Securing Public Instant Messaging (IM) At Work*

Nigel Williams, Joanne Ly†
Centre for Advanced Internet Architectures. Technical Report 040726A
Swinburne University of Technology
Melbourne, Australia
{182329,182268}@swin.edu.au

***Abstract*–This report studies the security of Instant Messaging (IM) applications within the workplace and suggests a secure method of implementing such systems. There are many secure enterprise IM applications available in the market. However, insecure public IM applications are more widely used within the workplace. Reasons for this include a failure at management level to provide clear direction on acceptable IM usage and a lack of monetary funds. By developing an add-on encryption system that is able to function over public IM networks, companies that have yet to define an IM strategy or do not have the resources to deploy an enterprise IM solution can begin to make the transition to a more secure messaging environment.**

***Keywords*-Instant Messaging, MSNP, OSCAR, encryption**

Contents

TABLE OF FIGURES

TABLE OF TABLES

* This report was originally developed and submitted for 'HET550: Design & Development Project 1', Semester 1, 2004

† The authors are undergraduate students in Engineering at Swinburne University of Technology

I. Introduction

Instant Messaging (IM) began its life as a trivial use of Internet technology offering little more than email without delay. IM services such as I-seek-you (ICQ), Microsoft's MSN Messenger® and AOL Instant Messenger® (AIM) were popular with computer-savvy youths who saw the applications as a new way of exchanging late night gossip, similar to telephone communication but with the ability to have multiple "conversations" concurrently.

Like the telephone, IM offered real-time conversations. It was also impersonal like email, whereby an interaction with a stranger could be as easily initiated just as it could be terminated. However, IM applications started to evolve and provide practical functionalities such as file transfers, application sharing and multimedia interactions via audiovisual means. This evolution of IM allowed it to shed its trivial stigma and become increasingly adopted by the mainstream as an alternative means of communication.

A. Background

In 1998, according to research firm IDC, more than 63 million people had IM accounts. By the end of 2002, the figure nearly tripled to 174 million accounts [1]. The increased use of IM was a catalyst for the natural progression of IM usage at home to IM usage in the workplace. At first, IM usage in the workplace was frowned upon and was seen as a "virtual water cooler" [2]. It was seen as a time-wasting tool used to send personal, non-work related messages and a drain on network resources.

This attitude changed as people started to see beneficial work related usages for IM applications such as being able to quickly arrange a meeting without having to make several phone calls or having to wait for all parties to reply to a meeting request email.

Such functionalities have seen IM programs become increasingly popular within the workplace for rapid, convenient business communications between employees in different physical locations. In 2003, IDC research found that there were over 43 million users of public IM in the workplace [3]. With the fast paced nature of businesses today and the growing need for instantaneous information exchange between employees, IM is no longer seen as a trivial application and instead has become a widespread productivity too.

However, IM usage does not come without its disadvantages. These come in the form of a number of operational issues including a severe security threat, as well as various sociopolitical implications that the organisation would need to deal with on a management level.

Examples of operational issues posed by IM usage within an organisation include:

- Most public IM services used by workers do not include encryption services, so there is a possibility of industrial espionage through listening in on conversations that are exchanging corporate sensitive information.
- Public IM servers, unlike the organisation's email servers, operate beyond the control of network administrators.
- Implementation of a more secure enterprise IM service internal to the organisation may create unwanted overheads in regards to resources and personnel.

Examples of sociopolitical implications that the organisation would need to deal with include:

- The Human Resources (HR) department and senior managers having to implement an "acceptable usage" policy to ensure that misuse of IM is eradicated or at least minimised.
- IM being seen as a replacement for emails, removing the formality of communications, which can be seen as unprofessional when communicating with clients.

B. Current Solutions

The need for secure IM applications for workplace usage has created a new market for enterprise IM solutions. These solutions offer the following features to provide security:

- Allowing organisations to be able to control the access to IM applications
- Audit and archive IM conversations
- Anti-virus checks, content filtering and anti-Spam protection
- The ability to lockout unauthorised IM and peer-to-peer file sharing connections
- Encryption
- Lock out unauthorized IM and peer-to-peer file-sharing connections

However, according to the results of a survey conducted by Osterman Research Inc. in March 2003, these secure enterprise IM servers (Lotus Sametime, Microsoft Windows Messenger and Microsoft Exchange IM) are not as extensively used as public IM within the workplace [4].

These findings were reflected in a Radicati Group study conducted in 2003. It found that nearly 80 percent of instant messaging in workplaces was done using public IM services [3]. The reasons behind the slow uptake of enterprise IM in comparison to public IM were a focal point in the project research.

C. Problem Statement

The project aims to provide a secure IM solution for organisations currently using public IM.

D. Project Purpose and Objectives

The objectives of this project are:

- To research and understand the technical concepts of IM technology in the workplace
- To research and understand the sociopolitical issues and implications associated with IM technology in the workplace
- To build a secure enterprise IM platform prototype based on the research findings

Upon commencing the initial research, it became apparent that the final objective should be modified to better address the attitudes of organisations towards enterprise IM solutions. The final objective was then:

- To build an encryption add-on for public IM clients to help minimise security risks whilst public IM remains the most widely used corporate IM solution.

E. Project Motivation

The Telematica Instituut in the Netherlands conducted a study in 2004 to see how public IM is adopted in the workplace [8]. The study investigated the use of IM from four months before to three months after IM was formally introduced in the organisation.

The study found that the use of IM increased fourfold after the formal introduction of IM, in terms of both users and conversations. IM users stated that they found IM improves the way they reach others and the way others reach them.

While this study showed the rapid adoption of public IM within the workplace and the benefits gained by employees, it did not take into consideration the security risks.

The project motivation was to identify security issues with the use of public IM at work and propose a solution that balances the security issues with the benefits of IM.

F. Outcomes

The first phase of the project produced the following outcomes:

- The supposition that many organisation's sociopolitical climate deems available enterprise IM solutions to be unsuitable for their current IM/communication needs
- The conclusion that providing a secure solution for public IM usage at work would be more relevant than providing another enterprise IM application
- The development of a simple prototype to prove the concept of being able to implement an encryption add-on to public IMs to help provide information security

II. Research Method

The primary sources of information used were purchased and loaned textbooks, the Internet and from practical testing and analysis. The proprietary nature of many instant messaging protocols and the general lack of documentation meant that the Internet and practical analysis were of particular importance. Practical analysis of protocol data also allowed a 'hands on' approach to understanding how the protocols functioned.

A. Software Tools

As much of the information available regarding public IM architectures is unofficial, it was necessary to carry out practical testing in order to verify the accuracy of the documentation. It was also necessary to make observations to account for any protocol changes that may have occurred since the publication of the documents. A number of techniques were employed to perform the tests, although the bulk of information was obtained using the network protocol analysis software *Ethereal*.

1. Ethereal and Protocol Analysis

Ethereal is a free graphical network protocol analysis tool available on a wide range of operating systems. It is able to actively collect network traffic from approximately 500 protocols off of a network interface and save this data to a file for later examination. It is also possible to display traffic information in real-time.

Ethereal was chosen due to its ability to automatically identify and categorise the Instant Messenger protocols being tested (MSN, OSCAR). The filter functions also allow protocol conversations to be isolated in sequence, so that specific information exchanges such as client requests and server responses can be tracked easily. The relative simplicity of the Ethereal software package was also a factor in the software selection process.

Ethereal was used to capture data exchanges between the group members when connected to various public IM services. IM protocol data was then isolated and examined.

2. Other Tools

A small number of network tools were also used when Ethereal was not required. These tools were *nslookup*, *netstat,* and *Kerio Personal Firewall 3*.

- **Nslookup**: Allows command line and interactive querying of a DNS server to resolve the IP of a chosen domain. This tool was used to quickly resolve the IP address of messaging servers.
- **Nestat**: Displays the current connections to a network interface organised by protocol. This was used to examine networks connections while performing various tasks (such as file transfer) on the IM networks.
- **Kerio Personal Firewall**: A simple Windows-based firewall application. This was used to observe the firewall subversion methods of the IM software.

B. Development Tools

The prototype was written in the Java programming language using the Crimson Editor and Mi text editors. It was compiled using the Java 2 SDK Version 1.4.2_03.

C. Research Variables

The major research variables listed affect the manner in which IM is implemented in the workplace and the possible extent to which it is exposing itself to security threats. These variables are important in determining what an ideal solution is for using IM at work.

1. IM System Attributes

These attributes define each IM system and differentiate it from other solutions.

a) IM functionalities

These are the features of an IM platform. This report examined the functionalities as defined in RFC2778 "A model for Presence and Instant Messaging" and RFC2779 "Instant Messaging/Presence Protocol" (RFC2778 and RFC2779).

b) IM Network Components

An IM network consists of servers and clients. How these components are implemented and interact will help shape how the IM system operates.

c) IM Communication Models

The methods used by IM network components to exchange information with each other.

d) Server Configuration

How servers, the backbone of an IM network, function in the network hierarchy. This includes the number of servers and the role that these servers perform.

2. Types of Security Threats

Security threats can come from inside and outside the organisation. Therefore, both internal and external security needs to be considered.

3. Employee and Senior Management Attitudes

The attitude of both the organisation's employees and senior management can affect how IM is implemented and used in the workplace. These attitudes need to be investigated to determine what the most suitable solution to the problem is.

a) Employee Attitude

If an employee bears a grudge against the organisation, they might decide to steal some sensitive data that they know

is currently being exchanged via IM by colleagues. Or if an employee bears a grudge against a rival organisation, they can use IM to gain sensitive data from the rival thus leaving the organisation in legal strife and possibly damage its reputation.

Also, if there is no user policy, an employee may take it for granted and use IM in an unacceptable manner. This may affect their own and possibly other employees' productivity.

b) Senior Management

Manager attitudes also have an impact on how IM is used. The ability of Managers to accept that there is use of IM in their network is vital. By not recognising IM use, they may be exposing the organisation to security risks and decreased employee productivity. If Managers have recognised IM use, then they must be proactive about minimising any detrimental affects (such as creating an acceptable usage policy) that IM usage can cause.

If managers decide to take the authoritarian approach to IM and ban usage altogether, it may be met with resentment by employees and create a culture of furtive IM usage at work.

III. Literature Review

The section describes the theory and concepts that were researched in order to comprehend the problem and devise an effective solution.

A. Instant Messaging Overview

This section will cover what defines IM including its functionalities and network configurations.

1. Terminology

MSNP	MSN Messenger Protocol
OSCAR	Open System for Communication in Real-time. Protocol used by ICQ and AIM.
Principle	A user within an IM network.
Public IM	Also called free IM. Programs that are downloaded and used without cost over public networks (e.g. AIM)

Table 1: IM Terminology

2. IM Functionalities

A set of protocols that defines what constitutes an Instant Messaging system did not come about until 2000. The release of RFC2778 "A model for Presence and Instant Messaging" and RFC2779 "Instant Messaging/Presence Protocol" attempted to provide a set of guidelines for the basic requirements of an IM system. The core functionalities described in the following sections are based on the recommendations of RFC2778 and RFC2779. The following section describes the most basic functionalities of an IM network. A more detailed description of these functionalities is located in RFC2778 and RFC2779.

a) Presence Service

The Presence Service disseminates Presence Information across the IM network. Presence Information is defined as the address at which a principle can be contacted and the current status of the user. It also provides privacy control, which allows a principle to select who can see their status and/or message them.

b) Messaging Service

Facilitates and delivers instant, real-time text messages between principles on the IM network.

c) Additional IM Features

The previous sections describe the most basic implementation of an IM system. Most modern IM systems, including the public systems studied in this report, include additional features in order to differentiate themselves. These additional features are listed in RFC2778 and RFC2779.

3. IM Network Components

a) IM Servers

IM servers provide IM services such as presence notification, account registration and instant messaging. A client must connect to the server in order to be given an online presence status on the IM network

An IM network may have multiple servers working in tandem to provide scalability for a large number of principles. Transfer between servers is generally transparent, meaning principles have no knowledge as to which server they may be connected to. The servers of the public IM networks are located on the premises of the organisation providing the service. For example, the MSN servers are located in an area chosen by Microsoft.

b) IM Clients

The client software is the public face of an IM network. Once installed on a workstation, the client software can be used to connect to the IM server to access IM services. Clients are also able to store specific user information locally. This information includes passwords and account names (for automated login) and records of chat sessions.

Clients for the public IM services are available as a free download on the Internet.

4. IM Communication Models

All IM networks are inherently client-sever based as the client software is used to access the IM network. While this is true for login, presence information and user location, other services may not be implemented using this model. There are two basic communications models that can be used on an IM network, Client-Server and Peer-to-Peer. [5] [30]

a) Client-Server

All information on the network will pass through the server in transit to its destination. In the instance of the client-server model being used for instant messaging services, all messages sent would travel via the server. Two clients will not directly communicate or know the location of the other. This is a centralized form of communication (Figure 1: Client Server Model).

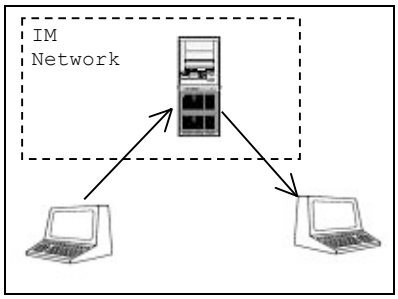


Figure 1: Client Server Model

b) Peer-to-Peer

Data sent directly between clients without passing through a central server is referred to as peer-to-peer communications. This method of communications is generally used for file transfers, to reduce load on the IM network servers. In a network that allows peer-to-peer messaging, the central server is used to track clients and provide the necessary information for clients to communicate directly. This is a decentralized form of communication (Figure 2: Peer-to-Peer Model)

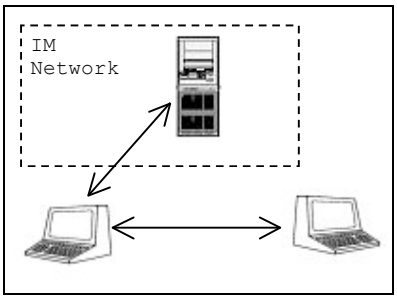


Figure 2: Peer-to-Peer Model

5. Sever Configurations

As noted in section III.A.3.a), an IM system may make use of more than one server, for reasons of network scalability.

a) Single Server Architecture

All IM tasks can be conducted on a single server; including account creation, authentication and IM services. Single server architecture allows for easier management and increased security but is not a scalable solution and may struggle with a large load of users.

b) Multiple Server Architecture

Most large IM networks operate using multiple servers. The role of servers in a multiple server architecture can be divided into two groups, replicated servers and distributed services.

(1) Replicated Servers

Individual servers capable of providing all IM functionalities can be replicated and interconnected to support a large network (Figure 3: Replicated Server Architecture). Principles are registered to a 'home' server that is able to contact other servers allowing communication with principles belonging to another 'home' server. This approach is used in the Jabber architecture. It is essentially a networked version of the single-server approach.

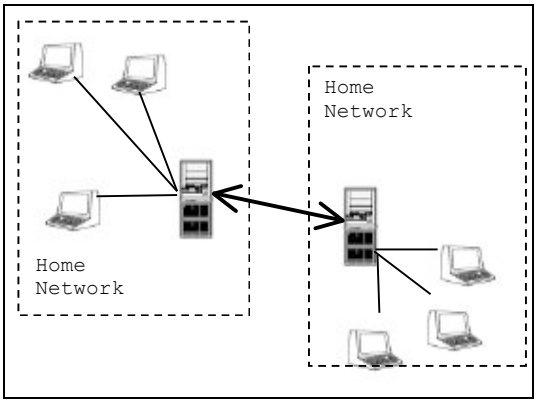


Figure 3: Replicated Server Architecture

(2) Distributed Services

Servers may be divided based on the role that they play within the IM network. One server type may fulfil the function of account registration or login, while another server type may provide notification and messaging. It is possible to have multiples of each server type. Maximum scalability is achievable by dividing servers based on service and then providing multiples of each server type. This method of server distribution is complex and requires a large infrastructure.

For example, the MSNP architecture provides a pool of authentication servers located at the domain messenger.hotmail.com. Upon accessing the domain, the client is directed to one of a number of authentication servers, which then redirects the client to one of a number of notification servers. This way, a server can be assigned based on a number of different metrics such as geographical location or server loads.

6. Account Creation

IM applications also vary in the way user accounts are created. The methods the can be used are:

- Dedicated port
- Dedicated server
- Web-based
- Integrated LDAP
- Pre-issued accounts

For a detailed discussion on account creation, refer to IM Account Creation.

7. Public IM Applications

According to an Osterman study [4], the two most used IM applications at work (including unofficial usage) were AIM and MSN whilst ICQ came fifth in the list.

AIM and ICQ are owned by AOL Time Warner and use the Open System for Communications in Real-time (OSCAR) protocol, while MSN is managed by Microsoft Corporation and uses the MSN Protocol (MSNP).

a) Protocol Description

This description focuses on specific security risks created by using the OSCAR and MSNP based services. These protocols were chosen as they together represent a large number of corporate IM users [4]. They also have strong third party protocol analysis and client development that provided a great deal of protocol information. It is a reasonable assumption that these protocols would be representative of the security issues posed by a number of different IM networks.

This summarised analysis describes the core functions of the public IM systems, which are login and presence notification, text messaging and file transfer. Detailed descriptions of each process can be found in Appendix C and Appendix D.

b) MSN Protocol (MSNP)

Servers and clients within the MSN network communicate using a series of UTF-8 encoded commands over TCP/IP connections. These commands are their parameters are always transmitted as plaintext (i.e. unencrypted). The MSNP predominantly uses a client-server based architecture.

MSN accounts are based on Microsoft's .NET Passport scheme. Users attempting to access the MSN Messenger network must first contact the Dispatch Server to request login, which then redirects the user to the .NET Passport Nexus server for authentication.

Communications with the dispatch server occur in plaintext and contain information such as the account name of the user and their operating system and hardware platform. Communications with the Nexus server occur over an encrypted Secure Sockets Layer (SSL) HTTPS connection. The SSL connection prevents the password from being transmitted in plain text. If authentication is successful, the client may connect to the Notification Server.

In the event that the default port of the MSN client is blocked by a firewall, MSNP has the ability to function over port 80 using a special HTTP wrapper. This wrapper communicates with a special MSN HTTP server, with all protocol traffic structured as HTTP data. This allows MSN to operate through a Web Proxy.

At certain times throughout a connection to the MSN presence service, the client will synchronise user data with the server. This includes the handles and account names of all contacts, as well as any phone number information the contacts have. The client will also synchronise the account details of the principle upon connection to the network, including information such as phone numbers, addresses, date of birth and location. The synchronisation process occurs in plaintext.

Text messaging occurs in plaintext using a client-server model. Messages are sent through a connection established with the MSNP messaging server, the Switchboard. At no stage during a messaging session do principles know of the IP address or any other information of the other. Multi-party chat sessions are also supported, although there is no ability to send a private message during a conversation.

File transfers are facilitated by the Switchboard Server, but are carried out peer-to-peer using the MSN File Transfer Protocol (MSNFTP). If two users have agreed to transfer a file, the Notification server will provide IP addresses. Generally the IP address of any network interface is also supplied, so in situations where two users are located on the same network, the network interface can be used for file transfer. MSNFTP does not alter the file in any way, transmitting the file directly as blocks of binary data.

c) OSCAR Protocol

OSCAR uses a binary/hexadecimal coded command syntax that is structured using a series of "frames". OSCAR frames are therefore not easily read, as commands do not translate directly into meaningful text. Frame sequences can be deciphered, however, if access to a command syntax key is available.

OSCAR provides two methods for user authentication, referred to as "Channel 0x01" and "MD5-based" authentication. Both methods transmit a portion of information in plaintext. This includes client version, account name and the country in which the user is located. The two methods differ in the way in which passwords are encrypted. Channel 0x01 authentication encrypts a password using a method called "roasting". This method is not secure and an intercepted password can be easily decoded. The method for decoding roasted passwords is detailed in section IX.D.5.a), Protocol Analysis – OSCAR. MD5-based login uses an MD5 authentication key acquired from the login server. This key is used to encrypt the password before it is transmitted to the server. The MD5 authentication key is changed at each login and a password cannot be easily recovered as in the "roasting" method.

OSCAR has been designed to operate on a wide range of ports. If the default port of an OSCAR client is blocked by a firewall, the clients are able to use any number of ports to tunnel information. For example, the "auto configuration" utility of the AIM client actively probes the network for open ports. Observation of this utility showed the ability of OSCAR to function over a number of ports, including port 80 (HTTP), 53 (DNS) and port 25 (SMTP). The OSCAR clients

are also able to function over a web proxy if auto configuration fails.

The ICQ and AIM clients handle user data in different ways; therefore OSCAR has two methods for data synchronisation. The ICQ network stores a large amount of user information, such as home and work details and hobbies/interests. This information is stored on the ICQ BOS and client, so must be synchronised at login. Information about contacts is synchronised at the request of the client.

The AIM client only contains limited information about the principle, so synchronisation only occurs in the direction of the server when a user changes their information. For example, if a user changed their screen name, the server would be notified of this change. All data other than passwords are sent as plaintext.

Text messaging is also possible using two methods, client-server based and peer-to-peer. AIM's implementation of text messaging is client-server based bye default, although peer-to-peer messaging can be selected in configuration. ICQ will send messages peer-to-peer if a direct connection is available between users. ICQ is also able to send 'offline' messages, which are text messages sent to a recipient while they are offline. Messages are stored on the ICQ servers until that user's status becomes online. The OSCAR protocol sends messages as plaintext.

d) Protocol Analysis

By combining the protocol descriptions of MSNP and OSCAR with the research variable, a picture of the security problems with the networks emerged.

- **Lack of Encryption:**

The major concern identified from the protocol analysis is the transmission of much information in plaintext over insecure networks. The IM servers are publicly accessible by their very nature, so sensitive data such as account names or phone numbers are being transferred across any number of networks to reach the IM servers. Text messaging also occurs in plaintext, so were any data interceptions to occur, messages would be easily read. If corporate data was exchanged using instant messages, an eavesdropper could potentially obtain a great deal of sensitive information. This information may then be used in other forms of attacks, such as account hijacking or message spoofing. Poor password encryption is also a concern, and the OSCAR Channel 0x01 authentication method is a particularly unacceptable situation that might lead to account hijacking.

- **Information Exposure**

The client-server of model over the Internet heightens the risk of data interception. For example, a message sent from one employee to another employee elsewhere in the office must travel via the IM servers located outside the company network. This needlessly exposes information to outside networks (Figure 4: Messages must pass through the Internet to Reach Destination).

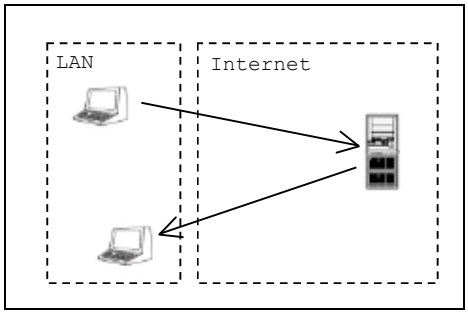


Figure 4: Messages must pass through the Internet to Reach Destination

Peer-to-peer messaging is somewhat more secure than client-server, in that text messages do not leave the network. The server must still supply IP addresses however; so internal addressing is exposed to external networks. The risk of receiving a spoofed instant message increases if an IP address is exposed.

- **Firewall Subversion and Malware**

The public IM protocols have been designed to run over a variety of network infrastructures. The ability to work around network security measures is therefore heavily integrated into the clients, as seen by the use of essential service ports such as HTTP and DNS. Having IM clients tunnelling through a corporate firewall is a serious breach of network security, as an attacker could exploit the opening to gain access to the network.

As described in section 5.2.2, IM networks provide a mechanism for the distribution of varying forms of malware. An Internet worm or virus that uses IM to propagate will spread into a corporate network if public IM clients allow access through the firewall.

The combination of firewall avoidance and the use of IM networks as a malware delivery system is extremely dangerous. The case study "Buddylinks and the AIM network" located in Case Study - Buddlinks and the AIM network highlights the potential damage that could be caused by such a combination.

B. General Security Problems with Public IM

The previous section outlined security problems with IMs using the MSNP or OSCAR protocols. This section details a more generalised list of security problems an organisation can face when public IM applications are running within its network.

1. Network Exposure

IM clients will often attempt to circumvent firewalls on a network. Allowing IM information into a network that is unauthorized may risk compromising network security. An attacker may exploit an IM client running behind a firewall and then gain access to the internal network using the IM clients' own firewall bypassing techniques. Public IM networks can also be potentially as a large distribution network for numerous forms of *malware*, as described in section 2.

2. Malware

Viruses, Trojan Horses and Worms are forms of malware that can be spread using IM networks. Many IM clients contain scripting elements that may be exploited by Trojan horse programs or worms, much like the Macro and VB Script viruses that are seen in many emails. A Trojan horse installed onto a client within a corporate network may compromise network security, as information may be tunnelled in and out of the network in the guise of IM traffic. The firewall evading nature of IM clients and the threat of various forms of malware exploits form the greatest risk to corporate network security.

3. Data Interception

Public IM clients do not use any form of encryption and messages must pass through several unknown networks to reach the IM server. Instant messages are therefore vulnerable to eavesdropping as there is no control over the communications channel nor is data protected in any way. Information recovered through data interception may include phone numbers or sensitive organisation information details. This information can be used for purposes of industrial espionage, or in other IM attacks such as *account hijacking*, *spoofing*, or *data manipulation*, described in the following sections.

In addition, IM clients allow a user to save conversations to a file, generally in plain text (MSN saves in XML formatted UTF-8, for example). If an attacker is able to gain physical access to a workstation, saved conversations are easily retrieved.

Techniques of data interception are described in section C.

4. Message Spoofing

If an attacker has access to appropriate information and tools, fraudulent instant messages can be constructed that appear to be from a legitimate source. For instance, an attacker may insert an instant message into an existing chat session and then capture the response. The attacker does not need access to the account of a user to spoof an instant message.

5. Account Hijacking

This is the process where an attacker gains sufficient information to log into a victim's IM account. The attacker may then impersonate that user in an attempt to gather information. Password information that is stored using a weak encryption algorithm or in plain text can increase the threat of an account being hijacked.

6. Man-in-the-Middle

If an attacker is able to place themselves between the server and client, messages may be processed or altered in transit. An attacker may also prevent the delivery of instant messages. This is shown in Figure 5: Man in the Middle attack.

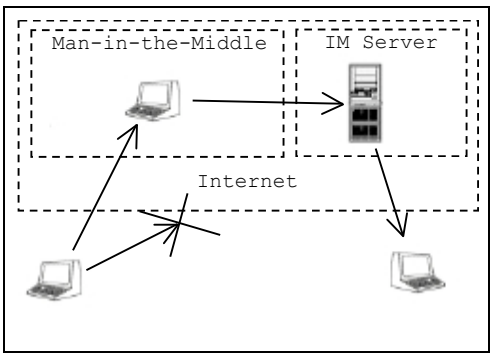


Figure 5: Man in the Middle attack

C. Methods of IM data interception

Data interception on an IM network can occur mainly through the practice of Packet Sniffing (see Packet Sniffing). For data interception to occur, an attacker must have access to the network segment(s) on which the data is travelling. The methods used to gain access to network data thus changes depending on the type of network that the victim is using to access the IM service.

1. Shared Media Networks

A shared network medium allows easy interception of data. As all data traffic is carried on a common channel, any node connected to the network may intercept traffic.

a) Shared Ethernet Networks

Ethernet networks that propagate information using hubs or that use a bus topology are considered to have a shared medium. This is due to information transmitted from any node on the network segment being visible to other nodes on the network. As all data is visible to all nodes, any node on the network is able to 'sniff' data destined for another node, thus making data interception simple. Such networks are becoming increasingly rare, as switching technology is now standard for wired networks.

Software tools available that will capture data from a shared media network are numerous. Network analysis tools such as *tcpdump* and *Ethereal* can be used, although several programs are available that provide easier real-time access to specific IM protocol information. Prominent programs are *MSN Sniffer* and *Aim Sniff*. These programs are easy to use and do not require knowledge of TCP/IP networking, providing an accessible eavesdropping tool.

b) Wireless Networks

Wireless networks share a common medium, the radio frequency (RF) spectrum. Although data from various transmitting nodes can be placed onto separate RF channels to prevent collisions, the medium is still considered shared as there is no way to prevent any receiver within range from tuning into any channel. Depending on the security level of the network, access to all data may be possible by simply being connected to the network. For example, a node connected to a public wireless access point might be able to capture IM data from any other nodes connected to the network. Poorly configured home LANs can also be easily broken into, after which all data broadcast can be observed

Several packet capture programs are available for data interception on wireless networks, two popular tools being *AirSnort* and *Kismet*. These programs are not specifically geared towards eavesdropping and could be considered the wireless equivalent to programs such as *tcpdump*.

Wireless networks that use higher-level security such as a IPsec and Virtual Private Networking are more difficult to obtain data from. It is still possible to intercept packets, but the contents may be encrypted, therefore requiring the additional (and often difficult) task of decryption.

2. Switched Networks

The majority of wired networks are now based on switching. In a switched network, a host will only have access to information that has been specifically addressed to it. There is no sharing of the network medium. This isolation of data increases performance, while it also prevents one host from accessing data destined for another host.

It is therefore difficult to intercept data on a switched network, simply by design. There are two approaches that can be taken when packet sniffing switched networks is required.

a) Software

Software tools are available that will 'spoof' a network switch, enabling packet capture. *Ettercap* is one such utility that is able to 'poison' the ARP cache of some switches to re-route data from specific ports through the port of the attacker. This is generally referred to as a 'man-in-the-middle' attack. *Arpspoof* is another tool that is able to sniff a switched network by sending false ARP information to a switch, allowing data to be 'mirrored' to the attacker's node. This type of technique can be used from a lower point in the network topology, although the attacker must be connected to the same switch as the victim.

b) Point of access

Data from several lower regions in a network topology must be concentrated at various points higher in the topology, such as network backbones or Internet gateways (Figure 6: Interception at a Gateway). If a packet sniffer is able to gain physical access to hardware high in the network topology, data interception is greatly simplified. A packet sniffer placed on the same segment as an Internet gateway would be able to gain access to all public IM traffic. The configuration of a switch may also be changed so that a copy of all data is re-transmitted out a specific port that the sniffer is located on. Once an appropriate position is found for the node, standard networking tools such as *tcpdump* can be used to capture data.

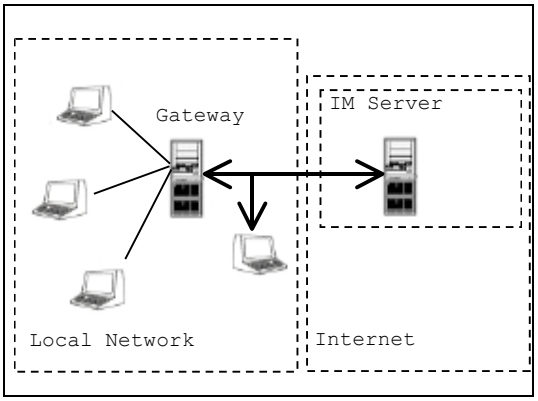


Figure 6: Interception at a Gateway

These approaches require administrative access to the network. However, it is never safe to assume that someone will not abuse their privilege if correctly motivated.

c) Data Interception and Modification

Packet sniffing can generally be described as a passive method of data interception. Section 5.3.2.1 identifies the software package *Ettercap*, which creates a Man-in-the-Middle situation. This situation can be used to actively alter or filter data as it passes through the attacker's node.

3. Other Methods

Data can also be intercepted by means of the installation of software such as keystroke loggers. While keystroke loggers are not strictly related to IM, it is certainly possible that an IM Trojan could be developed that will log conversations and transmit them to the attacker.

D. Encryption Overview

A way of keeping sensitive data hidden from eavesdroppers is to encrypt the data being sent in such a way

that only the receiver is able to decrypt it. The technology used in this coded communication exchange is known as cryptology.

Cryptology comprises of two main parts: communications security (COMSEC) and communications intelligence (COMINT). Cryptography is the study and practice of COMSEC. Cryptography involves disguising the original message (know as plaintext) into a covert form (known as ciphertext) through encryption. The encryption is done using a mathematical process known as a cryptographic algorithm, whose operations are controlled by a key. The person who receives the ciphertext needs to know the algorithm and the key to decrypt the message from ciphertext to plaintext.

COMINT involves obtaining secret communications (without the permission of the communicators) through methods such as eavesdropping, bugging rooms and tapping telephone conversations. If the information is encrypted, attempts to crack the cipher will be made through the use of cryptanalysis.

The project will primarily focus on COMSEC, in particular cryptography as a means to achieve the project objectives.

The generic scenario of information communication is shown in the figure below.

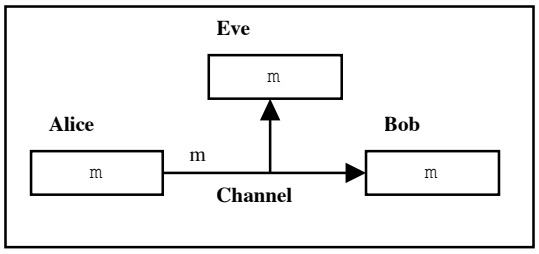


Figure 7: Generic Information Communication Exchange

Alice and Bob want to communicate with each other, over an insecure communication channel. Eve is eavesdropping on the communication exchange. Everything that Alice sends to Bob (in this case Alice is sending message *m*), Eve also receives and vice versa. In addition, the information exchanged can be clearly read by Eve. The aim of COMSEC is to allow Alice and Bob to communicate without Eve being able to read the information exchanged.

1. Encryption Concepts

If the data exchanged between Alice and Bob contains sensitive information, it is possible to keep the information confidential through the use of encryption.

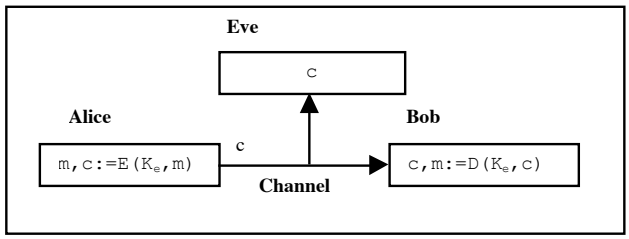


Figure 8: Encrypted Communication Scenario

The process of communication shown in diagram above is:

1. Alice will create a plaintext message *m* to be sent over the communications channel
2. The encryption mechanism will take *m* and perform a mathematical operation using encryption algorithm *E* on *m* and the key K_e to convert *m* into ciphertext *c*
3. The ciphertext is sent over the communications channel
4. Eve can pick up *c* anywhere along the communications channel and if the encryption algorithm is successful, be unable to decrypt the ciphertext
5. *c* will arrive at the destination and be decrypted using a decryption algorithm *D* on *c,* using the key K_e to reconstruct the message *m* so that Bob can read the message

Basically, encryption involves transforming plaintext into ciphertext via mathematical algorithms. What defines the operations of the algorithm is called a key. The key details how the algorithm will encrypt the information on the sender's end and is used to decrypt the information on the receiver's end. The main advantage for having a key is if an enemy is able to discover the details of the key used, only the key needs to be changed – not the whole algorithm. As a sound security practice, the key should be changed on a regular basis as to minimise the chance of having sensitive information exposed.

If a third party is able to systematically reconstruct the plaintext from the ciphertext without knowing the key pair within an appropriate timeframe, then the encryption scheme is said to be *breakable*. The timeframe is normally the lifespan of the data being protected. If the timeframe has expired, essentially, there is not a need to keep the information encrypted and so if anyone is able to break the code then it will not be of any concern.

2. Information Security

Information security aims to handle and minimise data communication problems. These problems include sensitive data being obtained and possibly altered, passing on sensitive data to an adversary who is posing as a trusted entity and disputes about the content of a past communication exchange.

The main objectives of information security are: confidentiality (or privacy), data integrity, authentication and non-repudiation.

Confidentiality is concerned with preventing unauthorised entities from being able to read the information that they have intercepted, achieved through encryption. Authorised entities will be able to read the information by knowing what encryption was used and reversing it through decryption.

Data integrity involves checking for any alterations of the data (information) received. These alterations include deleting and modifying all or part of the data as well as inserting additional data. It is not possible to completely prevent data alternation. However, it is possible to determine with high probability whether or not the data has been tampered with.

There are several cryptographic devices that are used to ensure data integrity including message authentication codes, digital signatures and keyed hash values. These devices are discussed in Data Integrity and Authentication.

The main function of *authentication* is to establish the identity of the user or system that originated the information. Authentication helps verify to the receiver whether the originator is an authorised user of the system or whether they are sanctioned to view sensitive information.

Cryptographic devices that are used for authentication include digital signatures and message authentication codes as well as key agreement procedures. A more in-depth discussion is found in Data Integrity and Authentication.

Non-repudiation involves proving the integrity and origin of the information independently by a third party. Disputes can occur between two parties about "who said what". For example, one party may have agreed to complete a task by a certain date but later deny ever doing so. Non-repudiation helps resolve these disputes.

The aim of non-repudiation is to prevent any denial in involvement in a previous action. This is done through the usage of a digital signature calculated by a private key known only to the entity that generated the digital signature. The signature binds the information to an entity, thereby making any denials from the entity implausible.

a) Combining Objectives

Most data security systems combine two or more of these objectives to gain a satisfactory level of security. Sometimes these objectives involve implementing physical measures, such as using biometric information to authorise users, as well as cryptographic mechanisms.

Some cryptographic algorithms are able to serve multiple objectives. For example:

- Digital signature algorithms can provide authentication, data integrity an non-repudiation
- Message authentication codes can provide authentication and data integrity if symmetric keys are unique to each pair of users
- Certain types of encryptions can provide confidentiality, data integrity and authentication

To be able to meet all of the information security objectives, there needs to be a combination of cryptographic (and possibly physical) mechanisms implemented.

E. Enterprise IM Applications

The growing use of IM as a business tool has been recognised by a number of companies who now provide products that attempt to address the security issues of public IM networks. There are two basic approaches taken by the companies producing Corporate IM solutions, Standalone IM and Proxy IM [6].

1. Proxy Solutions

Proxy IM solutions act as a gateway to public IM networks and attempt to layer some security or auditing over the connections.

a) AIM Enterprise Gateway 2.0

Designed solely for use with OSCAR based AIM and ICQ Lite clients. This gateway server creates a searchable log of instant message traffic and provides some data security features. Messages destined for internal users are redirected and do not use the AIM infrastructure. Messages can be optionally encrypted. This service may operate without internal management, relying on the public AIM infrastructure, or can be managed internally using AOL's Federated Authentication scheme, at extra cost.

b) Akonix L7 Enterprise

An L7 server is placed inside the company network and acts as a gateway to public IM infrastructure. All IM traffic must pass through the server, which logs messages and generates reports based on a number of factors, including key words used in text messages. The filtering functions of L7 allow specific IM and file-sharing traffic to be blocked. There is no encryption of data transmitted over the IM networks. The server is compatible with AIM, ICQ, MSN and Yahoo! Messenger and is managed internally using the Microsoft management Console (MMC).

c) IMlogic IM Manager

Acts as a proxy server for all the popular public IM systems and will also proxy other corporate IM solutions. The main use of IM Manager is logging and filtering of IM traffic. Security measures include built-in virus scanning and the ability to detect and block spamming activity. Management offers the ability to use AOL's Federated Authentication, while MSN account names can be managed locally through IMlogic's agreement with Microsoft.

d) Yahoo! Business Messenger

J2EE servlets are used within the corporate network and allow the operation of the Yahoo! Enterprise client. Presence monitoring and Message delivery are performed outside of the company network by Yahoo!. Messages between enterprise clients are encrypted with SSL, while messages between the enterprise client and public client occur in clear text. Yahoo! manages the IM network and provides a guarantee of service uptime.

2. Standalone Solutions

Standalone IM solutions involve the creation of an entire internal messaging and presence system.

a) Bantu IM & Presence Platform

A Java/Web Interface based solution that can be hosted internally or by Bantu. The Bantu platform is able to log message data, but does not generate reports based on keyword filters. End point connections are secured using SSL. The Bantu platform is able to communicate with existing public IM networks, although messages are not encrypted. The network can hosted and managed internally, or can be hosted and managed by Bantu.

b) Lotus Instant Messaging and Web Conferencing (Lotus Sametime)

A standalone system that is able to natively interface with AOL networks and can interface with other networks at extra cost. Basic editions of Lotus provide logging of messages but will not create reports. Messages between clients are encrypted. Clients are available as Windows native applications or as platform independent Java applications. The system is managed locally on a Lotus Domino server while accounts can also be authenticated with an LDAP server.

c) Jabber Extensible Communications Platform (XCP)

Internal solution based on the open source Jabber Extensible Messaging and Presence Protocol (XMPP), with the ability to connect to other Jabber networks. Clients can be those supplied by Jabber Inc or any third party client that supports XMPP. SSL can be optionally used between servers and clients. Extended IM functions are provided by integration with Microsoft NetMeeting. XCP servers operate and deploy in a manner similar to Email servers, so are managed internally.

d) Microsoft Office Live Communications Server 2003 (LCS)

Complete internal messaging system that replaces Microsoft's Exchange IM. Has the ability to log message traffic and offer simple reports. Communications are encrypted using TLS and authentication occurs with NTLM and Kerberos. The client software is called Windows Messenger, which is a business oriented version of MSN Messenger. LCS does not communicate with public IM networks, although may communicate with the MSN network with the purchase of Messenger Connect for Enterprise. The network is managed using MMC and the Active Directory.

F. Public IM vs Enterprise IM

An Osterman study in 2003 asked respondents "Which of the following IM products are in use in your organisation, even unofficially?"[4]

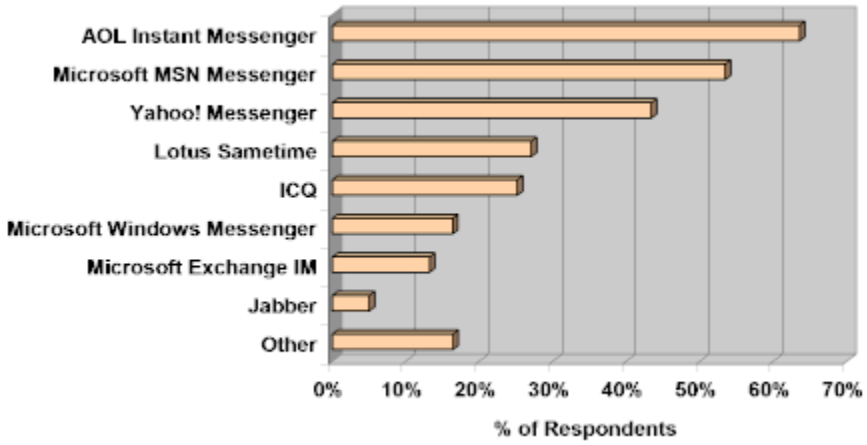


Figure 9: IM Products used within Organisations

From the figure above (Figure 9: IM Products used within Organisations), it can be seen that the most used three IMs were public ones (AIM, MSN and Yahoo!). The highest ranked enterprise IM was Lotus Sametime, which was ranked fourth in the list. The survey also showed that Lotus was used less than half as much as AIM or MSN.

In 2003, there were over 43 million users of IM in the workplace [3]. In the same year, it was found that about 80 percent of IM usage at work was done using public IM services such as AOL, MSN and Yahoo! Messenger [3].

Hence it seems that although there are enterprise IMs in the market that can make IM more secure at work, they are not being used as much as the insecure public IMs.

G. Sociopolitical Issues with Workplace IM Usage

IM does not suffer solely from technical issues. As with any workplace tool, especially one used in communications, there needs to be a usage protocol to ensure that the organisation does not suffer from any negative consequences from utilising IM.

1. Management Attitude towards IM

Osterman Research surveyed 150 companies in 2003 asking them what they felt was most important in the context of enterprise IM [9]. High on the list was proving value of IM usage to senior management.

Supporting these results, Michael Gartenberg of Jupitermedia stated: "If I were to poll ten CIOs and ask if instant messaging was being used strategically for business communication within their organisations, nine out of the ten would probably say no and eight out of nine would probably be wrong." [1]

With this attitude, it is no surprise that only a quarter of companies surveyed in 2003 had active acceptable IM usage policies implemented [3]. Without these policies, keeping the organisation's network secure would be near impossible as there would be employees who will use IM improperly and hinder any technical efforts to minimise security threats by the organisation.

2. Employee Attitude Towards IM

A 2003 survey [2] found that more than 65% of office workers in the UK and 39% in the US used IM for personal conversations at work as well as 80% of users in the UK admitting to using IM to gossip at work. Only 11% in the UK and 39% in the US used IM for business purposes only.

This survey demonstrates how a large majority of employees view the usage of IM in a relaxed manner, possibly without much thought to the impact such usage can have on the organisation resource.

This reiterates the point that policies need to be implemented before IM develops into a entirely acceptable workplace tool. Employees must know the boundaries between what is proper and improper IM usage behaviour.

3. Acceptable Usage Policies

As stated in the previous sections, policies need to be adopted in regards to instant messaging. These policies should include a general guideline for usage, concise definitions on what is and what is not acceptable and how unacceptable behaviour will be dealt with.

Senior management need to make it clear to employees what they think is appropriate usage of IM. For example, they need to state if they think that IM is appropriate for communications with customers or that employees should

stick to formalised, traditional modes of communication such as email or telephone.

Management and employees should also be concerned with whether they are compromising the organisation's liability or their own reputation. Currently, the legal system is still trying to establish where instant messages stand in regards to libel, defamation, fraud and other legal matters. However, the organisation needs to be aware that whatever is stated in IM communications, especially about other people, the organisation or other companies may damage the organisation's reputation. Therefore, a practice of IM etiquette needs to be outlined in the policy and adopted in the workplace.

4. Legal Issues

The usage of IM at work also generates several legal issues for the organisation.

a) Lawful Interception of Data

Law Enforcement agencies, such as federal police and intelligence agencies, can legitimately intercept information exchanged in an organisation. Therefore, the organisation needs to be careful about what is exchanged over IM.

b) Unlawful Interception of Data

Data exchanged in an organisation can be unlawfully intercepted by external entities such as rival organisations and by internal entities (employees). If the data intercepted is sensitive, its disclosure could be damaging to the organisation.

c) Libel/Defamation Cases

Like any other communication tool in the workplace, IM can be a possible source of libel or defamation suits. The content of IM conversations must be strictly regulated to prevent such lawsuits.

H. Literature Review Conclusion

From the information gathered from the literature review, the following points were deduced in regards to IM and the workplace:

- IM has evolved into a relatively complex, useful communication tool
- The usage of IM at work has major security issues that needs to be managed
- The usage of IM at work requires management to be proactive in regulating to ensure legal issues are minimised as well as ensuring that employee productivity is not negatively impacted.

It was also found that there were many enterprise IMs in the market, however the current trend favours the use of public IM. The enterprise IMs provide organisations with practical solutions to the IM securing problems. Hence it was deduced that the reason for its lack of utilisation was not a technical one.

When looking into the attitudes of managers and employees towards IM, it was believed that the problem with the lack of enterprise adoption might be a sociopolitical one.

IV. Project Development Process

This section details the project progression, including the decisions made and the reasoning behind them. The first major phase of the project consisted of three sub-phases:

- The project scope was focused on developing an IM application prototype. The literature review was originally geared towards this focus.
- Upon preliminary revision of the literature review, the scope of the project changed
- The project scope is now focused on developing an encryption add-on for a public IM application

The original aim of the project was to design and produce a close to ideal, viable IM enterprise solution to be used in organisations.

However, whilst conducting the literature review, it was found that there were many sufficient enterprise IM solutions available in the market. The problem was more of a sociopolitical one where employees and senior management seemed to object to the usage of enterprise IM for various reasons. This in turn created an influx of public IM usage at work, creating major security hassles.

Therefore, the scope of the project changed to focus on developing an add-on to public IMs that could be widely adopted whilst public IMs were more predominately used than enterprise IMs to help keep an organisation's network more secure.

A. Literature Review Findings

On the onset of the project, the aim was to develop a secure IM application that could be used in the workplace.

1. Jabber

During the early stages of research, it was determined that the IM application would be developed based on Jabber.

Jabber is an extensible, open source IM system that in many ways was designed to be lightweight and deployed within a private network [7]. The ability to connect multiple servers together when necessary and the ease at which customised servers and clients could be created made it an ideal choice as a base for a secure IM platform. Despite some initial research into potential Jabber servers and clients for modification, development based on Jabber was eventually abandoned.

2. Ideal Solution

As a result of researching the different IM structures and security issues, an ideal solution for a secure IM was determined. This ideal solution is detailed in the following sections. It was planned that Jabber would be used to develop this solution.

a) Internal Hosting

The greatest risk to a corporate network using a Public IM service is the exposure of information to networks not under the control of the corporation. For this reason, the first step in securing corporate IM is to deploy the IM service within the organisation Intranet. This immediately provides a number of benefits.

External dependency on IM service is no longer an issue, as the service is hosted and controlled as part of the internal network. Most important, however, is the elimination of threats introduced by having data moving into and arriving from the Internet. Threats from viruses and Trojans are reduced, while the potential for data interception is greatly limited, as information only passes through controlled networks, which are easier to secure. (Figure 10 Internal IM Hosting)

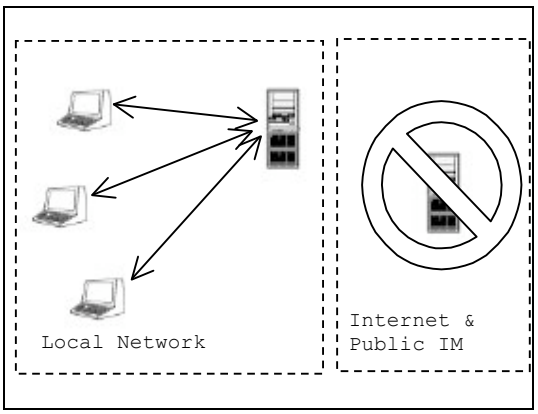


Figure 10 Internal IM Hosting

b) Server Architecture

Although a Single Server solution does not have the scalability of the Distributed Services architecture, a corporate network will more than likely not have to support millions of users, as is the case with the public IM solutions. The architecture should therefore be initially based on a Single Server solution, with the ability to expand into the Replicated Server architecture as the need arises.

This architecture is also well suited for use in multiple offices, with single servers located in each office communicating with each other, allowing inter-office communications.

c) Communications Model

All communications should occur through the central server, as per the client-server model. While peer-to-peer messaging reduces load on the IM server and can localise network data, there is an associated risk in the exposure of IP addresses. The distributed nature of peer-to-peer communications can also be difficult to control. The client-

server model of communication used within a switched network environment can provide predicable data paths, which can be secured more easily than a distributed mesh of instant message connections.

The client-server model is also better suited to IM networks with servers located in different branch offices. A peer-to-peer message between principles in different offices would require it own connection between the networks. The client-server model allows all data to be channelled through the single connection between servers. This is illustrated in Figure 11 Inter-server communication

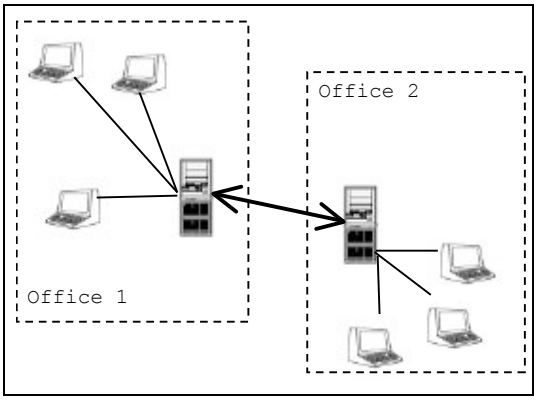


Figure 11 Inter-server communication

d) Functionalities

A Presence Service and Instant Message Service should be supported as a basic implementation. The system should also have to ability to add extra functionalities, such as whiteboard features, if required.

e) Account creation and management

It is not necessary for an internal messaging network to be listening for client account registration requests. Therefore integration with a network directory such as a LDAP (see section IX.B.4, IM Account Creation) is the preferred method of account management. If LDAP integration is not available, accounts should be manually configured by administrators. The client should have no facility for initiating account registration.

f) Client

A standalone application is preferred. Although this requires that client software be installed on all workstations, a web-based interface increases the load of IM servers and network traffic. A Java based client would provide cross-platform compatibility.

g) Encryption

Instant messages should be encrypted. Although the probability of data interception decreases when an IM service is hosted internally, there is still some risk of eavesdropping from within the network itself. Communications between IM servers should also be encrypted, as they carry an aggregate data stream of instant messages and would be an ideal target for data interception.

3. Why isn't Enterprise IM Widely Adopted?

As the research progressed, it was noted that there were many feasible enterprise solutions in the market that had similar properties to what was concluded to be the "ideal solution". However they weren't being adopted (see section III.F). The research then focused on why these applications were not being utilised in the workplace even though they were more beneficial – security-wise – than a public IM.

So why is it that enterprise IM applications aren't adopted? From studying various surveys and reading about management and employee attitudes towards IM, the following probable reasons were deduced:

- Senior managers don't see IM as a work tool and therefore do not support spending the time and resources to implement an enterprise IM system
- Some employees may feel that if an enterprise IM was adopted, then IM usage would be overtly formalised and regulated are therefore resist the move to enterprise IM
- Companies may not have the monetary resources to purchase an enterprise IM system
- Companies may not have any available IT resources to manage the system
- Companies may see that it takes to much time and effort to coordinate HR and senior management time to develop acceptable user policies for IM
- Employees may oppose the implementation of enterprise IM as it may take away their ability to communicate with others outside the company (namely family and friends)

B. Solution Amendment

Given the results of the assessment of why many organisations were not implementing enterprise IMs, it was concluded that the original planned solution of developing another enterprise IM was unnecessary. An alternative solution to the problem was then investigated.

1. Alternative Solutions

Unfortunately, real world issues such as those identified in section IV.A.3 meant that the ideal solution as found section 6.1.2 may not be feasible. For example, the infrastructure and expertise required in order to integrate an idealized IM system may prove unattractive to smaller companies that wish to use corporate IM, but cannot see the benefits outweighing the costs.

The following section details the alternative solutions that were considered to solve the problem.

a) Outsourced Hosting

A possible alternative to deploying an internal server for IM services is to outsource server operations to a third party company. Such a system would be operationally identical to the ideal solution, the main difference being that the IM server is located and managed off-site rather than internally.

An evident advantage of this solution is that no or limited infrastructure need be added to the company network, and there is no change in the duties of network administrators. A simple matter of client installation and configuration is all that would be required in order for the system to become operational. Technical help and client updates would also handled by the company operating the server. A similar approach to this method is seen in the enterprise IM solutions offered by Yahoo! and Bantu (see section III.E.1.d) and III.E.2.a) respectively).

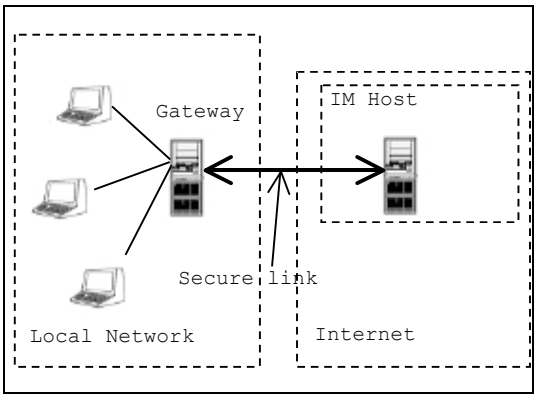


Figure 12: Outsourced Hosting

The major concern with such a system is that there must be a large level of 'trust' in the company hosting the service, as they are able to control and intercept all IM traffic. Certain agreements would have to be made between the host and subscriber, to ensure that information is not interfered with by the host company. Issues of external dependency would also need to be addressed, such as reliability and network uptime. Security measures used over the connection between the IM host and the corporate network would also need to be considered.

Whether this option may be cheaper for a company in the long term is debatable, as ongoing hosting expenses would be incurred. For ease of implementation, cost and the level of security obtained, outsourced hosting proves to be an attractive solution.

b) Securing Public IM

The cheapest (although least secure) method of securing IM in the corporate environment would be to leverage the existing public IM infrastructure and attempt to secure data passing over this network with encryption. While solutions such as AIM Enterprise Gateway (section III.E.1.a)), Akonix L7 Enterprise (section III.E.1.b)) and IM Manager (section III.E.1.c)) already attempt to leverage existing public IM networks, they still require that a gateway server be installed in the network. This solution would provide security directly from the client level.

Implementation would be simplified, as installation can occur over existing network infrastructure that already has access to the Internet. In addition, there are no agreements to be made with any third party host. Installation of the appropriate security plug-in is all that should be required.

Although this solution seems to contain all the security flaws that made public IM usage insecure in the first place (such as threat of malware and firewall tunneling) , in terms of feasibility of implementation this method would be the most realistic to achieve, especially for smaller companies wishing to take advantage of IM and require some level of security.

2. Revised Solution

From the IM security research and the assessment of the alternative solutions listed in the previous sections, the proposed solution had been changed to develop an encryption add-on to a public IM. This ensures that sensitive information that may be exchanged by employees over public IM systems remains undisclosed whilst senior management decide how to deal with the IM situation in their organisation.

C. Encryption Add-On Development

From the literature review (section III.D), it was established that cryptography could be used to keep information secure. There are many cryptographic algorithms. These algorithms are classed into groups called primitives.

Following the study of cryptographic algorithms, an algorithm was chosen for the proof of concept development. The aim of the proof of concept was to prove that an add-on can be implemented to a public IM application to keep the information exchanged secure.

1. Cryptographic Algorithm Primitives

Cryptographic algorithms can be categorised under three main primitives: unkeyed, symmetric keyed and asymmetric keyed (Figure 13:Cryptographic Algorithm Primitives).

Unkeyed algorithms don't require the implementation of any keys. The main subclass in this group is the arbitrary length hash functions. Hash functions generate a hash value (basically a small message summary) from a large message source. Hash functions are used as components in many cryptographic processes, including digital signatures, key establishments and random number generations. Hash functions are also used for Message Authentication Codes

(MAC), however these use a symmetric key and are therefore classed under symmetric key algorithms.

Symmetric key algorithms are also known as secret key algorithms. They use a single key to manipulate the data. Symmetric keys are shared by authorised entities and should be kept secret from everyone else. Symmetric keys are used in the following ways:

- Providing confidentiality through using the same key to encrypt and decrypt the data. Unauthorised entities should not know this key.
- Form part of the key establishment process
- Generate pseudorandom numbers
- Provide authentication and data integrity checks in the MAC process as the same key used to generate and validate the MAC

Asymmetric key algorithms are also known as public key algorithms. They function by using a set of two related keys (or a key pair). The key pair consists of a public key and a private key. The public key can be known by anyone, authorised or unauthorised. The private key should be known only to the entity that owns the key pair. The relation between the two keys will not expose what the private key is through knowing the public key. Asymmetric key algorithms are used in the following way:

- Calculating digital signatures
- Establishing cryptographic keying material

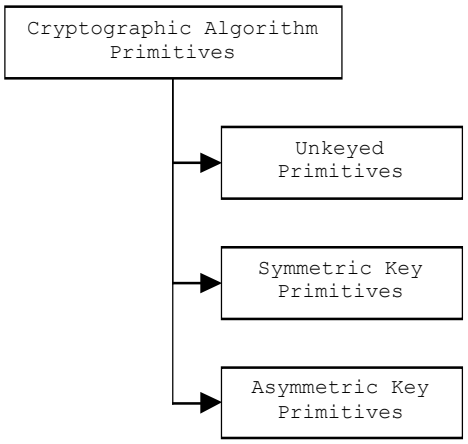


Figure 13:Cryptographic Algorithm Primitives

Each of the algorithms in the classes should be evaluated according to the following criteria to assess which is the best for the required application:

1. *Security level:* Usually, this criterion is difficult to determine. Normally, the level of security is defined by the upper bound of the level of work necessary to defeat the targeted information objective/s. This upper bound is also known as the work factor.

2. *Functionality:* As stated previously, a combination of cryptographic algorithms is needed to cater for all four of the information security objectives. The functionality defines which objective/s the algorithms are the most suitable to.

3. *Modes of Operation:* The algorithms will display varying characteristics depending on how they are applied and what type of inputs used. Therefore, the mode of operation will determine what type of functionality the algorithm will provide.

4. *Performance:* This measures the efficiency of an algorithm, depending on its mode of operation.

5. *Implementation Ease:* This measures how easy it is to implement the algorithm in a practical environment. For example, sometimes the environment may cause there to be a performance or level of security trade off because the hardware or software may be below the minimal requirements needed to run a more efficient or secure algorithm.

2. Cryptographic Keys

To understand the basic operation of an algorithm, a discussion about cryptographic keys is required. Cryptographic keying material is the cryptographic key and the information, which depends on the type of key, required to enable usage of the key. It is imperative that the cryptographic keying material is protected in order to make the effort of information security worthwhile (see Kerckhoffs' Principle for more discussion). For example, there is no point in having a locking system for the car if the keys are left in the door of the car. The protection mechanism required depends on the type of key and the information security objective for which the key is being implemented to meet.

This section details the cryptographic keying material protection requirements as outlined by the National Institute of Standards and Technology (NIST) [10]. The cryptographic keying material should be accessible for as long as the cryptographic application associated with it is operational. There will also be some types of keys that need to be archived, as they are required beyond the key's "operational" period, referred to as the key's cryptoperiod.

First of all, the keying material needs to be protected from any alterations. This type of protection is referred to as integrity protection. Integrity protection can be provided using the following:

- Cryptographic integrity mechanisms such as cryptographic checksums and hashes, MACs and digital signatures
- Non-cryptographic integrity mechanisms such as cyclic redundancy checks (CRCs) and parity checks
- Physical mechanisms such as key cards and biometric information to authorise access to computers that contain the keying material.

Secondly, private keys need to be kept confidential. This is normally done via encryption methods such as key wrapping or using physical mechanisms as listed above. In addition, there needs to be association protection. Association protection involves ensuring the availability of an information security service by making sure that the keying material is used with the correct data in the correct application.

These concerns are dealt with through cryptographic key management. Key management techniques will be a major focus for the second phase of the project.

3. Proof of concept development

The aim of the proof of concept is to develop a simple encryption add-on for public IMs to prove that they can be made more secure for use in the workplace.

The proof of concept does not cover any key management, data integrity and authentication issues. These will be dealt during the second phase of the project.

a) Selecting a Protocol

Jabber and MSNP were initially given consideration when choosing a protocol for the development and testing of a prototype, due to their ACSII based command structure. Although the Jabber protocol is more flexible than MSNP, the simple design of MSN and its widespread use made it appealing as a platform for the development of the proof of concept.

b) Method of development

There were two possible approaches to incorporating the encryption add-on. These were API Hooking of the official MSN client or the modification of an open source alternative.

(1) Interface Hooking/API Hooking

The official MSN client does not allow third party development of plug-ins. Therefore the plug-ins that are available communicate with the MSN client through API hooking. Hooking is the practice of intercepting MSN function calls and redirecting them to custom written code. demonstrates how hooking creates a layer between the window interface and the program functions that provides custom functionality.

(2) Open Source Client Modification

There are a number of open source clients that are able to communicate with the MSN network. These programs are available in a range of languages and may be edited and recompiled. It was decided that the prototype would be developed using one of these clients, for a number of reasons:

- Easy access to text messaging related code.
- Allowed focus to be on concept development rather than API hooking.
- Clients available in multiple languages.
- Clients available on multiple operating systems. This was important, as development was to occur across Windows XP and Mac OS X.

c) Open Source MSN Clients

There are a number of open source clients that will communicate with the MSN network. It was decided that the client to be used should not be multi-protocol and should only have the ability to communicate with the MSN network. A single-protocol client was chosen because of the reduced complexity of the program. Three MSN-only open source clients were considered for use as a test-bed for the prototype.

(1) Alvaro's/Another MSN (aMSN)

aMSN is a multi-platform client written in tcl/tk script with a large documentation and developer base. Supports many of the MSN features and also mimics the interface of the official client.

(2) openMSN

A Windows only Visual Basic 6 client that has basic messaging support and 'skinned' interface.

(3) TjMSN

A simple multi-platform Java client with support for instant messaging and file transfers. The current version of TjMSN contains support for the addition of plugins. This version was not available at the time of client selection.

d) Client selection process

The client to be modified was chosen based on a number of key factors, including multi-platform support, complexity and language. Although aMSN had the largest development community, the group was unfamiliar with tcl/tk script, meaning this had to be discounted. A lack of platform portability meant that openMSN was also unsuitable. TjMSN thus became the final choice, although this was not strictly by default. The group is most familiar with the Java language, while TjMSN itself is platform independent and includes only the features required for the prototype, resulting in simple code relationships.

e) Chosen Algorithm

The algorithm chosen for the proof of concept was DES (Data Encryption Standard)[11]. DES is a type of block

cipher. Block ciphers are encryption algorithms that work on fix-sized blocks of data. Block ciphers are part of the symmetric key primitives and therefore require a secret key.

DES was developed in the 1970s and became the first modern algorithm that had its full implementation details published. Compared to the algorithms available today, it is no longer useful. However, it was chosen for the proof of concept as:

- It is widely known so there was a lot of documentation and software libraries available to use
- It provides a basic introduction to implementing an encryption algorithm and will allow for a better understanding during the second phase of the project when a more feasible algorithm is chosen
- It demonstrated how factors such as performance and level of security could be affected by the algorithm used and the manner in which it was implemented. This recognition of such factors will be utilised in the second phase of development when the actual add-on is to be developed.

V. Results and Discussion

As stated in section IV.B.2, the design approach to providing a secure solution for workplace IM is to leverage an existing public IM network, initially the MSN Messenger network. By providing encryption and authentication routines to a third party client on a public IM network, identified threats such as data interception, spoofing/account hijacking and man-in-the-middle attacks (section III.B) can be reduced. The goal of the prototype was to demonstrate the ability to provide encrypted sessions over a public IM infrastructure and as such did not require key management, authentication or particularly strong encryption.

A. Implementing the add-on

. The proceeding sections outline the design decisions made in regards to building the add-on based on the research undertaken throughout the project.

1. Setting up and Managing the Encryption Session

It was recognised that in order for an encrypted conversation to take place, there would have to be some negotiation between the two clients. It was decided that a command signalling protocol should be established for the setup, management and termination of encrypted sessions

a) Session setup

- Request for encrypted session
- Acceptance/rejection of request
- Exchange of encryption key

b) Session Management

- Encryption and decryption of instant messages
- Separation and concatenation of long encrypted messages
- Transmission of encrypted messages

c) Session Termination

- Transmission of intent to end session
- Deactivation of encrypted session
- Notification of both parties that encryption has been deactivated
- Allow the resumption of clear text messaging

2. Protocol Command Propagation

Before a specific command syntax could be developed, a method of command transmission was required.

a) Content Subtype

As shown in the MSN protocol description (Protocol Analysis – MSNP), MSN clients are able to distinguish between different types of messages using the MIME content type/subtype field. As MIME headers are examined only by the MSN client, the use of custom subtypes is not specifically prohibited. Thus the option of identifying encryption protocol command messages by defining a custom subtype was given initial thought. Upon further consideration, however, it was decided that a protocol command syntax based on MIME header subtypes would not be appropriate, as it would limit its use to MSN Protocol networks. This did not fit the long-term design goal of creating an encryption add-on for use on any popular public IM network.

b) In-Message Signalling

Popular public IM protocols transmit messages as plain ASCII or UTF-8 encoded text. A command sent as ASCII within a message body would therefore be adaptable for use on any network. It was decided that the add-on should identify and act on command sequences embedded within the body of a text message.

3. Command Syntax

As commands are sent within the body of the message, the syntax was chosen so as to not be in a form that could be typed during a regular conversation. It was decided that commands should consist of a special escape sequence, followed by a space then a 2-digit command code.

a) Escape Sequence and Command Codes

The escape sequence decided upon was "###@", a sequence unlikely to appear during a regular conversation. It was decided that commands should be categorised into family and function in a manner similar to that of OSCAR. Within the two digit command code, the first integer represented the family, while the second represented the function. The initial command families created were the *Establishment (1x)*, *Management (2x)*, and *Termination* (3x) families.

Command Code	Function
11	Request encrypted session
12	Receive request for encrypted session
13	Acceptance of request
21	Message to be encrypted and transmitted
22	Message to be decrypted
23	Part 1 of split message for re-assembly and decryption
24	Part *n* of split message
25	Final part of split message
30	Indication of intent to stop encrypting messages

4. How add-on works

The add-on works by taking message strings and performing operations on them before passing it to the message display or message transmit functions of the client. The section below demonstrates how an encryption request is made.

a) Requesting Encryption

The user requesting encryption will press a button to activate encryption. This generates the message "###@ 01". Before this message can be transmitted, the add-on code recognises the sequence and constructs a new message "###@ 02 *Encryption Key*".

This message is transmitted to the receiver. It should be noted that this is not a secure method of exchanging keys and is for demonstrational purposes only. The add-on creates another message "You have requested an Encrypted session" and displays this in the sender's message history.

The receiving client will identify the sequence "###@ 02 *Key*" as a request for an encrypted session. The add-on will create a new message, "Do you wish to activate encryption?" and place this on the message received/history window of the user. The key will be stored for use if the receiver agrees to use encryption.

VI. Conclusion

IM allows for real-time communication between users as well as other functionalities such as file transfers, web conferences and SMS text messaging capabilities.

The IM industry has seen a rapid growth of users in recent years and has managed to progress from being simply used as a chat program at home to a possible communication tool at work.

There are two main types of IM applications: public and enterprise. Public applications are free and able to be downloaded off the Internet. They are the most widely used type of IM application in both the home and workplace. In the workplace, however, they pose a serious security risk to an organisation. The data that is exchanged over public IMs can be intercepted and distributed. This would be damaging to an organisation if the data were sensitive.

Enterprise IM solutions were created to specifically target the security problems that public IM posed on organisations. Enterprise IMs are not free and must be purchased as with any proprietary software package.

The aim of the project was to originally build a secure IM prototype that can be used at work. However, upon the discovery that the reasons why the enterprise solutions weren't used were of a sociopolitical nature instead of a technical one, the original aim was deemed impractical.

While enterprise IMs provide a more secure environment, surveys suggest that public IMs are still more predominantly used at work. Looking at the employee and manger attitudes, the following were some of the reasons deduced to explain why enterprise IMs were not widely used:

- Some managers did not see the purpose in implementing IM at work
- Some employees may resist being restricted to using enterprise IMs as could cut off their ability to communicate with people outside of work
- Companies may not have the monetary resources to purchase an enterprise IM system
- Companies may not have any available IT resources to manage the system

It was decided that a more constructive solution would be one that would make public IMs more secure whilst they were still more used than enterprise IMs. The aim changed to make an encryption add-on in an attempt to secure public IM at work.

A proof of concept prototype was made using DES encryption for a Java-based, open-source MSN IM client TjMSN. This prototype proved that public IMs could be easily made secure using a simple add-on.

As it was a simple demonstration, the prototype will need to be further developed to include such information security objectives such as authentication, data integrity and key management to make the prototype more secure. Also, DES is an old encryption algorithm and is succeeded by more efficient, newer algorithms. These algorithms will need to be looked at in depth to create a more practical, efficient add-on than the prototype developed.

VII. RECOMMENDATIONS

The prototype developed for the proof of concept was a simple design that would be impractical to implement in the real world. The reason for such a simple design was to develop a familiarity with how to implement a cryptographic algorithm for a specified purpose.

For the second phase of project development, the following is recommended to develop a viable solution:

- Choose a more effective algorithm to do the encryption of the messages sent
- Provide a key management system required to ensure that the keys used for encryption will distributed appropriately
- Provide a data integrity mechanism to ensure that messages aren't tampered with by a third-party
- Provide an authentication mechanism to ensure that only authorised parties are privy to sensitive information

When developing these features, considerations will need to be made towards:

- The level of security that would be provided
- The performance of the add-on. The add-on must be practical and not inconvenience users.

Once the MSNP-based prototype has been developed into a feasible add-on, an attempt will be made to try to adapt the code for an OSCAR-based IM. This will test the portability of the code. As AIM and MSN are the top two IMs used, it would be an advantage if the add-on could be used for both.

VIII. REFERENCES

[1] C. Metz, "IM Everywhere", http://www.pcmag.com/article2/0,1759,1359298,00.asp, November 2003 (as of June 3rd 2004)

[2] Blue Coat Systems, Inc, "Establishing an Internet Use Policy for Instant Messaging" Whitepaper, http://www.bluecoat.com, 2004 (as of June 3rd 2004)

[3] Websense, "Key Internet Usage Statistics", http://www.websense.com/company/news/stats.php

[4] Osterman Research, "Osterman Research Results", http://www.ostermanresearch.com/results/surveyresults_im0104.htm, 2003 (as of June 3rd 2004)

[5] Symantec "Securing Instant Messaging", February 2002.

[6] Fracis Chu "FaceTime, Others Plot IM's Corporate Future", eWeek, 2003 (as of June 3rd 2004)

[7] DJ Adams, "Programming Jabber", O'Reilly, 2002

[8] H. de Vos, H. ter Hofte, H. de Poot, "IM [@Work] Adoption of Instant Messaging in a Knowledge Worker Organisation", Telematica Instituut, 2004 (as of June 3rd 2004)

[9] Osterman Research, "Osterman Research Reults", http://www.ostermanresearch.com/results/surveyresults_1003b.htm, 2003 (as of June 3rd 2004)

[10] National Institute of Standards and Technology, "Special Publication 800-57: Recommendation for Key Mangement", June 2003

[11] U.S. Department of Commerce/National Institute of Standards and Technology, "FIPS PUB 46-3: Data Encryption Standard (DES)", 25 October 1999

[12] N. Ferguson, B. Schneier, "Practical Cryptography", Wiley Publishing Inc., 2003

[13] S. Singh, "The Code Book: The Secret History of Codes and Code-Breaking", Fourth Estate, 1999

[14] D. Kahn, "The Business of Babel: Cryptology in the 80s", 1988 Yearbook of Science and the Future, Britannica, 1987

[15] A. Menezes, P. van Oorschot, S. Vanstone, "Handbook of Applied Cryptography", CRC Press, 1996

[16] BBC h2g2, "Basic Cryptanalysis", http://www.bbc.co.uk/dna/h2g2/alabaster/A613135 (as of June 3rd 2004)

[17] P. Branch, "Lawful interception of the Internet", June 2003

[18] M. Sarrel, "Corporate IM", http://www.pcmag.com/print_article/0%2C3048%2Ca=110098%2C00.asp, November 2003 (as of June 3rd 2004)

[19] S. Gaudin, "IBM Manager: IM Muscles Up for Corporate Users", http://itmanagement.earthweb.com/secu/article.php/1448271, August 2002 (as of June 3rd 2004)

[20] John Clyman, Enterprise IM Reviews, http://www.pcmag.com/print_article/0%2C3048%2Ca=110098%2C00.asp, November 2003 (as of June 3rd 2004)

[21] M. Enbysk, "Blame it on Instant Messaging", http://www.bcentral.com/articles/enbysk/default.asp (as of June 3rd 2004)

[22] L. Vaas, "IM Genie Out of the Bottle", http://www.eweek.com/print_article/0,3668,a%253D19359,00.asp, December 2001 (as of June 3rd 2004)

[23] M. Day, S. Aggarwal, G. Mohr, J. Vincent, RFC2779 Instant Messaging/Presence Protocol Requirements, February 2000 (as of June 3rd 2004)

[24] M. Day, J. Rosenberg, H. Sugano, RFC2778 A Model for Presence and Instant Messaging, February 2000

[25] R. Movva, W. Lai, Instant Messaging and Presence Protocol, August 1999

[26] A. Shutko, OSCAR (ICQ v7/v8/v9) protocol documentation, http://iserverd.khstu.ru/oscar/ (as of June 3rd 2004)

[27] M. Mintz, A. Sayers, MSN Messenger Protocol, http://www.hypothetic.org/docs/msn/index.php (as of June 3rd 2004)

[28] AIM Doc, http://aimdoc.sourceforge.net/OFTdoc/oft.html (as of June 3rd 2004)

[29] J. Price, ASCII chart and other resources, http://www.jimprice.com/jim-asc.htm (as of June 3rd 2004)

[30] M. Wrzensinska, "A Secure Instant Mesaging System", June 2002

[31] AP, IM game spreads virulent ad-delivery software, www.cnn.com, February 2004 (as of June 3rd 2004)

IX. APPENDICES

A. RFC2778 and RFC2779

There has traditionally been no set of protocols that define what constitutes an Instant Messaging system. In 2000, the release of RFC2778 "A model for Presence and Instant Messaging"[24] and RFC2779 "Instant Messaging/Presence Protocol"[23] attempted to provide a set of guidelines as to the basic requirements of an IM system. The core functionalities described in the following sections are based on the recommendations of RFC2778 and RFC2779.

1. Presence Service

A method of distributing and managing the presence information of principles connected to the IM network. Presence Information is defined as the address at which a principle can be contacted and the current status of the user. The status of a user describes whether that user is contactable at the present point in time.

a) User Status

There are a variety of statuses that a principle can set theirs to. The statuses are used to inform presence subscribers whether a principle is:

- At their workstation and available to be messaged
- At their workstation, currently busy and unavailable for messaging
- Away from their workstation, for various reasons such as being out to lunch
- Offline

b) Privacy Control

The Presence Service also allows a principle to choose who is able to subscribe to their presence information and/or message them.

c) Presence Notification

This is the ability to notify subscribers to changes in the presence information of a principle.

2. Messaging Service

Facilitates and delivers instant text messages between principles on the IM network.

a) Instant message

An instant message is a short text message that is delivered between users in near real-time.

3. Additional IM Features

Sections 1 and 2 describe the most basic implementation of an IM system. Most modern IM systems, including the public systems studied in this report, include additional features in order to differentiate themselves.

a) Formatted Instant Messages

Most IM clients allow instant messages to be formatted using different fonts and font colours. The clients will also identify 'emoticons' and convert them into small images. Most applications are also able to recognise when a URL is sent to another user and will mark it as such so that when the receiving user clicks on the URL, a web browser will load the link.

b) Multi-Party Chat

Many IM applications also offer the option of adding multiple users to the current messaging conversation, in a manner similar to IRC.

c) Offline Messaging

The ability to message offline contacts so that when they come on, they will receive the message. This is achieved by storing messages on a central server and delivering it once an *online* presence notification is sent by the desired recipient.

d) SMS capabilities

Users are able to send a short text message to another user's mobile phone (depending on phone network support).

e) File Transfer

Most IM networks now allow users to send all manner of files to a contact, such as an image, audio file or a word processing document.

f) Audio/Video conferencing

Video and audio communications facilitated by the IM network.

B. IM Account Creation

A principle must have an account or be a member of an IM system before they are able to use the IM services such as presence notification. A principle obtains this account, generally consisting of an account name and a password, by registering with the system by some method. Several methods are detailed below.

1. Dedicated Port

An IM server may have a designated port that accepts connections solely for the creation of new accounts. This type of implementation would typically be found in a single-server architecture system. The ability to connect to the registration port would be integrated with the IM client.

2. Dedicated Server

Account registration can also be provided by a single dedicated server. As with the dedicated port method, it is likely that the client will have the ability to register accounts built in. The ICQ network is able to register new principles in this manner.

3. Web-Based

A website designed to handle registrations removes the need for registration functions within the client software. The user account details are stored directly into a database that is then accessed for authentication during login. [Figure N.1]. MSN and AIM use this method of registration.

4. Integrated LDAP

Some IM architectures are able to communicate with an LDAP (Lightweight Directory Access Protocol) Server, such as those found on some Intranets. If the authentication system of the IM network is able to integrate with the LDAP, then account registration is not necessary, with accounts based on the LDAP database. For example, an IM server can automatically allow connections from users listed in the database of a network LDAP server.

5. Pre-issued accounts

A user database could be created using a special software tool available to the IM network administrators. Accounts can then be issued to principles manually. Account creation cannot be initiated by client software. This is similar to LDAP integration and would be used in situations where LDAP was not available.

C. Protocol Analysis – MSNP

Information regarding MSNP was obtained mostly from sources [25] and [27], and from practical testing.

1. Server Overview

There are four main components which together form the basis of the MSN Messenger architecture. The first and most obvious of these is the MSN Messenger Client, which is the interface to the messenger network that the user is presented with. Less obvious to the user, however, are three equally important yet operationally distinct servers. These are the Dispatch Server (DS), Notification Server (NS), and the Switchboard Server (SS).

a) Dispatch Serve (DS)

The Dispatch Server, accessed via the domain name `messenger.hotmail.com` is the first connection made to the MSN network by the client software. The function of this server is to direct the user to an appropriate NS (based on the supplied .NET Passport). There is also a DS for HTTP connections, which is located at `gateway.messenger.hotmail.com`. The MSN Client will often store the address of the appropriate NS once located, so future connections can be made directly to the NS rather than through the DS.

b) Notification Server (NS)

The Notification Server manages user presence on the network. Logging into a NS announces to the system (and to presence subscribers) the availability or otherwise of the user. The connection to the NS could be regarded as the 'central' point which defines being connected to the MSN Messenger network. In addition to presence information, the NS stores and synchronizes user account details, provides notification of new Hotmail messages and checks the version of the MSN client being used.

The NS does not, however, handle Instant Messaging. Instant Messages are routed through the Switchboard Server (SS), although it is a primary function of the NS to setup connections to the SS.

c) Switchboard Server (SS)

The function of the SS is to establish and manage chat sessions between users on the network. Instant Messages in MSN do not travel peer-to-peer, but are forwarded to recipients through the SS. The SS is not concerned with the contents of messages sent, but will forward packets received based on the MSN protocol command contained within. Requests for file transfer and other out-of-band invitations are also sent through the SS, though the end result may not necessarily involve the SS itself (file transfers occur peer-to-peer, for instance).

It is important to note that connecting to the SS does not mean disconnection from the NS. A connection to the NS is maintained while in conversation as it is the NS which provides presence information on the network. While it is possible to be connected to the SS and disconnect from the NS, disconnecting from the NS signals the user as 'offline' from the perspective of the system. The official client does not allow connections to the SS without an active connection to the NS.

The obvious analogy of the MSN architecture would be that of the telephone operators and switchboards of the past. The NS can be compared to an operator, which, when a chat session is required, connects the user to the switchboard (SS), which carries the conversation.

As mentioned, the three main server types within the MSN system do not interact directly, but through client requests. This independent structure allows the system to address issues of network scalability, as there is no numerical limit on the number of servers of each type that can be deployed. Thus the system can theoretically support a large number of simultaneous users, adding servers as they are required.

d) User Account creation

MSN Messenger currently uses Microsoft's '.NET Passport' for user accounts. The .NET scheme is Microsoft's attempt to create an all-encompassing login system for online shopping and other services. As a result, there is no direct mechanism within the MSN Messenger network which creates user accounts. User accounts can be obtained at the Hotmail or the .NET passport sites. Once an account is created, it is possible to then sign into the MSN Messenger network directly.

2. MSN Messenger Protocol Syntax

Servers and clients within the MSN network communicate using a series of UTF-8 encoded commands over TCP/IP connections. Multiple commands may be located within one TCP packet, or a single command could be spread across multiple packets. Commands consist of a three capitol-letter command code, generally followed by transaction ID, none or multiple parameters and are terminated with a carriage return\new line (CRNL, \r\n). Commands are asynchronous in that the client may send multiple commands without first waiting for a server response. The server is not required to answer these requests in any particular order. A typical command would look like:

```
CMDCODE TransID Parameter1 parameter2\r\n
```

a) Commands Codes

All data sent through the MSN network is be preceded by a command code. Sent as plain text, they consist of three capital letters, the meaning of which is usually self

explanatory. Some important Command Codes are explained throughout the analysis.

b) Transaction Identifier (TrID)

As commands can be issued asynchronously and server replies may be out of sequence, a method of tracking command responses is required. Commands that require a response from the server therefore contain a TrID. The TrID is located one space after the command code, and is an integer with a value between 1 and $(2^{32} - 1)$. If the server must send multiple responses for a single request, each response will contain the TrID of the original command.

c) Parameters

A command may have none or multiple parameters. Separated by a single whitespace, parameters may be letters or numbers, and are case sensitive. A singe parameter cannot contain spaces. The final parameter is immediately followed by a CRNL, which signals the end of the command.

d) Example Command

The command identifying the version of MSN protocol being used would resemble the following:

```
Client -> Server: VER 1 MSNP10 MSNP9 CVR0\r\n
```

Command Code: VER = Version
Transaction ID: 1
Parameters: MSNP10 MSNP9 CVR0

This command informs that server that the client supports MSNP10, MSNP9 and Client Version 0 (CVR0)

e) Payload Commands

Payload commands are a special type of command which span multiple lines. Payload commands are divided into two sections, a command section and a payload section. The format of the command section is similar to that of a single-line command, with one exception. The last parameter of the command will always contain an integer representing the payload length (bytes) before the CRNL.

An important payload command is the `MSG` command, which performs a number of functions, including instant messaging.

f) Payload Command Parameters

The `MSG` command has different parameters depending on whether it is being sent or received. `MSG` commands can only be sent to the SS, but may be received from either the SS or the NS. When receiving a `MSG`, the command will contain three parameters and no TID. The first and second parameters are the account name and handle of the sender. The third parameter is the size of the payload. For example, a `MSG` received during a SS session might appear as `MSG friend@hotmail.com Gus 130\r\n`.

When sending a `MSG`, there will also be three parameters. The first parameter is a TrID. The TrID is required as the client may have chosen to have the SS provide acknowledgement on the delivery of the `MSG`. The second parameter is a single letter representing the type of acknowledgment requested. There are three types of acknowledgement that a client may specify, those being positive, negative and unacknowledged (none). The server will reply with an `ACK` command of the same TrID on the successful delivery of a `MSG` (positive) or with `NAK` of the failure of delivery (negative). No reply will be sent if no acknowledgement is requested, or if negative acknowledgement is requested and a message is delivered successfully. For example, if a client wished to receive a notification if message delivery is unsuccessful, then the outgoing `MSG` may appear as follows (where N = negative acknowledgement): `MSG 14 N 124\r\n`.

An important note is that outgoing messages do not contain any information in regards to the recipients. The SS will simply forward the payload text of the message to any other principles that are connected to the switchboard session. For this reason, 'private' messages cannot be sent in a multi-party chat, as there is no way in which to specify a recipient.

g) Payload

The SS does not process information in the payload; it simply forwards the data to the receiving client which then interprets the information. The payload itself is a MIME (Multipurpose Internet Mail Extensions) encoded stream, consisting of a header and a body. The header is based on a standard MIME header fields as defined in RFC-1521.

h) Payload Header

As with the MIME standard, the header begins with a MIME version followed by a content-type/subtype. In MSN, the content-type is always 'text', but the subtype will vary depending on the operation being performed. It is the subtype that allows the client to decide what to do with the information located in the payload body. For instance, the subtype "plain" indicates the payload body is a plaintext instant message. A 'character set' parameter is also present and will generally be UTF-8, although other ISO standards and non-English sets are supported. These form the basic set of parameters required in a payload header.

The end of the header is defined by a single line break. As the SS does not interpret any payload information, there is no real limit on what information can be included in the header of messages and a third party client may include support for non-official fields and parameters.

i) Payload Body

There is no restriction of what can be placed in the payload body, although the encoding must be that as specified

in the header (e.g. UTF-8). As there is no delimiter at the end of the body, the client will assume the body is complete when it counts the number of bytes specified in the payload length parameter. A plaintext message body will simply contain plaintext data, while other message types, such as request for file-transfer, will contain file-transfer specific instructions (in plaintext) to be interpreted by the client.

An example of an outgoing Instant Message payload command is shown below.

```
Client -> Server:
MSG 5 N 90\r\n
MIME-Version: 1.0\r\n
Content-Type: text/plain; charset=UTF-8\r\n
\r\n
Hi there!
```

Command: MSG (Message)
TrID: 5
Parameters: Negative acknowledgement ('N', inform if message undeliverable). Payload size 90 bytes.

Payload Header: MIME-Version, Content-Type and Character Set. The content subtype of this `MSG` identifies the payload as being an Instant Message ('`plain`'). There is no need to include the details of the recipient as the message is routed through the established switchboard connection.
Payload Body: "Hi there!" will appear on the screen of the recipient.

Once the client has read through 90 bytes, it will assume the next arriving character will be the start of a new command.

3. Text Messaging

All instant messaging on the MSN network occurs through the SS. When a user wishes to send an instant message to another user, a connection to that user (if available) is established through the SS. Messages are sent as `MSG` payloads in clear text, generally encoded as UTF-8.

MSN Messenger also supports multi-party switchboard sessions. There is no support, however, for private messages within a multi-user session. Any message sent during a multi-party session is replicated and sent to all participants, due to the nature of switchboard sessions.

When a switchboard session has been established, the client and server will generally exchange only three types of commands in relation to Text Messaging.

a) Outgoing Messages:

When a client wishes to send an Instant Message, it will issue a MSG payload command to the switchboard similar to the following:

```
MSG 10 N 133\r\n
MIME-Version: 1.0\r\n
Content-Type: text/plain; charset=UTF-8\r\n
\r\n
How are you?
```

The SS will examine only the first line of the command. The MSG command simply informs the SS that the payload is to be forwarded to any other principles connected to the SS session. The SS will then forward the payload as an Incoming Message.

b) Incoming Messages

The SS will alter a sent message so that when it is received, it will include the account name and user handle of the source. As an incoming message is an asynchronous server command and is not replied to, there is no TrID.

```
MSG friend@hotmail.com Buddy 125\r\n
MIME-Version: 1.0\r\n
Content-Type: text/plain; charset=UTF-8\r\n
\r\n
Good
```

c) Server Acknowledgements

The SS will reply to `MSG` commands if requested.

4. MSN File Transfer Protocol

MSN is able to support the transfer of files between users through the application of its own file transfer protocol, the MSNFTP (MSN File Transfer Protocol). These transfers occur on a peer-to-peer basis, and as such do not pass through the MSN network once established.

a) Connection establishment

File transfers can only be initiated through an active switchboard session between two users and are established using the same method as for other out-of-band invitations. It is during the initiation stage that the two clients negotiate who is to act as server and who is to act as client. While in most situations the sender would act as server, the MSNFTP protocol has the ability to allow the receiver to become the server in the event that the sender cannot accept incoming connections. This usually occurs when the sender is located behind a firewall or Network Address Translator. A typical request for file transfer, as sent through the switchboard server, would be of the following appearance:

```
Sender -> Recipient
MSG 3 N 289\r\n
MIME-Version 1.0\r\n
Content-Type: text/x-msmsgsinvite; charset=UTF-8\r\n
\r\n
Application-Name: File Transfer\r\n
Application-GUID:{5D3E02AB-...}\r\n
Invitation-Command: INVITE\r\n
Invitation-Cookie: 1534\r\n
Application-File: File.txt\r\n
Application-FileSize: 100\r\n
Connectivity: Y\r\n
```

It can be seen that the request for file transfer retains the standard MSG payload format. It is the content-type which notifies the client to the nature of the payload. The content-type "text/x-msmsginvite" informs the client that the text located within the payload is an invitation to an out-of-band session. The content-type does not specifically identify the payload as being a request for file transfer. As with plaintext messages, the payload occupies the number of bytes given in the initial command (289 bytes in this example), *after* the blank line. It is the payload that then identifies the message as an invitation to file transfer.

b) Payload Field - Invitation

Application-Name: This is a plaintext representation of the process being requested

Application-GUID: A Windows concept whereby applications are given Global Unique IDentifiers. The GUID supplied should be that of the File Transfer process.

Invitation-Command: Indicates the type of invitation. INVITE indicates an invitation is being given, while ACCEPT indicates that the message is an acceptance of an invitation.

Invitation-Cookie: Integer between 1 and 2^{32}. Is randomly chosen by the sender and is used to uniquely identify a negotiation between clients.

Application-File: The name of the file to be transmitted

Application-FileSize: Size of file, in bytes

Connectivity: Whether the sender is able to receive incoming connections. If 'Y', then the sender will act as server. If 'N', reciever will act as server.

The example below shows what an expected acceptance to the above invitation would look like:

```
Sender <- Recipient
MSG friend@hotmail.com friend 310\r\n
MIME-Version: 1.0\r\n
Content-Type: text/x-msmsgsinvite; charset=UTF-8\r\n
\r\n
IP-Address: 211.28.77.64\r\n
IP-Address-Internal: 192.168.0.3\r\n
Port: 6891\r\n
AuthCookie: 215354\r\n
Sender-Connect: FALSE\r\n
Invitation-Command: ACCEPT\r\n
Invitation-Cookie: 1534\r\n
Launch-Application: FALSE\r\n
Request-Data: IP-Address:\r\n
```

c) Payload Field – Acceptance

IP-Address: The IP address of the receiver

IP-Address-Internal: A second IP address, usually that of an attached NIC

Port: A port to which TC P connections can be made, 6891 by default.

AuthCookie: A random integer as with Invitation-Cookie, to uniquely identify a file being transferred.

Send-Connect: Specifies whether the file sender will be connecting the to receiver. If the sender acts as the server, this will be FALSE. It will be TRUE if the receiver must act as server.

Launch-Application: Whether an external application is to be launched

Request-Data: IP-Address: Requests the IP address of the sender.

d) MSNFTP Description

Once client/server negotiations between the users have been completed, the client will attempt to connect to the server with the supplied IP address and port. When a connection has been established, the MSNFTP takes over the file transfer as a peer-to-peer connection. The SS is no longer used to relay information in regards to the current file transfer.

e) Connection Setup

There are a number of basic commands which are used by the MSNFTP. These are: VER (version), USR (user), FIL (file), TFR (transfer), CCL (cancel) and BYE (end session).

A session will begin when the receiver sends a VER command, supplying the transfer protocol it wishes to use. The server will reply with VER and a matching protocol, or disconnect the client if no common protocol is found.

The receiver will then authenticate with the server, by sending a USR with two parameters, the first being a .NET passport, the second being the AuthCookie of the file. If the authentication is successful, the sender will reply with a FIL and a parameter indicating the size of the file.

The receiver will then send the TFR command, and the sender may begin transferring the file.

f) File Transfer

MSNFTP sends a file as a sequence of binary blocks, with header length of 3 bytes and a maximum body size of 2045 bytes.

The first byte of the header indicates whether there is any file data left to be transmitted. The byte will be '0' if there is still remaining data, and '1' if the transfer is complete. The length of the body is then specified by the remaining two bits, by adding the value of the second byte to the value of the third byte multiplied by 256. For example, if the body was 2016 bytes, then the value of the second byte would be 224 and the third would be 7 [224 + (7*256) = 2016]. In Hexadecimal the three byte header would appear as: 00 E0 07. The body contains the file information itself, in sequential binary form.

g) Session Termination

If a file transfer has been successfully completed, the receiver will send the BYE command, which will terminate the session. If the session must be terminated for reasons other than successful completion, the receiver (or sender) may send the CCL command. The CCL command may be sent at any stage during a file transfer.

5. MSN Login

The following sequence shows the initial login phase (simplified) when attempting to connect to the MSN Messenger network (from the perspective of the client).

```
-> XX represents outbound information, destination
XX
<- XX represents inbound to the client, of origin
XX.
```

a) Resolve Dispatch Server IP

```
1.-> DNS query UDP: messenger.hotmail.com
2. <- Query response IP:207.46.104.20 (a DS)
```

The initial DNS query allows the client to retrieve the IP address of the server(s) which will allow logon to the MSN Messenger network. The DNS query must occur as there are multiple Dispatch Servers in the MSN network. These servers are all bound to the same domain, and when a request is made the network will provide the IP address of any one of a number of DSs. Therefore, the client need only know the domain name to resolve to have access to a DS.

b) Communicate with Dispatch Server

```
207.46.104.20 = a resolved Dispatch Server (DS).
Communication now occurs using TCP.

1.  -> DS: VER 0 MSNP9 MSNP8 CVR0\r\n
2.  <- DS: VER 0 MSNP9 MSNP8 CVR0\r\n
3.  -> DS: CVR 1 0x0409 winnt 5.1 i386 MSNMSGR
5.0.0540 MSMSGS user@hotmail.com
4.  <- DS: CVR 1 6.1.0211 6.1.0211 5.0.0527
http://download.microsoft.com/download/8.../SETUPNT.
exe http://messenger.msn.com
5.  -> DS: USR 2 TWN I user@hotmail.com
6.  <- DS: XFR 2 NS 207.46.106.29:1863 0
207.46.104.20:1863
```

The client begins dialogue with the DS by sending the VER (version command) listing the protocol versions supported by the client (1). The DS then responds with the supported protocols, in this case the server supports both MSNP8/9. If the client provides an unsupported protocol, the DS will terminate the connection.

The client then sends a CVR message to the server, which details client information, such as operating system and architecture, the login passport and language (0x0409 = English, AU) of the user. The response of the DS, in this case, is the URL of the most recent MSN client, as the version used to log in is below that of current and recommended version (6.1.0211).

Once initial information exchange has occurred, the client issues the USR command, which represents a request for authentication (5). The 'TWN' and 'I' following the USR indicate that the client wishes to use TWN "tweene" authentication, and that this is the initial 'I' step. The DS then responds with the XFR 'transfer' command (6). The NS parameter signifies a transfer to a notification server, with the IP address and port following. The IP address following the '0' is that of the DS.

c) Tweener Authentication

Communications with the passport server occur over an encrypted Secure Sockets Layer (SSL) HTTPS connection. The SSL connection prevents the password and server responses from being transmitted in plain text. The client transmits the account name and password over this link, for authentication with the server. If authentication is successful, the passport server creates a 'ticket' which is passed back to the client. The client then attempts to log into a NS, using this ticket. The NS will match the given ticket with a copy it has of the ticket, a successful match meaning access to the network is granted.

d) Client Profile Update

```
1.  -> NS: USR 4 OK user@hotmail.com nick 1 0
2.  <- NS: MSG Hotmail Hotmail 458\r\n
MIME-Version: 1.0\r\n
Content-Type: text/x-msmsgsprofile; charset=UTF-
8\r\n
LoginTime: 1082684770\r\n
EmailEnabled: 1\r\n
MemberIdHigh: 229374\r\n
MemberIdLow: -2140007172\r\n
lang_preference: 1033\r\n
preferredEmail: \r\n
country: AU\r\n
PostalCode: \r\n
Gender: \r\n
Kid: 0\r\n
Age: \r\n
BDayPre: \r\n
Birthday: \r\n
Wallet: \r\n
Flags: 3\r\n
sid: 507\r\n
kv: 5\r\n
MSPAuth: 5ONKvLNDUDDEE189huA….\r\n
ClientIP: 211.28.51.118\r\n
ClientPort: 64523\r\n
```

The command received in (1) shows that the server has accepted the authentication process and has granted access to the Presence Service. As the MSN Client only stores limited user information locally, there is a need to update the client with the current user profile (2). The information is sent as a clear-text message (MSG) payload.

e) Contact List Syncronisation

```
1.  -> NS: SYN 5 0
2.  <- NS: SYN 5 1 5 5
3.  <- NS: GTC A
4.  <- NS: BLP AL
5.  <- NS: PRP MBE N
6.  <- NS: PRP WWE 0
7.  <- NS: LSG 0 Individuals 0
8.  <- NS: LSG 1 Coworkers 0
9.  <- NS: LSG 2 Friends 0
10.  <- NS: LSG 3 Family 0
11.  <- NS: LSG 4 Other%20Contacts 0
12.  <- NS: LST jcitizen@hotmail.com Johnny 11 4\r\n
BPR PHM 9123 4567\r\n
```

Upon updating the user profile, the client then sends the SYN command to the server (1). The client currently contains version "0" of the contact list. The reply of the NS (2), is that it contains the version "1" of the contact lists, therefore the contact lists must be syncronised (3-13). The following points summarise the contents of the synchronisation.

- LSG #: Group Lists that the user has created, the number representing the order.
- LST: Name Lists to which the user has subscribed presence information. The parameters of LST include the passport and nickname of the user. Also included are some presence information ('11' – this user is not blocked) and the group in which the LST is organized (4 – Other Contacts).
- LST Payload: contains extra information, in this case the phone number of 'Johnny'

f) Hotmail Notification

```
-> NS: MSG Hotmail Hotmail 221\r\n
MIME-Version: 1.0\r\n
Content-Type: text/x-msmsgsinitialemailnotification;
charset=UTF-8\r\n
\r\n
Inbox-Unread: 5\r\n
Folders-Unread: 0\r\n
Inbox-URL: /cgi-bin/HoTMaiL\r\n
Folders-URL: /cgi-bin/folders\r\n
Post-URL: http://www.hotmail.com\r\n
```

The final process of login and authentication is the notification of new email in the user's Hotmail inbox (if applicable). This is sent as a `MSG` payload with the Content-Type: `msmsgsinitialemailnotification` informing the client that the payload is an email notification.

6. Firewall Implications

a) MSN HTTP Protocol

In the event that port 1863 has been blocked by a firewall, it is possible to use MSN through HTTP. By connecting to a special dispatch server located at `gateway.messenger.hotmail.com:80`, MSN Protocol commands can be sent within the body of HTTP requests. In order for HTTP connections to be made, the client must have access to HTTP and DNS ports. The messenger will not work, however, if the 'hotmail.com' domain cannot be resolved by DNS.

As HTTP servers will not send information without a request, the MSN client must continually 'poll' the server so that any server responses that are awaiting delivery may be sent. Polling is not necessary however, if the client is actively sending requests. Commands from the client are sent as `POST` requests to the server, which processes them with a CGI script.

The client wrapper is HTTP1.1 compliant, while the server attempts to maintain HTTP1.0 compliance as a precaution to prevent a proxy from closing the connection.

D. Protocol Analysis – OSCAR

Although both AIM and ICQ use the OSCAR protocol, the networks do not have any interoperability. The OSCAR protocol also has several differences in implementation between networks, which are noted where appropriate. OSCAR information was taken mainly from references [26] and [28] , and also through practical testing.

1. Server Overview

The OSCAR architecture consists of four distinct servers. These are the Authentication Server (AS), Basic OSCAR Service (BOS) Server, Advertising Server (Ad) and an Out-of-band Services server. As it is the protocol base for two distinct networks, it is also able to support two client applications. These are the AIM and ICQ clients.

a) Authentication Server (AS)

The Authentication Servers are located at the domains login.oscar.aol.com and login.icq.com for AIM and ICQ respectively. This domain is the first connection point for the client software. The client will initially contact the AS and make a request for authentication. If the request is granted and authorization is successful, the client will be supplied with the address of the BOS and given an authorization cookie. This authorization cookie is used to access the various servers in the OSCAR architecture. The client will disconnect from the AS once authorization is complete.

b) Basic OSCAR Service (BOS)

The BOS supports the major functions of the IM network, such as presence notification and text messaging. In addition, the AIM BOS provides localized weather information and stock quotes. The BOS contains several other differences between the ICQ and AIM networks. ICQ contains a great deal of user details, which are stored and synchronised between the ICQ client and BOS. The AIM network stores this information on an AOL web server, so the only the email address, screen name and account name/password of the user is stored on the AIM client and BOS. A connection to the BOS is maintained throughout a session on the OSCAR network.

c) Advertising Server (Ad)

The advertising server is how AOL Time Warner generates revenue from ICQ and AIM. Both clients will connect to the advertising server at regular intervals to retrieve advertising information.

d) Out-of-Band Services

The Out-of-band Services Server simply allows the clients to establish sessions of a type which are not implemented directly by the BOS. This may include voice or video conferencing.

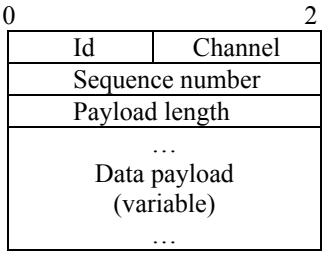
2. OSCAR Protocol Syntax

Servers and clients on the OSCAR network communicate over TCP/IP using series of frames and binary commands. Multiple OSCAR commands can occur within a single TCP packet, or can be spread across multiple TCP packets. There are two basic types of frames, the FLAC/FLAP and SNAC. It was not possible to find a source which described the meaning of these acronyms.

Due to the number of OSCAR commands and their hexadecimal nature, examples featuring actual sequences of protocol data have been reduced. An attempt has been made to provide a textual explanation of the process involved.

a) FLAP Datagram

All commands are placed into a datagram known as a FLAP. This datagram contains a 6-byte header and variable length payload. The payload generally consists of a smaller frame called a SNAC, although may occasionally contain a TLV (Type Length Value) frame. The FLAP is used to manage the connection between the OSCAR system and client. All FLAP communications are synchronous and occur in sequence. A typical FLAP frame has the structure as shown below (size in bytes):



Id: This is always 0x2A, and indicates the start of the FLAP frame.
Channel: a FLAP may be on one of five channels, where each channel identifies the function of the FLAP data. The channels available are:

- 0x01: New Connection Negotiation
- 0x02: SNAC data
- 0x03: FLAP error
- 0x04: Connection close negotiation
- 0x05: Keep alive

Sequence number: FLAP sequence numbers are used for error detection. The origin of the sequence occurs in the range 0x0000 to 0x8000 and is selected randomly. The sequence number is incremented for each new FLAP command. In the event the sequence reaches 0x8000 it will be reset to 0x0000.

Payload Length: This specifies the size of the payload in the FLAP payload field. The end of the FLAP frame is simply the point at which the payload field finishes.

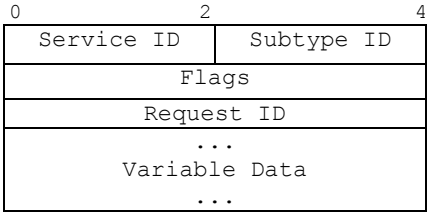
The FLAP header for a command requesting login might thus appear as: `2a 01 11 4F`

```
2a = frame start
01 = channel 0x01 (login request)
11 = sequence number
4F = payload length
```

The majority of FLAP communications occur over the SNAC (0x02) channel, as SNAC is the main form of communication used with the BOS.

b) SNAC Datagram

The SNAC provides the basic command set that is used in communications between the client and server. A SNAC datagram also contains a header and payload. The typical SNAC structure is shown in below.



Service/subtype ID: Service IDs represent a family of SNAC services. A family is a collection of related commands, for example the "Location services (0x0002)" family. Each family contains a number of subtypes. A subtype of the Location services family is "Set user Information (0x0004)". The Service Families available are shown in the following table. The Service ID and Subtype ID therefore indicate the purpose of the SNAC.

Flags: the flag portion of the OSCAR protocol has not been fully explored and documented.

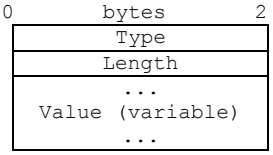
Request ID: Similar to the Transaction ID of MSNP, and allows the client to keep track of the responses to specific requests, if required.

Family ID	Name
0x0001	Generic service controls
0x0002	Location services
0x0003	Buddy List management service
0x0004	ICBM (messages) service
0x0005	Advertisements service
0x0006	Invitation service
0x0007	Administrative service
0x0008	Popup notices service
0x0009	Privacy management service
0x000a	User lookup service (defunct)
0x000b	Usage stats service
0x000c	Translation service
0x000d	Chat navigation service
0x000e	Chat service
0x000f	Directory user search
0x0010	Server-stored buddy icons service
0x0013	Server Side Information (SSI) service
0x0015	ICQ specific extensions service
0x0017	Authorization/registration service

The data payload of the SNAC can contain single or multiple parameters. Each parameter within the SNAC is framed within what is called a "Type Length Value" (TLV) field. There is no end delimiter for a SNAC frame; it will end at the point at which the FLAP frame ends, as specified by the FLAP payload length field.

c) Type Length Value (TLV) Field

The TLV is used to organize data parameters within a SNAC or FLAP. Like the SNAC and FLAP, it can be considered as having a header and body of itself. The TLV structure is shown below.



Type: A 16bit value that represents the category of the value in the payload. For example, a roasted password is given the type code of 0x02.
Length: The length of the value in the payload.
Value: The actual data, of a type specified in the *type* parameter.

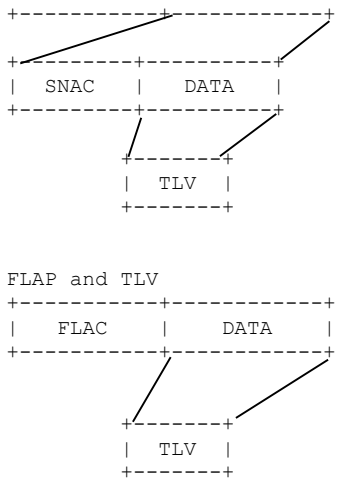
An example TLV could be the transmission of a roasted password "pass". This would appear as: `00 02 00 04 83 47 F2 B7`
Type: 00 02 (0x02 – data is roasted password)
Length: 00 04 (0x04 - password is 4 bytes long)
Value: 83 47 F2 B7 (the password "pass" after roasting, 4 bytes long)

Multiple TLVs may appear within a single SNAC or FLAP, depending on their function.

d) Overall Structure

The two possible structures of an OSCAR command frame are shown in the diagrams below.

```
FLAP, SNAC and TLV
+-----------+------------+
|   FLAC    |    DATA    |
```



3. Text Messaging

OSCAR supports the same modes of messaging as in MSN. Messages are transmitted as ACSII strings within the payload of a SNAC datagram. These strings are generally formatted within HTML tags. OSCAR also has to ability to send messages peer-to-peer. In AIM this can be optionally selected, while ICQ will attempt peer-to-peer messaging where possible. A hex dump of a transmitted text message is shown below (some important sections appear in bold):

```
2a 02 1e 90 00 88 00 04 00 06 00 00 32 05 00 06
45 34 46 30 37 34 00 00 00 01 0d 6c 61 7a 79 61
73 73 6d 61 73 74 65 72 00 02 00 62 05 01 00 03
01 01 02 01 01 00 57 00 00 00 00 3c 48 54 4d 4c
3e 3c 42 4f 44 59 20 42 47 43 4f 4c 4f 52 3d 22
23 66 66 66 66 66 66 22 3e 3c 46 4f 4e 54 20 4c
41 4e 47 3d 22 30 22 3e 73 6f 20 74 68 69 73 20
69 73 20 41 49 4d 2e 2e 2e 3c 2f 46 4f 4e 54 3e
3c 2f 42 4f 44 59 3e 3c 2f 48 54 4d 4c 3e
```

```
*.......  ....2...
E4F074..  ...lazya
rtmaster ...b....
......W.  ...<HTML
><BODY B GCOLOR="
#ffffff" ><FONT L
ANG="0"> so this
is AIM..  .</FONT>
</BODY>  </HTML>
```

This hex dump shows the complexity of the OSCAR command structure when compared to MSNP. The FLAP start code (2a) can be seen at the beginning of the sequence. The channel (02) means the data is contained within a SNAC frame. The sequence `04 00 06 00 00` is the beginning of the SNAC frame. 00 04 is the SNAC family (Inter-Client Basic Message, ICBM), 00 06 is the subtype (CLI_SEND_ICBM) and 00 00 are the flags. The sequence 00 01 0d is the type and length for the name of the sender, in this case "lazyartmaster". The sequence 00 02 00 62 is the type and length for the message. The message is:

```
<HTML><BODY B GCOLOR="#ffffff" >
<FONT LANG="0"> so this is AIM... </FONT></BODY></
/HTML>
```

This message is formatted as an HTML page. Only the text "so this is AIM.." would appear to the user. There is no code which signals the end of the frame, as in this case the frame ends with the ASCII code for ">", the last character. Like MSNP, instant messages are used to set up other types of communications, such as file transfer.

4. File Transfer

OSCAR supports file transfer through the use of the OSCAR File Transfer protocol, OFT [28]. The OFT operates in a similar manner to MSNFTP, but with several differences in implementation. In OFT, the receiver will act as the server. For example, if X requested a file from Y, then the BOS provides the IP address and port of X. Y then makes a direct connection to X. The sender then sends a file called "listing.txt" which contains information regarding the file to be transferred and the platform of the sender. After validating the listing file, the sender will transmit the requested data in raw binary format. The receiver will terminate the connection once it has received the required file.

5. Login

OSCAR supports two methods for authentication and login. These are known as Channel 0x01 Authorization (password roasting) and MD5-Based authorization. Due to the complicated OSCAR command structure and the number of steps involved in authorization, the process will be described rather than shown.

a) Channel 0x01 Authorization

This method is often used by ICQ clients and simply consists of a login request from the client and a reply from the AS. The client will first connect to the AS, before sending a "cli_ident" FLAP frame. An extract from such a frame is shown below.

```
2A 01 00 06 00 80 00 00 00 01 00 01 00 06 31 32
33 34 35 36 00 02 00 04 83 47 F2 B7
```

The first ten bytes are FLAP header information. The initial highlighted section is the Type and Length of the username (UIN). The UIN is "123456" encoded in ASCII. The second highlighted section is the Type and Length of the password. In this case it defines a "roasted" password of length 4 bytes. The roasted form of the password "pass" is shown as 83 47 F2 B7. The roasted password can be decoded by using the roasting array key.

Decoding a roasted password: Channel 0x01 authentication uses a special array of binary numbers that are used to "roast" a password. The array consists of: [0xF3, 0x26, 0x81, 0xC4, 0x39, 0x86, 0xDB, 0x92, 0x71, 0xA3, 0xB9, 0xE6, 0x53, 0x7A, 0x95, 0x7C]. In order to roast a

password, the ASCII equivalent of each character in the password is xored with the corresponding number in the array. An encoded password is decoded by simply xoring the encoded form with the array once more. The example below demonstrates the encoding and decoding of the password "pass" using the roasting method.

Password = pass

Letter	ASCII Code	Binary
P	70	01110000
A	60	01100001
S	73	01110011
S	73	01110011

First four digits of roasting array: F3 26 81 C4

Array value	Binary
F3	11110011
26	00100110
81	10000001
C4	11000100

Exclusive OR of corresponding values:
(password top, array bottom)

```
01110000 01100001 01110011 01110011
11110011 00100110 10000001 11000100
10000011 01000111 11110010 10110111
```

The password would therefore be transmitted as: 83 47 F2 B7 (as shown in login example). To recover a password, simply Exclusive OR the roasted password with the array.

(roasted password top, array below)

```
10000011 01000111 11110010 10110111
11110011 00100110 10000001 11000100
01110000 01100001 01110011 01110011
```

b) MD5-Based Authentication

A more secure method of login then channel 0x01. The client will connect to the authentication server and request authentication. The server will respond by sending an MD5 key, which the client uses to hash the account password. The client then transmits the hashed password to the server for verification. If successful, the server will then provide a ticket for login to the BOS.

6. Firewall implications

In the event that a firewall has blocked the default port of 5190, OSCAR has the ability to tunnel protocol information through various other ports that may be available. The "auto configuration" feature of AIM allows the user to probe the network in order to find any ports that might be open. To gain a picture of what ports AIM can operate on, a number of tests were carried out using *Kerio Personal Firewalls* outgoing connection dialogue.

The outgoing connection dialogue allows the user to block network access on a per application basis. AIM was denied all network access by default, meaning in order to access the Internet it would have to be given specific permission by the firewall. The "auto configure" utility was then progressively denied access on connection attempts, and the port number used was noted. It was thus possible to observe several of the ports through which AIM can operate. These are summarised below.

Port	Protocol
5190	OSCAR default
31	Daytime (RFC 867)
20	FTP-data
21	FTP
23	Telnet
25	SMTP (mail)
37	Time (RFC 868)
53	DNS
69	Trivial File Transfer
70	Gopher
79	Finger
80	HTTP

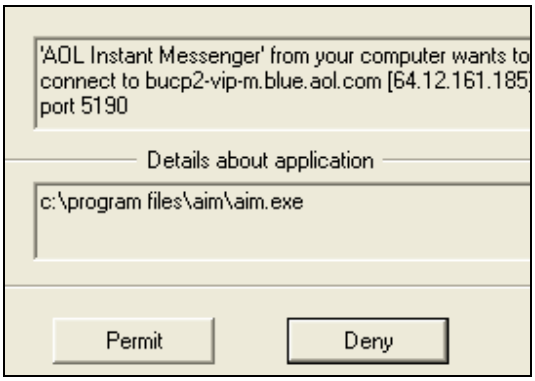


Figure 14: Outgoing connection dialogue

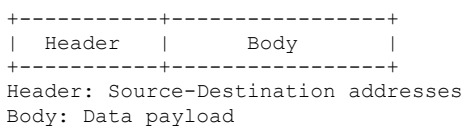
a) HTTP Proxy support

OSCAR also supports the use of an HTTP proxy server, if connections through other ports are not available.

Encrypted digital signatures also allow the receiver to verify the authenticity of a message.

E. Packet Sniffing

Information is exchanged over networks (and the Internet) by segmenting data into small packages called 'packets'. On the simplest level, these packets can be divided into two sections, a header and body. The header contains information in regards to the source, origin and size of the packet, while the body contains the actual data payload. Packet sniffing is the process whereby a software package is used to monitor data packets travelling along a section of a network. Header information can be used to identify the source and destination nodes of a packet, while the actual data being transmitted can be recovered from the body of the packets. The figure below shows a typical packet structure.



F. Data Integrity and Authentication

The main function of data integrity is to check whether any alterations of the data (information). Data can be altered by deleting and modifying all or part of the data as well as inserting additional data.

From the scenario below, how would Bob be able to tell whether the message that he received was the one that Alice had sent?

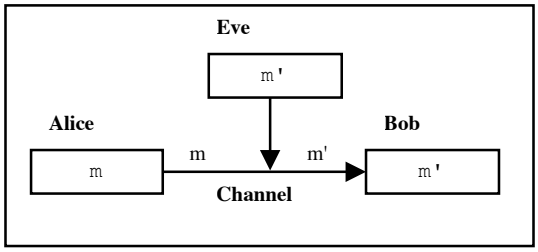


Figure 15:Data Integrity and Authentication Scenario

It is not possible to completely prevent data alternation. However, it is possible to determine with high probability whether or not the data has been tampered with.

There are several cryptographic devices that are used to ensure data integrity including message authentication codes (MAC), keyed hash functions and digital signatures.

These methods will only give a partial solution to ensuring data integrity. Eve can still delete messages sent by Alice (or Bob) as well as repeat old messages or change the message order. To counter this, the data integrity methods will be combined with a numbering scheme so that Bob and Alice can differentiate from old messages as well as detect whether a message has been deleted by Eve.

As well as using a numbering scheme, authentication can be used so that Bob can verify that it is actually Alice that is sending the message. Digital signatures and MACs used for data integrity are also used to provide authentication. In fact, the usage of MACs and digital signatures in data integrity is basically the same as in authentication. In both situations, the methods verify that it was Alice sent Bob the message.

However, in the aim for data integrity, the context is that if Bob cannot verify that it is Alice that sent the message then he assumes that the message has been tampered with in some way and discards it. In the case of authentication, the methods tell Bob that it is actually Alice that he is communicating with.

a) Message Authentication Codes (MAC)

A message authentication code (MAC) prevents the tampering of messages sent during a communication exchange. Although encryption would be able to keep Eve from being able to read the message, it would not prevent Eve from tampering with it.

MACs use a secret key that is only known to Alice and Bob. Alice sends the message m as well a MAC value that was computed by a MAC function using the key. Bob uses the key to generate a MAC value for the message received and checks it with the MAC value that Alice sent. If the values do not match, then Bob will discard the message under the assumption that the message has been tampered with.

b) Digital Signatures

Digital signatures work in the same way as MACs, however instead of using a secret key, it uses a public key.

In this situation, Alice will have a key pair consisting of a private key known only to her and a public key that she publishes. To send a signed message to Bob, Alice generates a signature using the secret key. Up to this point, the process is the same as with the use of MACs.

The difference is that Bob will verify the signature using Alice's public key, instead of using a secret key shared by Alice as with MACs. Furthermore, anybody who obtains Alice's public key can verify that it is Alice that sent the message.

G. Case Study - Buddlinks and the AIM network

The exposure of many AIM users to the 'BuddyLinks' adware Trojan [31] provides a real world example of many of the common IM security threats.

In February 2004, numerous AIM users received a message from contacts listed in their 'buddylist' that contained a link to a game parody of either Osama bin Laden or Saddam Hussein [Figure 5.1]. Hidden in the 'terms of agreement' documentation that arrived with the game were conditions that authorized advertising and distribution software to be installed. This software would execute and operate on its own accord. Once installed, the distribution software would send the game download URL to contacts on the 'buddylist' of the infected computer, without the knowledge of the user. In this way, it would seem to contacts that the link being sent to them was coming from a trusted friend. The advertising software would display advertisements on the infected computer or hijack web browsers and redirect them to other URLs.

Figure 16: Buddylinks game

The program achieved enough network penetration to force AOL to alter their own advertising information and warn users not to download the software [Figure 5.2].

Figure 17 AOL Warns users

The Buddylinks software used a combination of IM security flaws to propagate through the AIM network. Message *spoofing* made the program link appear to be coming from a trustworthy source – a friend on a users 'buddylist'. The scripting functions of AIM allowed the software creators to *hijack* the account of the infected user, sending spoofed messages to contacts without informing the user. This also showed the use of IM networks as a malware delivery system. *Network Exposure* also occurred, as the advertising software had the ability to connect to external servers to retrieve new advertisements.

Despite the dubious nature of the Buddlylinks software, it was high-profile in its behaviour and essentially harmless. This may not always be the case. A malicious program installed on a company network may log keystrokes, routinely sending a record to the virus writer – and may go unnoticed. The Buddylinks case serves to highlight that the social engineering skills of virus writers, together with the eagerness of users to download what appears to be amusing software, should not be underestimated. An instant message claiming to be a link to an amusing program could very well be circulated into a corporate network, much like the amusing emails that circulate on a daily basis. It is likely that many users would install such software, thus potentially compromising a network.

H. Kerckhoffs' Principle

In 1883, August Kerckhoffs published what he deemed were the essential six attributes that a military cipher should contain. They are [16]:

1. The system should be, if not theoretically unbreakable, unbreakable in practice.
2. Compromise of the system should not inconvenience the correspondents.
3. The key should be rememberable without notes and should be easily changeable.
4. The cryptograms should be transmissable by telegraph.
5. The apparatus or documents should be portable and operable by a single person.
6. The system should be easy requiring neither knowledge of a long list of rules nor involving mental strain.

These attributes, with modifications to adapt to the modern computer world, are still relevant today.

The second attribute was later expanded and became what was known as 'Kerckhoff's Principle'. The principle states:

"*In a cryptographic system the security of the system should not be dependent upon security of the method of encryption use"*

These days, the principle is normally stated as "*The security of the encryption scheme must depend entirely on the secrecy of the key, and not on the secrecy of the algorithms."*[12] This principle forms an important basis for the design of cryptosystems.