using asymmetric key algorithm (RSA algorithm 2048 bit key), using authentication based on password, and using Random number password that is created randomly. Moreover every message must be encrypted by using Triple-DES algorithm 168 bit key before transfer travelling in channel.

Authentication is the important part to identify if the client site is the exact one that can access the server, and identify if the server site is the real server. Encryption message is also important to privacy and integrity of data. Therefore by implementing the bIRC model in IRC, every person and business companies can use IRC like a secure conferencing room in a distributed network architecture at a low cost. The bIRC model can eliminate the security problem of IRC protocol, and is more secure than other conventional chat systems, and it does not decrease the efficiency of the chat system.

5.2 Recommendation

The bIRC model can be extended to provide more private communication to a client by allowing the client to create his own channel, and the client can set his own channel key, classifying who can access the channel. If other users request to join in your own channel, server will send the request (including user information, user's public key) to you for allowing, if you allow, you encrypt channel key by using user's public key and send it to requested user. The requested user uses private key to decrypt and uses the channel key to encrypt and decrypt messages in this channel. (This extended option can protect untrustworthy server)

The bIRC model is designed to be used for enhancing the security in IRC protocol, provide authentication based on asymmetric key algorithm, authentication based on password, and also provide privacy to message by using symmetric key to encrypt message that is sent between clients.

Though the bIRC model is a powerful model to prevent exploitation of the vulnerabilities of IRC protocol, because it uses secure encryption algorithm, but it must work together with a strong security policy to protect system, such as the user must keep his pass phrase and private key completely secure.

In reality, no encryption algorithm is one hundred percent secure. An encryption algorithm may be breakable, meaning that given enough time and data, an analyst could determine the algorithm. The best we hope for is to make it so expensive to decrypt that the effort just isn't worth the money or is so time consuming that the information is worthless by the time it is decrypted.

BIBLIOGRAPHY

- [1] Oikarinen, J., Reed, D. Internet Relay Chat Protocol. May 1993.
- [2] Kalt, C. Internet Relay Chat : Architecture. April 2000.
- [3] Kalt, C. Internet Relay Chat : Channel Management. April 2000.
- [4] Kalt, C. Internet Relay Chat : Client Protocol. April 2000.
- [5] Kalt, C. Internet Relay Chat : Server Protocol. April 2000.
- [6] Pfleeger, Charles, P. Security in computing. United States : Prentice Hall PTR Prentice-Hall, Inc., 1997.
- [7] P.C. Van Oorschot and M.J. Wiener, Parallel Collision Search with cryptanalytic Application, Journal of Cryptology, Vol. 12, Number 1, 1999.
- [8] <http://www.mirc.com/>
- [9] <http://www.mirc.co.uk/irc.html>
- [10] National Institute for standards and Technology homepage of AES. (<http://www.nist.gov/aes>)
- [11] <http://pajhome.org.uk/crypt/rsa/>
- [12] <http://web.mit.edu/kerberos/www/>
- [13] http://www.ciscoworldmagazine.com/webpapers/2001/04_guardent.shtml#c3
- [14] <http://www.garlic.com/~lynn/sectax.htm>
- [15] <http://www.adfa.oz.au/~lpb/seminars/cauugs98.html>
- [16] <http://www.cis.ohio-state.edu/cgi-bin/rfc/rfc1851.html>
- [17] <http://www.rsa.com>
- [18] <http://www.secureaction.com/chat/>
- [19] <http://csrc.nist.gov/cryptval/des.htm>
- [20] <http://www.tropsoft.com/strongenc/des3.htm>

- [21] <http://www.itl.nist.gov/fipspubs/fip180-1.htm>
- [22] <http://www.newircusers.com/>
- [23] http://www.ecse.rpi.edu/Homepages/shivkuma/teaching/sp99/i20_security/index.htm
- [24] <http://www.hut.fi/~jlohikos/IRC.html>



St. Gabriel's Library, An

