Centro Universitário de Brasília – UniCEUB
Faculdade de Tecnologia e Ciências Sociais Aplicadas – FATECS
Curso de Engenharia da Computação

Michelle Ribeiro Abuchahin

Projeto de Comunicação Segura TCP/IP

Brasília
2008

Michelle Ribeiro Abuchahin

Projeto de Comunicação Segura TCP/IP

Trabalho apresentado ao
Centro Universitário de
Brasília (UniCEUB)
como pré-requisito para a
obtenção de Certificado de
Conclusão do Curso de
Engenharia da Computação

Orientador: Prof. Marco Antônio Araújo

Brasília-DF

Dezembro de 2008

Dedicatória

À Deus,
Por ter me dado força nas horas em que fraquejei.
À Minha Mãe e Meu Pai,
Por todo amor e carinho que me confortou.
Ao Meu Irmão Diego,
Por estar sempre ao meu lado nos momentos em que mais precisei.
Aos Amigos Verdadeiros,
Por todo o apoio que me deram nesta caminhada.
Dedico-lhes essa vitória com emoção.

Agradecimentos

À Deus por ter me dado força e esperanças na conclusão de mais uma etapa da minha vida.

À minha mãe, Francisca, estrela guia da minha vida, pessoa pela qual tenho o maior respeito e admiração no mundo, aquela que não mediu esforços e abriu mão dos seus próprios sonhos para que o meu pudesse se realizar. Mãe jamais me esquecerei de você me dizendo: “Filha que tem Deus não tem medo”, isso me deu coragem e vontade de lutar por esse objetivo que se concretizou.

Ao meu pai, Américo, por ter me dado oportunidade de chegar até aqui, priorizando os meus estudos e me ajudando em tudo aquilo que era possível e até mesmo impossível dentro de suas limitações.

Ao meu irmão Diego, por sua tranqüilidade, pelo carinho e amizade propiciado durante todas as horas em que estivemos juntos.

Ao meu Tio Bosco, aquele que sempre me fez rir nos momentos de tristeza.

Ao meu orientador professor Marco Antônio Araújo, pela atenção, eficiência, objetividade e dedicação na condução desse projeto.

Ao professor Francisco Javier, pelo direcionamento e apoio incondicional na aprovação da proposta desse projeto.

A todos os amigos verdadeiros que me incentivaram nesta caminhada. Especialmente aos meus dois amigos que comungaram comigo o mesmo sonho de tornar-se engenheiro: Giuliano Boccucci e Luciano Cortez, jamais me esquecerei de vocês e que a nossa amizade dure eternamente.

“Após a dor vem a alegria, pois Deus é amor e não te deixará sofrer.”

(Simone Telésforo)

RESUMO

Este projeto tem como finalidade garantir segurança em um software desenvolvido na linguagem de programação Visual Basic (VB), uma vez que os softwares de comunicação instantânea presentes no mercado atualmente, como o Microsoft Messenger, o Yahoo! Messenger, ICQ e Google Talk não são considerados um meio seguro para o tráfego de informações, pois na maioria das vezes não oferecem mecanismos para garantir a confidencialidade, a autenticidade e a integridade da comunicação e esses três elementos são considerados os pilares de segurança em comunicadores instantâneos.

Na troca de mensagens em um comunicador instantâneo, a grande maioria dos usuários não tem ciência de que o teor das mensagens trafegadas poderá ser interceptado e até mesmo modificado por usuários não autorizados. Além disso, o emissor não tem a confirmação de que a mensagem que chegou até o receptor condiz com a sua mensagem original e vice-versa.

O software utiliza de mecanismos para aumento do nível de segurança nas mensagens enviadas e recebidas. São utilizadas técnicas de criptografia, ou seja, codificação de dados em informações aparentemente “ilegíveis” em mensagens “legíveis” apenas para o usuário que tiver a permissão de recebimento da mensagem e estiver diretamente envolvido no processo de comunicação. Para tanto, será utilizada uma chave criptográfica, tanto para o emissor quanto para o receptor. Assim sendo, a mesma poderá ser aceita ou não. Se a chave for descartada ela não poderá ser utilizada na próxima tentativa de estabelecimento da conversação. A conexão entre o emissor e receptor será realizada via socket, ou seja, finalidade para um elo bidirecional de comunicação entre dois programas.

Palavras-chave: Segurança, Criptografia, Comunicação Instantânea.

ABSTRACT

This project aims to ensure security in a software development in programming language Visual Basic (VB), since the instant communication software on the market today, as Microsoft Messenger, o Yahoo! Messenger, ICQ e Google Talk aren't considered a safe environment to traffic information, because most cases its don't provide mechanisms to ensure confidentiality, the authenticity and integrity to the communication and this three elements are considered the base of security in instant communicators.

In an exchange of messages in an instant communicator, the vast majority of users aren't conscious that the content traffic of messages can be intercepted and even modified by unauthorized user. And also, the issuer doesn't have confirmation that the message that reached the receiver matches its original message and vice-versa.

The software uses mechanisms to increase the level of security on the messages sent and received. Are use encryption techniques, that is, codes of data on information apparently "unreadable" messages in "legible" only to the user that have the permission to receive the message and is directly involved in the process of communication. There will be used both a cryptographic both key to the sender and to the receiver, since been the gone. It could be accepted or not. If the key is discarded it can't be used in the next attempt to establish the conversation. The connection between the transmitter and receiver will be conducted by socket, that is, finally for a bidirectional communication link between two programs.

Key words: Security, Encryption, Instant Communication.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1 - Introdução

1.1 Motivação

No meio social e profissional a comunicação instantânea é um intercâmbio utilizado para troca de informações em tempo real. O Brasil é o líder no uso de mensageiros instantâneos no ranking de países avaliados pelo Nielsen//NetRatings, com 6,6 milhões de usuários residenciais ativos. De acordo com a pesquisa Ibope//NetRatings, apresentada em 21/03/2005, o volume representa 60,2% de usuários residenciais ativos, que totalizaram 11 milhões de pessoas em fevereiro. A Espanha é a segunda colocada na lista, com 56% de internautas residenciais que utilizam mensageiros instantâneos[1].

Apesar de o Brasil ser o líder no uso de mensageiros instantâneos, ele ocupa a 10ª posição de países com maior número de internautas no mundo, conforme podemos observar na Figura 1.1, sendo os Estados Unidos o líder mundial[2].

Tabela 1.1 – Países com Maior Número de Internautas
(http://www.e-commerce.org.br/STATS.htm#C em 12/09/2008)

#	Pais	Usuários da Internet (milhões)	População (milhões)	Adoção da Internet
1	United States	209	299	70 %
2	China	123	1.306	9 %
3	Japan	86	128	67 %
4	Germany	51	83	61 %
5	India	40	1.112	4 %
6	United Kingdom	38	60	63 %
7	Korea (South)	34	51	67 %
8	Italy	31	59	52 %
9	France	30	61	48 %
10	**Brazil**	**30**	**188**	**16 %**

O foco do processo de sistema de comunicação instantânea da grande maioria dos softwares que estão no mercado pode ser descrito como um meio em que se deseja estabelecer contato com outros usuários. Cada usuário adiciona na sua lista várias pessoas com as quais ele deseja se comunicar e essa mesma ação será recíproca, permitindo que o contato apareça ativo e disponível para conversação. Levando-se em conta a privacidade, os usuários ali adicionados podem ou não aceitar a conversa, quando a conversa é aceita, inicia-se o tráfego de mensagens[3].

Na verdade este tipo de comunicação deveria garantir a segurança da informação, mas é importante salientar que não há um ambiente totalmente seguro em comunicação instantânea e que a tecnologia empregada complementa-se à educação das pessoas que terão acesso às informações[4].

O volume de ataques a serviços de mensagem instantânea e a redes P2P[1], para troca de arquivos, cresceu 3.295% no terceiro trimestre de 2005 em relação ao mesmo período de 2004. De acordo com a IMlogic, empresa norte-americana que monitora o segmento, o crescimento acumulado das ameaças nos três primeiros trimestres de 2005 a 2.083% maior que os noves primeiros meses do ano de 2004[5].

Dos incidentes verificados, 87% dizem respeito à propagação de worms[2], 12% correspondem a transmissão de vírus[3] e cavalos-de-tróia[4] por meio de transferência de arquivos, e 1%, à exploração de vulnerabilidade no usuário[5].

Entre as tendências observadas pela empresa está a sofisticação dos ataques, com o aumento do número de worms em múltiplas línguas, o que facilita a propagação para diversas geografias[5].

1.2 Objetivo

Com a grande divulgação e utilização da comunicação instantânea, esse meio de comunicação tornou-se alvo principal dos ataques virtuais. O objetivo principal deste projeto é garantir que as mensagens trafegadas em um sistema distribuído[5] de comunicação em rede aberta, como a internet[6] (um meio não seguro), só serão entendidas pelo usuário de origem e o usuário de destino. Assim sendo, podemos amenizar o risco de interceptação e posterior leitura das mensagens trafegadas na comunicação, para tentarmos garantir a integridade dos dados. Para tal garantia será utilizado um algoritmo criptográfico associado a um software de comunicação instantânea através de socktes[7].

---

1 P2P, Peer-to-Peer (do inglês: Ponto-a-Ponto), rede linear, rede distribuída ou rede não hierárquica é uma topologia de redes caracterizada pela descentralização das funções na rede, onde cada terminal realiza tanto funções de servidor quanto de usuário[6].

2 Worms, (verme, em português), em computação é um programa auto-replicante, semelhante a um vírus. Entretanto um vírus infecta um programa e necessita deste programa hospedeiro para se propagar, já os worms são um programa completo e não precisam de outro programa para se propagar[7].

3 Vírus, em informática, um vírus de computador é um programa malicioso desenvolvido por programadores que tal, como um vírus biológico, infecta o sistema, faz cópias de si mesmo e tenta se espalhar para outros computadores, utilizando-se de diversos meios[7].

4 Cavalos-de-tróia, trojan horse ou cavalo de tróia é um programa que age como a lenda do cavalo de Tróia, entrando no computador e liberando uma porta para um possível invasor[7].

5 Sistema distribuído, a computação distribuída, ou sistema distribuído, é uma referencia à computação paralela e descentralizada, realizada por dois ou mais computadores conectados através de uma rede cujo objetivo é concluir uma tarefa em comum[8].

6 Internet, é um conglomerado de redes em escala mundial de milhões de computadores interligados pelo Protocolo de Internet que permite o acesso a informações e todo tipo de transferência de dados[8].

7 Socktes, um soquete (do inglês socket) é generalizando uma tomada que designa uma cavidade ou região usada para ligar algum artifício específico[9].

1.3 Estrutura da Monografia

No capítulo 1 foi apresentada a motivação que me levou ao desenvolvimento deste projeto e o seu objetivo proposto.

No capítulo 2 é apresentado o conteúdo teórico que dará base para o desenvolvimento deste projeto.

No capítulo 3 é apresentado a segurança em mensagem instantânea, como ocorre sua estrutura, dentre os quais os malefícios e benefícios.

No capítulo 4 é apresentado a proposta de segurança, desde a ferramenta a ser utilizada até a execução do programa.

No capítulo 5 são apresentados os testes e resultados da execução do programa, demonstrando se os objetivos de intercepção e reprodução das mensagens com integridade e confiabilidade foram atingidos.

Para finalizar, no capítulo 6 é apresentada a conclusão e indicações para a continuidade do projeto.

Capítulo 2 – Softwares de Mensageria

Este capítulo mostra o conteúdo teórico que é utilizado para o desenvolvimento do projeto.

2.1 Sistema Distribuído

Um sistema distribuído é uma "coleção" de computadores independentes, que parecem para o usuário com um simples sistema. Esta definição tem dois aspectos. Primeiro trata do hardware: as máquinas são autônomas. Segundo trata do software: os usuários pensam que estão tratando de um único sistema. Ambas as definições são essenciais[10].

Os sistemas distribuídos são organizados por meio de camadas de software que é logicamente, uma posição entre um alto nível – camada que consiste a interação do usuário com a aplicação – e a camada que fica abaixo do sistema operacional[10].

O principal objetivo de um sistema distribuído é o fácil acesso que os usuários podem ter aos recursos remotos, como por exemplo: impressoras, computadores, dados, arquivos, páginas da web, etc. Existem muitas razões para partilhar estes recursos. A mais obvia é a econômica. Conectando usuários e recursos facilitam a troca de informação, como é melhor ilustrada pelo sucesso da internet, com um simples protocolo[8] é possível trocar arquivos como áudio e vídeo[10].

No entanto, com o aumento das conexões e dos compartilhamentos a segurança passou a ser muito importante. Atualmente, os sistemas proporcionam uma pequena proteção contra escutas escondidas ou intrusos na comunicação[10].

2.1.1 Comunicação Segura

Comunicação segura está diretamente relacionada neste projeto com os softwares de mensageria. Um mensageiro instantâneo ou comunicador instantâneo, conhecido nos Estados Unidos por IM (Instant Messaging), é uma aplicação que permite o envio e recebimento de mensagens em tempo real. Com a utilização destes programas o usuário é informado quando algum de seus contatos adicionados em sua lista de contatos está on-line[9]. A partir deste momento podem

---

8 Protocolo é o termo usado para um conjunto de informações ou dados que passam por um preparo para serem repassados a outros programas[6].

9 On-line a tradução literal para o português é "na linha", mas com o significado mais de claro de "ao vivo", "conectado" ou "ligado"[6].

manter conversas através de mensagens de texto as quais são recebidas pelo destinatário instantaneamente. Geralmente esses programas incorporam diversos outros recursos, como por exemplo: envio de figuras ou imagens animadas, conversação com áudio, utilizando-se das caixas de som e microfones do sistema, além de vídeo conferência, ou seja, utilizando de webcam[10].

Um dos pioneiros foi o ICQ, software que rapidamente alcançou sucesso em todo o mundo e abriu caminho para o desenvolvimento de diversos outros protocolos e aplicações por parte de outras companhias. Um IM sempre está associado a um serviço de mensagens instantâneas, esse serviço difere do e-mail na medida em que as conversações ocorrem em tempo real[11].

Mas estes programas não devem ser considerados imunes à monitoração por terceiros a menos que utilizem programas especiais que codifiquem (utilizando métodos de criptografia[11]) entre os dados transmitidos por transmissor e receptor e vice-versa[11].

A utilização de comunicadores vem aumentando muito nos últimos anos, se alastrando dentro das empresas. Isso vem gerando uma série de problemas para as empresas. Algumas empresas começaram a adotar o uso de comunicadores coorporativos, também chamado de messenger coorporativo. A diferença entre esses comunicadores e os comunicadores populares, está no controle. O usuário não possui autorização para adicionar contatos e toda a conversa é logada para posterior análise[11].

2.1.2 Problemas dos IM

Seguem os problemas de segurança que o uso dos IM existentes no mercado apresentam[3]:

- Risco de escapar informações confidenciais[3]:

- Os canais de comunicação criados pelo IM não conseguem ser filtrados pelo firewall[12] das empresas, permitindo que terceiros localizados fora da rede corporativa possam acessar as informações trafegadas[3].

- Exposição legal[3]:

---

10 Webcam é uma câmera de vídeo de baixo custo que capta imagens, transferindo-as de modo quase instantâneo para o computador[12].

11 Criptografia é um estudo dos princípios e técnicas pelas quais a informação pode ser transformada da sua forma original para outra ilegível, de forma que possa ser conhecida apenas por seu destinatário (detentor da "chave secreta"), o que torna difícil de ser lida por alguém não autorizado. Assim sendo, só o receptor da mensagem pode ler a informação com facilidade[13].

12 Firewall é o nome dado a um dispositivo de uma rede de computadores que tem por objetivo aplicar uma política de segurança a determinado ponto de controle da rede. Sua função consiste em regular o tráfego de dados entre redes distintas e impedir a transmissão e/ou recepção de acessos nocivos ou não autorizados de uma rede para outra[14].

- As informações trocadas podem ser armazenadas localmente na máquina de um usuário ou até mesmo em um servidor e esse conteúdo poderá ser revelado em uma investigação pessoal ou até mesmo em ataques maliciosos, levando a exposição da privacidade dos usuários destes softwares de IM[3].

- Entrada de vírus[3]:

- A utilização de IM é uma porta de entrada para vírus que podem prejudicar tanto o desempenho do seu computador e de uma rede de computadores[3].

Segundo um levantamento da AKONIX os ataques com códigos maliciosos em redes de mensagens instantâneas, cresceram 78% em relação a 2006[15].

Os códigos utilizados nos ataques eram novos ou variados, detectados pelo IM Security Center da AKONIX, em parceria com seus usuários e outras empresas de comunicadores[15].

2.2 Principais Softwares de IM Disponibilizados Atualmente no Mercado

O MSN Messenger é um programa de mensagens instantâneas criado pela Microsoft Corporation. O programa permite que um usuário da Internet se relacione com outro que tenha o mesmo programa em tempo real, podendo ter uma lista de amigos “virtuais” e acompanhar quando eles entram e saem da rede. Ele foi fundido com o Windows Messenger e originou o Windows Live Messenger[16].

O sucesso do MSN Messenger pode ser explicado por ele ser integrado ao serviço de e-mail do Hotmail, por ser incluso com o Windows XP e por ter uma intensa publicidade junto ao público jovem[16].

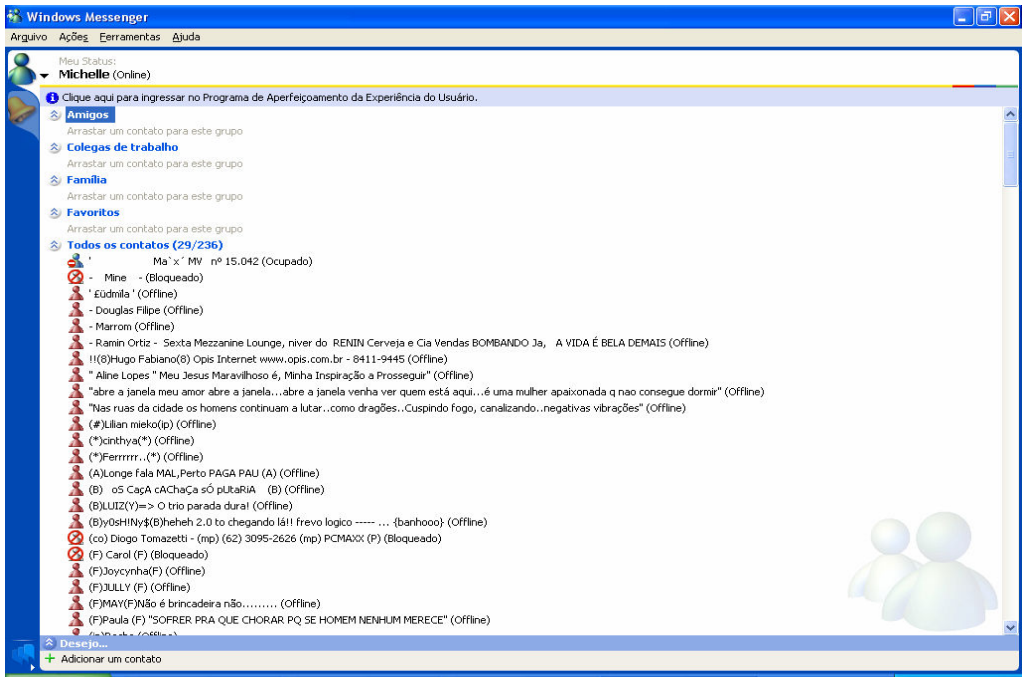

Figura 2.1 – Login no Windows Messenger
(Michelle Ribeiro Abuchahin em 20/09/2008).

O Windows Live Messenger é a nova geração do MSN Messenger o novo programa introduz novos recursos além de incluir os já existentes no MSN Messenger. O Windows Live Messenger surgiu depois da proposta da Microsoft em reunir os serviços do MSN ao sistema operacional Windows[17].

Dentre as várias modificações os novos recursos adicionados foram[17]:

- Troca de mensagens com o status off-line[13];
- A última imagem de exibição selecionada será armazenada remotamente, é a mesma poderá ser acessada em outro computador[17].
- Busca rápida de contatos[17].
- Possibilidades de mostrar as fotos dos contatos on-line e off-line nas grades das janelas de contatos dentre outras[17].

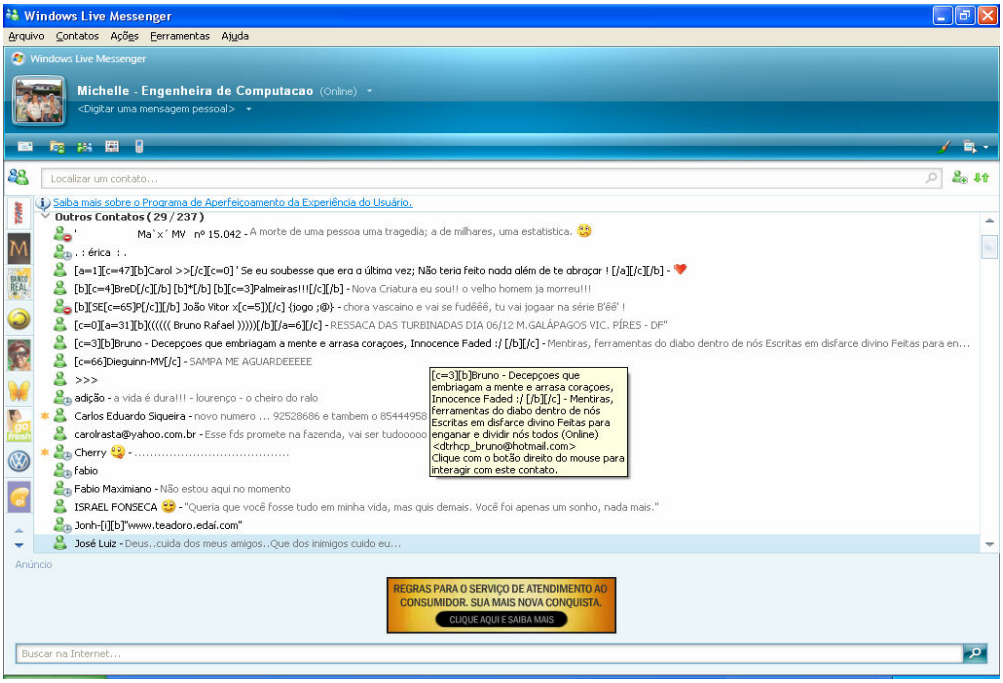

Figura 2.2 – Login no Windows Live Messenger
(Michelle Ribeiro Abuchahin em 20/09/2008).

O ICQ é um programa de comunicação instantânea pela Internet. A sigla é um de um acrónimo[14] feito baseado na pronúncia das letras em Inglês (I See You), em português, “Eu procuro você”. O ICQ foi o pioneiro desta tecnologia tendo sua primeira versão lançada em 1997 por uma empresa israelita, chamada Mirabilis[19].

Em 1999, a Time Warner AOL, adquiriu a Mirabilis englobando o serviço. Hoje a Mirabilis se chama ICQ.LLC, ligada por nome, porém independente dos serviços da AOL[19].

13 Off-line traduz-se na indisponibilidade da entidade perante ao sistema[6].
14 Acrónimo é uma palavra formada pelas letras ou sílabas iniciais de palavras sucessivas de uma locução, ou pela maioria destas partes[18].

Apesar de sua despopularização em alguns países, o ICQ continua ativo com uma equipe apresentando novas versões regularmente e com opções inovadoras como troca de SMS de telefones celulares, VOIP, além de uma melhoria significativa na aparência e novos sons criados pela dupla de DJ's Infected Mushroom[19].

A partir de 2005, novas empresas reforçam a estratégia do ICQ, redistribuindo usuários customizados de acordo com as características de seus respectivos países[19].

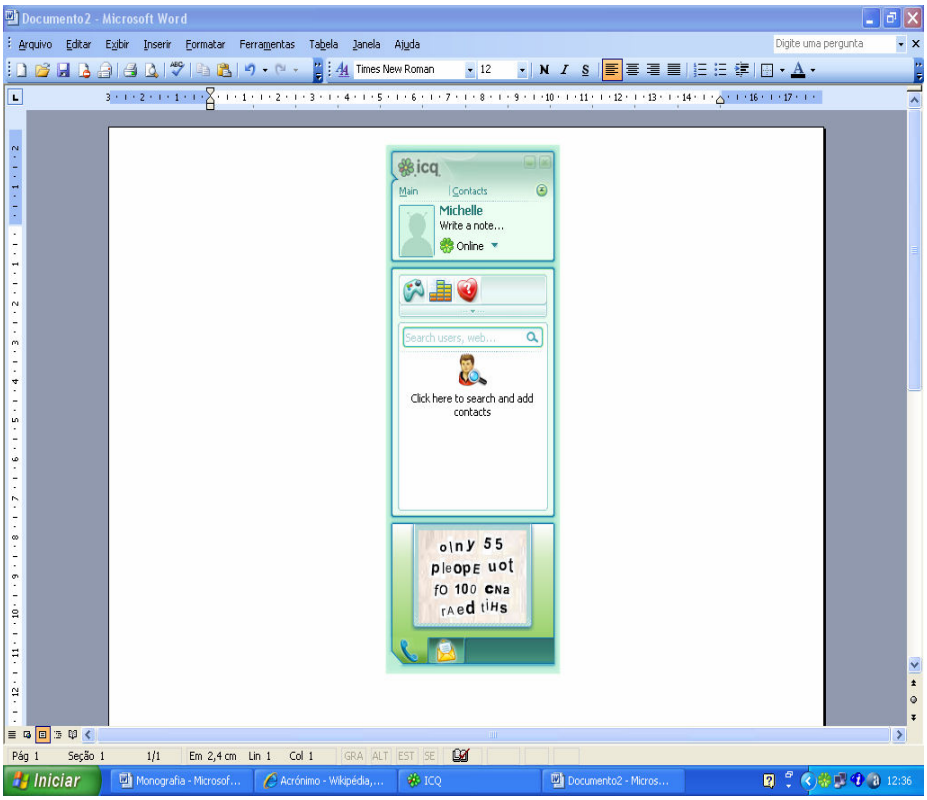

Figura 2.3 – Login no ICQ
(Michelle Ribeiro Abuchahin em 20/09/2008).

O Yahoo! Messenger é um programa de software através do qual pessoas podem conversar, se escrever e se ver à distância. O software é da empresa Yahoo! e possui recursos com a integração Flick[15] e player de mídia, que exibe imagens e reproduz vídeos de sites como o YouTube[16] quando um link é enviado a outro usuário[20].

Existe também o Yahoo! Messenger para Web que é um navegador que não precisa ser instalado, já que funciona diretamente no navegador, parecido com o MSN Web Messenger[20].

15 Flick é um site da web de hospedagem e partilha de imagens fotográficas.
16 YouTube é um site na internet que permite que os usuários carreguem, assistam e compartilhem vídeos em formato digital[21].

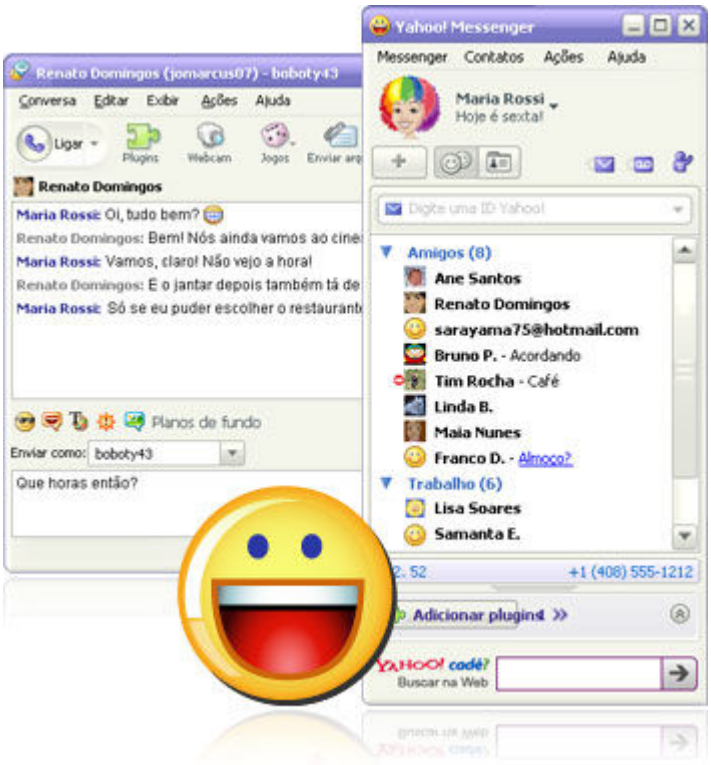

Figura 2.4 – Login no Yahoo! Messenger
(http://br.messenger.yahoo.com, acesso em 20/09/2008).

2.3 Malwares[17] em IM

A troca de mensagens instantâneas é um dos meios mais fácies de se distribuir aplicativos maliciosos. Serão apresentados a seguir alguns tipos de malwares.

2.3.1 Vírus

Um vírus é um código de computador que se anexa a um programa ou em um arquivo para poder se espalhar entre os outros computadores, infectando-os à medida que se desloca. Os vírus podem danificar seu software, hardware e arquivos[22].

Se uma pessoa escreve um vírus, provavelmente em linguagem de montagem, ele então o insere cuidadosamente em um programa de sua própria máquina, utilizando uma ferramenta chamada droper[18]. Então este programa é distribuído geralmente em uma coleção de softwares grátis na Internet. Uma vez instalado na máquina da vítima, os vírus ficam aguardando que o programa infectado seja executado.

17 Malware é proveniente do inglês malicious software, é um software destinado a se infiltrar em um sistema de computador alheio de forma ilícita, com o intuito de causar algum dano ou roubo de informações (confidenciais ou não)[7].
18 Droper é um cavalo de Tróia que grava outros cavalos de tróia[22].

Ao ser iniciado, ele começa a infectar outros programas da máquina e então dispara o payload[19][22].

O que geralmente acontece em troca de mensagens instantâneas, é a troca de arquivos on-line, se o arquivo estiver com o vírus no momento em que o receptor executá-lo estará automaticamente instalando um vírus em sua máquina, uma vez que os arquivos trafegados não são verificados ou conseguem burlar o sistema de segurança utilizado na máquina, ou seja, conseguem facilmente atravessar os antivírus[20] e firewalls.

2.3.2 Worms

Assim como os vírus, são capazes de criar cópias de si mesmo, sendo que essa operação ocorre automaticamente. Primeiramente ele controla os recursos no computador que permite o seu transporte de arquivos e informações. Depois que ele contamina o sistema, ele se desloca sozinho[25].

Um worm pode enviar cópias de si mesmo para todas as pessoas que estão em sua lista de contatos em um sistema de mensagem instantânea e os computadores dos seus “amigos” farão o mesmo, causando um dominó de alto tráfego na rede que pode tornar mais lentas tanto as redes coorporativas assim como a Internet. O seu poder de alastramento é grandioso, fazendo com que as redes sejam obstruídas.

2.3.3 Cavalo de Tróia

Assim como o mitológico cavalo de Tróia parecia ser um presente, mas na verdade escondia soldados gregos em seu interior e tomaram à cidade de Tróia, os cavalos de Tróia da atualidade são programas de computadores que parecem ser úteis, mas só que na verdade comprometem a segurança e causam inúmeros danos. Os cavalos de tróia se alastram quando as pessoas são seduzidas a abrir um programa por pensarem ser de uma fonte legítima[25].

Não é seguro baixar arquivos de mensagens instantâneas de usuários que não estão cadastrados em listas de contatos, pois certamente nestas mensagens estará anexo um malware que irá comprometer o desempenho do seu computador.

---

19 Payload refere-se ao dado real sendo transmitido, ele é seguido por um cabeçalho que identifica o transmissor e o receptor do dado sendo transportado e é logo descartado assim que chega ao destinatário[23].
20 Antivírus são softwares projetados para detectar e eliminar vírus de computador[24].

2.4 Criptografia

O envio e recebimento de informações sigilosas é uma necessidade antiga, existente há centenas de anos. Com o surgimento da internet e sua facilidade de oferecer informações de maneira precisa e extremamente rápida, a criptografia tornou-se uma ferramenta fundamental de modo a permitir que apenas o emissor e receptor tenham livre acesso à informação trabalhada[26].

O termo Criptografia surgiu da fusão das palavras gregas “Kryptós” e “gráphein” que significam respectivamente “oculto” e “escrever”. Refere-se a um conjunto de conceitos e técnicas que visam codificar uma informação de forma que somente o emissor e o receptor possam acessá-la, evitando que um intruso consiga interpretá-la[26].

Muitas técnicas sugiram com o passar dos anos e as mais conhecidas utilizam-se do conceito de chaves, as chamadas “chaves criptográficas”, que são um conjunto de bits[21] baseado em um determinado algoritmo capaz de codificar e decodificar informações. Se o receptor da mensagem usar uma chave incompatível com a chave do emissor não conseguirá acessar a informação[26].

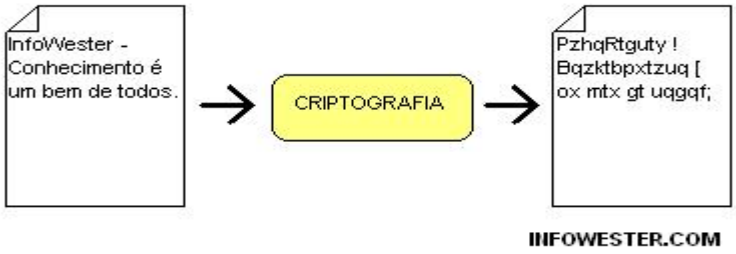

Figura 2.5 – Exemplo de Criptografia
(http://www.infowester.com/criptografia.php, acesso em 23/09/2008).

Com a utilização de chaves um emissor pode usar o mesmo algoritmo para vários receptores. Basta que cada um receba uma chave diferente. Além disso, caso um receptor perca ou exponha determinada chave é possível trocá-la mantendo-se o mesmo algoritmo[26].

Quanto mais bits forem utilizados mais segura será a criptografia. Caso um algoritmo use chaves de 8 bits, apenas 256 chaves poderão ser utilizadas na decodificação, pois 2 elevado 8 é 256. Esse exemplo, deixa claro que a utilização de 8 bits é inseguro, pois se até uma pessoa pode simular 256 combinações imagine um computador. Porém se forem utilizados 128 ou mais bits para chaves, teremos uma imensa quantidade de combinações, pois imagine o resultado de 2 elevado a 128[26].

21 Bits é a menor unidade de medida de transmissão de dados usada na computação[27].

Existem dois tipos de chaves: simétricas e assimétricas.

As chaves simétricas tratam-se de um tipo de chave aonde o emissor e o receptor fazem uso da mesma chave, ou seja, uma única chave é usada na codificação e decodificação da informação. Existem vários algoritmos que utilizam chaves simétricas[26]:

*"**DES (Data Encryption Standard):** criado pela IBM em 1977, faz uso de chaves de 56 bits. Isso corresponde a 72 quadrilhões de combinações. É um valor absurdamente alto, mas não para um computador potente. Em 1997, ele foi quebrado por técnicas de "força bruta" (tentativa e erro) em um desafio promovido na internet"[26];*

***"IDEA (International Data Encryption Algorithm):** criado em 1991 por James Massey e Xuejia Lai, o IDEA é um algoritmo que faz uso de chaves de 128 bits e que tem uma estrutura semelhante ao DES. Sua implementação em software é mais fácil do que a implementação deste último"[26];*

***"RC (Ron's Code ou Rivest Cipher):** criado por Ron Rivest na empresa RSA Data Security, esse algoritmo é muito utilizado em e-mails e faz uso de chaves que vão de 8 a 1024 bits. Possui várias versões: RC2, RC4, RC5 e RC6. Essencialmente, cada versão difere da outra por trabalhar com chaves maiores"[26].*

A utilização de chaves simétricas tem muitas desvantagens, pois primeiramente é necessário usar uma grande quantidade de chaves caso várias pessoas estejam envolvidas no processo além disto tanto o emissor quanto o receptor precisam conhecer a chave utilizada, durante a transmissão a chave pode cair em mãos de pessoas mal intencionadas[26].

Já as chaves assimétricas conhecidas como "chaves públicas" trabalham com duas chaves: uma é denominada privada e a outra é denominada chave pública. Nesse método, uma pessoa deverá criar uma chave de codificação e enviá-la para quem for mandar as informações a ela, esta é a chave pública, e a chave privada é secreta[26].

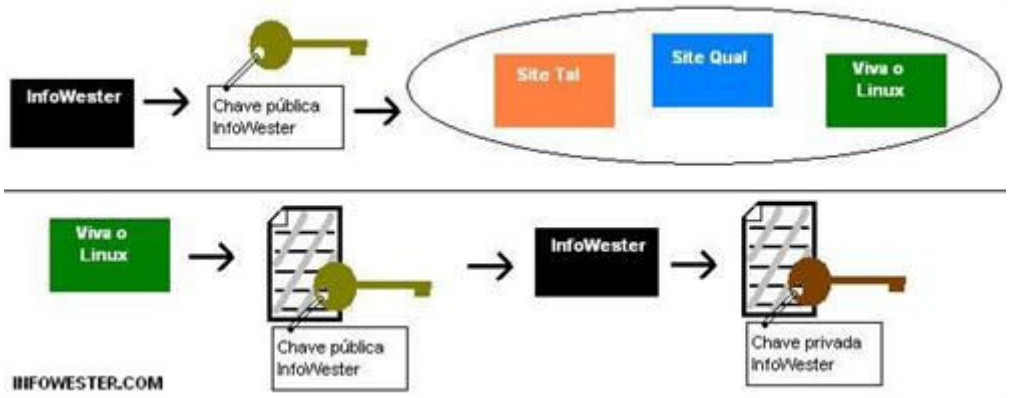

Figura 2.6 – Exemplo de Chave Assimétrica
(http://www.infowester.com/criptografia.php, acesso em 23/09/2008).

Entre os algoritmos que utilizam chave assimétrica estão:

> **Para compreender o algoritmo de tem-se que: RSA (Rivest, Shamir and Adleman):** criado em 1977 por Ron Rivest, Adi Shamir e Len Adleman nos laboratórios do MIT (Massachusetts Institute of Technology), é um dos algoritmos de chave assimétrica mais usados. Nesse algoritmo, números primos (número primo é aquele que só pode ser dividido por 1 e por ele mesmo) são utilizados da seguinte forma: dois números primos são multiplicados para se obter um terceiro valor. Porém, descobrir os dois primeiros números a partir do terceiro (ou seja, fazer uma fatoração) é muito trabalhoso. Se dois números primos grandes (realmente grandes) forem usados na multiplicação, será necessário usar muito processamento para descobrí-los, tornando essa tarefa quase sempre inviável. Basicamente, a chave privada no RSA são os números multiplicados e a chave pública é o valor obtido[26].

"ElGamal: *criado por Taher ElGamal, esse algoritmo faz uso de um problema matemático conhecido por "logaritmo discreto" para se tornar seguro. Sua utilização é freqüente em assinaturas digitais"[26].*

Um recurso conhecido por Assinatura Digital é muito utilizado nas chaves públicas, pois permite provar que um determinado documento eletrônico é de uma procedência verdadeira. O receptor da informação utilizará a chave pública fornecida pelo emissor para se certificar da origem. Além disto, a chave fica integrada ao documento de forma que qualquer alteração por terceiros imediatamente a invalide[26].

Ao utilizar criptografia é necessário a garantia de três princípios básicos: confidencialidade, autenticação e integridade. No próximo capítulo é apresentado a importância da autenticidade neste projeto.

Capítulo 3 – Segurança em IM

Este capítulo mostra a segurança em mensagem instantânea, como ocorre sua estrutura, dentre os quais os malefícios e benefícios.

3.1 Riscos em IM

A troca de mensagens instantâneas se tornou imprescindível para milhares de usuários, sistemas como os expostos acima: Windows Messenger, ICQ e o Yahoo! Messenger transformaram a forma como as pessoas se comunicam com amigos, parentes e colegas de trabalho. De acordo com analistas de mercado, espera-se que o número de usuários coorporativos de Mensagens Instantâneas cresça ainda mais, chegando a até 300 milhões em 2005[28].

Mais a maioria dos sistemas utilizados atualmente foram criados tendo como foco o dimensionamento e não a segurança. Segundo Carey Nachenberg, Diretor Arquiteto da Symantec Advanced Concepts Group escreveu em 21/05/2002: *"Literalmente nenhum sistema de Mensagens Instantâneas possui recursos de criptografia, e a maioria possui recursos para ignorar firewalls corporativos, tornando difícil para administradores controlar seu uso dentro da empresa. A maioria desses sistemas possui um gerenciamento de senhas inseguro e está vulnerável a defraudações de contas, além de potenciais ataques de negação de serviços".*

Quando nos referimos à segurança temos que nos atentar a três objetivos gerais, conforme a tabela 3.1:

Tabela 3.1 – Objetivos e Ameaças à Segurança
(TANENBAUM, Andrew S. Sistemas Operacionais Modernos).

Objetivo	Ameaça
Confidencialidade dos dados	Exposição dos dados
Integridade dos dados	Adulteração dos dados
Disponibilidade do sistema	Recusa de serviço

O primeiro objetivo que é a confidencialidade dos dados, ou seja, manter secretamente os dados. O proprietário dos dados deverá decidir para quem os dados deverão ser disponibilizados e sendo assim, ao tomar esta decisão o sistema deverá garantir que não ocorrerá a liberação dos dados para as pessoas que não foram escolhidas[27].

O segundo objetivo que é a integridade dos dados, ou seja, usuários não autorizados não serão capazes de modificar quaisquer dados sem a permissão do proprietário. O que significa que está sendo referenciado tanto a alteração de dados bem como a remoção e inclusão de dados falsos[27].

O terceiro objetivo que á disponibilidade do sistema, ou seja, ninguém poderá perturbar o sistema e deixá-lo inutilizável[27].

Os mensageiros instantâneos tornaram-se extremamente populares, mas a sua utilização em massa pode trazer riscos aos usuários. Em 2002, os usuários do MSN Messenger que recebiam a sugestão para acessarem uma página da Web em inglês, poderiam estar sendo alvos de um novo vírus. Batizado como Coolnow, Menger, JS.Menger, JS/Coolnow, JS/Exploit-Messenger e outros, o vírus estava contido em páginas com códigos JavaScript[22], capazes de acessar toda a lista de contatos, sendo assim o vírus enviava o link para toda a lista dando continuidade ao processo[30].

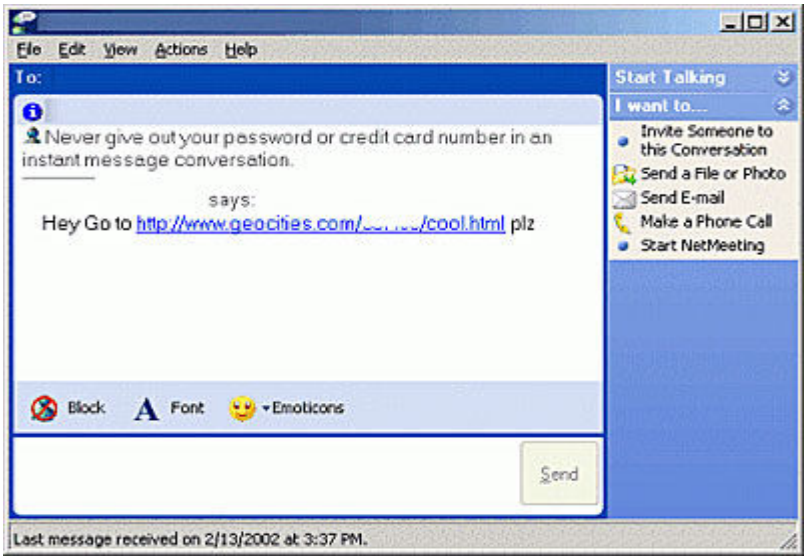

Figura 3.1 – Vírus no MSN Messenger em 2002
(http://www.terra.com.br/informatica/2002/02/14/006.htm, acesso em 17/10/2008).

Já em 2005, o comunicador MSN Messenger foram ameaçados por dois vírus: o Fatso.A e o Kelvin.B, os vírus foram responsáveis por infectarem todos os usuários on-line. Ambos, enviavam através do sistema P2P o link de uma página na internet onde há uma cópia de códigos maliciosos. Ao clicar neste link, o usuário faz o download do vírus e infectar o computador, conforme imagem abaixo[32]:

22 JavaScript é uma linguagem de programação criada pela Netscape em 1995, que a principio se chamava LiveScript para atender as seguintes necessidades:
Validação de formulários no lado usuário (programa navegador) e interação com a página[31].

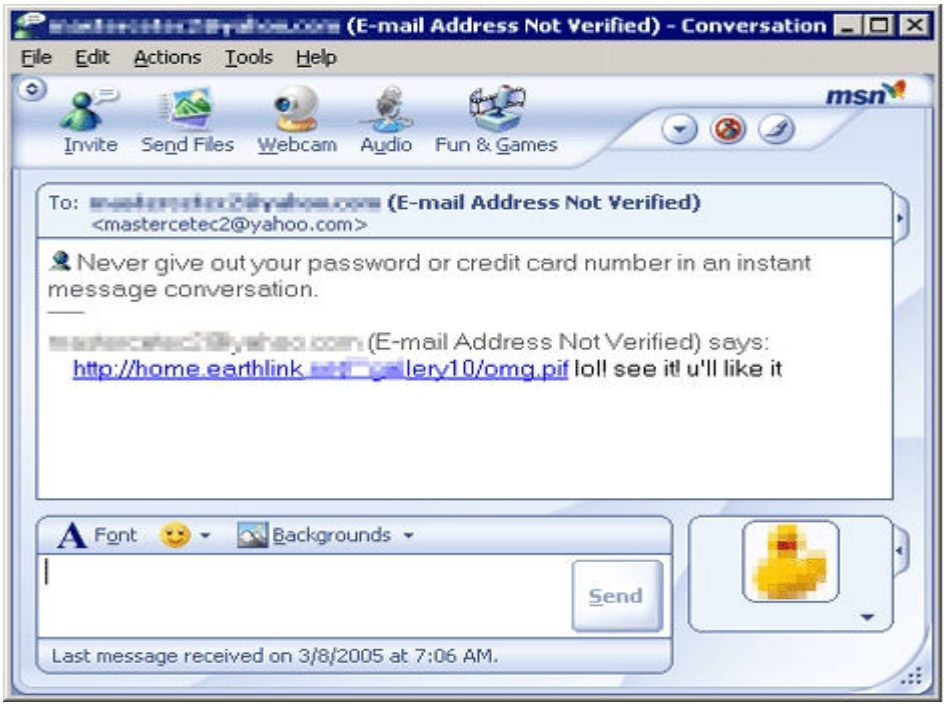

Figura 3.2 – Vírus no MSN Messenger em 2005 (http://www.trendmicro.com/br/about/news/pr/archive/2005/pr060205.htm, acesso em 17/10/2008).

E em 2007, um novo cavalo de tróia, começou a se espalhar via o Windows Live Messenger, os usuários recebiam uma mensagem com um arquivo ZIP que contém um arquivo malicioso[33].

3.2 Vulnerabilidades em IM

Os comunicadores instantâneos públicos mais conhecidos e utilizados na internet são o Microsoft Messenger, Google Talk, Yahoo! Messenger e ICQ. Apesar de serem utilizados em massa é notável que a maioria dos usuários não estejam ciente das vulnerabilidades destes programas. Em pesquisa realizada pelo fórum pcs, 57% dos usuários utilizam o MSN Messenger[34].

Muitas das coisas que tornam as mensagens instantâneas úteis também fazem delas um risco. Por exemplo, a maioria das ferramentas de mensagem instantânea oferecem um método para enviar e receber arquivos. Embora útil, esse recurso também é um ponto principal de vulnerabilidade. Os anexos da mensagem instantânea, assim como os anexos de e-mail, podem transmitir vírus destrutivos, cavalos de tróia e worms. Há também um novo tipo de worm de envio de e-mails em grande escala, que usa os recursos mais básicos das mensagens instantâneas para causar estragos. Das muitas maneiras que algumas ameaças combinadas usam os mecanismos de e-mail e catálogos de endereços para se propagar, esses worms usam seu software de mensagem instantânea para se distribuir entre todos os

membros de sua lista de amigos. Para seus amigos e colegas, eles aparecem como se estivessem recebendo uma mensagem sua. Na realidade, a mensagem é gerada por um worm e, em alguns casos, ela pode conter um link para um site da web que automaticamente faz o download de outro pedaço do código malicioso. É, definitivamente, uma falha muito grave de segurança[34].

Devido à facilidade de criar uma identidade de mensageiro instantâneo e entrar em contato com as pessoas, a mensagem instantânea é um meio perfeito para realizar fraudes online, roubo de identidade e outros ataques. Indivíduos mal-intencionados podem usar todos os tipos de métodos, incluindo invasões de contas e personificação de usuários legítimos, para ganhar a confiança e obter informações de usuários de mensagens instantâneas ingênuos. Elas são um perigo em particular para as crianças, que podem ser abordadas por estranhos, com intenções ilegais. Mesmo se não estiverem procurando obter sua identidade, alguns impostores podem simplesmente enganar você, e anotando suas informações particulares. Por último, há o spim[23]. O spim, nome dado ao spam enviado em uma mensagem instantânea está em ascensão, e é mais do que apenas um problema inconveniente e demorado. Alguns spims podem conter linguagem ofensiva ou links para sites na web com conteúdo impróprio para menores[34].

3.3 Dicas Para Proteção

Acima foram expostos algumas das ameaças na utilização de mensagens instantâneas, seguem algumas dicas, afinal a grande maioria utilizam estes softwares:

1. Escolha apelidos, não utilize nomes e e-mails verdadeiros e nem quais outras informações que permitam sua identificação pessoal[35];

2. Fale apenas com as pessoas que estão na sua Lista de Contatos, para evitar o spim e configure suas mensagens instantâneas para bloquear mensagens de pessoas desconhecidas[35];

3. Pense em criar um e-mail secundário para cadastramento nos serviços de mensagens, pois alguns serviços associam o seu apelido com o seu endereço de e-mail[35];

4. Nunca divulgue senhas ou números de contas por mensagem instantânea, pois estas informações poderão ser acessadas por intrusos[35];

5. Configure corretamente para que o serviço de mensagem instantânea não abra automaticamente quando o computador é ligado[35];

---

23 Spim é spam em mensagens instantâneas, trata-se de uma mensagem eletrônica não-solicitada enviada em massa[36].

6. Faça corretamente as atualizações para manter seu sistema operacional protegido[35];

7. Configure o antivírus para examinar automaticamente todos os arquivos anexos de mensagens instantâneas recebidos[35];

8. Cuidado ao compartilhar arquivos, pois os cavalos de tróia ficam ocultos, evite arquivos com extensões .exe, .lnk, .bat, .dll, .bin e .cmd[35];

9. Atualize constantemente seu antivírus[35];

10. Controle a utilização de mensagens instantâneas pelos seus filhos, instale softwares que monitorem as informações enviadas e recebidas pelos seus filhos[35].

Além das dicas expostas acima é necessário que estes aplicativos cada vez mais se utilizem de soluções baseadas em segurança tanto e hardware como em software. No próximo capítulo é apresentado a aplicação com segurança em mensagem instantânea.

Capítulo 4 – Descrição do Projeto

Este capítulo mostra como foi desenvolvido o software e suas funcionalidades.

4.1 Topologia

A topologia utilizada é estrela e seguirá a seguinte estrutura conforme figura:

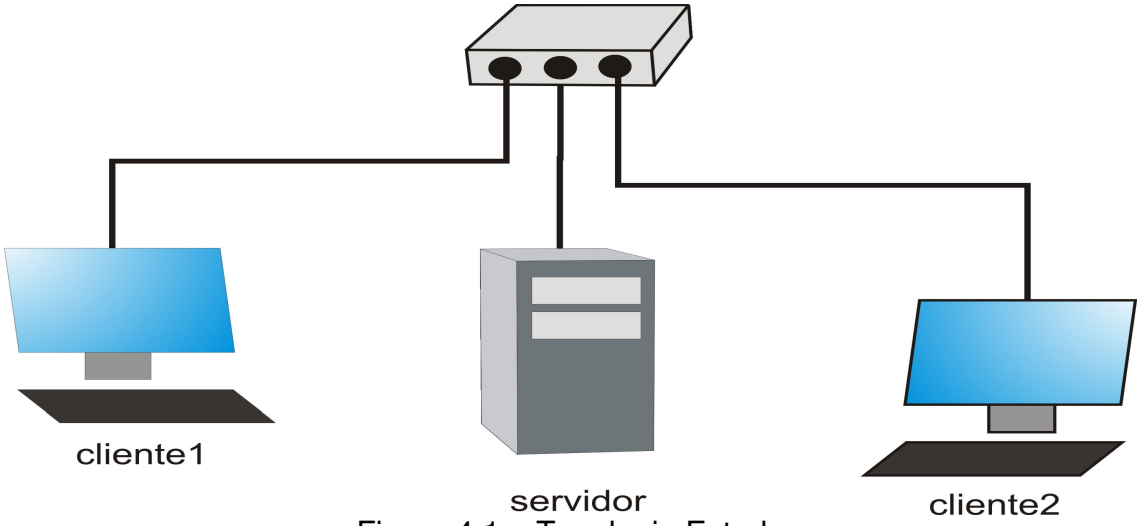

Figura 4.1 – Topologia Estrela
(Michelle Ribeiro Abuchahin em 31/10/2008).

Duas máquinas representadas pelo “Usuário 1” e “Usuário 2” estão conectadas a uma terceira máquina representada pelo servidor, o aparelho para conectar as três máquinas é um hub[24], além disto as máquinas estão ligadas em rede por um cabo par trançado[25]. Esta é a opção de conexão para demonstração do projeto.

Figura 4.2 – Cabo Par Trançado
(Michelle Ribeiro Abuchahin em 31/10/2008).

24 Hub ou concentrador em linguagem de informática é o aparelho que interliga diversas máquinas (computadores) que pode ligar externamente redes TAN, LAN, MAN e WAN[8].
25 Cabo par trançado é um tipo de cabo no qual dois fios são entrançados um ao redor do outro para cancelar interferências eletromagnéticas de fontes externas e interferências mútuas em cabos vizinhos. É o tipo de cabo mais utilizado para interligar computadores em rede[8].

4.2 Especificações do Projeto

Objetivando a demonstração deste projeto é utilizado um programa desenvolvido em Visual Basic cujo nome é MiConversa, Mi por serem as duas primeiras iniciais do meu nome(Michelle) e Conversa por ser um programa com finalidade acadêmica para demonstração de conversação entre duas pessoas que estão em computadores distintos.

A escolha da linguagem de programação foi baseada no fato do Visual Basic ser uma linguagem orientada a objeto, o que deu um suporte na interface gráfica do projeto, além da sua interação com o Windows.

O MiConversa foi desenvolvido para utilização acadêmica e quem estiver interessado em utilizá-lo torna-se necessário a aquisição do Visual Basic 6.0, pois para a utilização do mesmo é necessário o serial contido na caixa que acompanha o cd de instalação.

O processo de comunicação remota que acontece entre os comunicadores instantâneos é desenvolvido em socket o que permite a comunicação das funções entre máquinas diferentes e interligadas numa rede de computadores, o socket é representado pela função winsock, ou seja, as máquinas irão se comunicar falando a mesma “língua” conectadas em rede.

O MiConversa foi testado e desenvolvido apenas em sistemas operacionais rodando na plataforma Windows. Para a utilização em outras plataformas seria necessário instalar outra máquina virtual correspondente a cada tipo de sistema operacional e testá-lo.

4.3 Interface MiConversa

Para a utilização do programa se faz necessário a abertura de dois executáveis, um correspondente ao servidor e o outro correspondente a troca de mensagens.

O servidor será o responsável pela demonstração de todo o processo de conversação bem como de liberação de chaves aleatórias para cada uma das máquinas envolvidas no processo de conversação.

Já os usuários serão responsáveis pela conversação em si, nestas telas é aonde ocorre toda a conversação e demonstração de como estas mensagens podem ser transmitidas criptografadas e recepcionadas descriptografadas.

A compilação do MiConversa gerou dois executáveis Servidor MiConversa.exe para o servidor e MiConversa.exe para o usuário.

A interface inicial do MiConversa é igual para todos os usuários que estão no processo de comunicação, para acesso será criado um nome de usuário e senha no Servidor MiConversa.exe.

A tela inicial do Servidor MiConversa mostra o que está acontecendo no processo de comunicação, quem solicitou a conexão, sucesso e erro, permissão de usuários e o trânsito de mensagens na forma em que está sendo trafegada na rede, ou seja, criptografada.

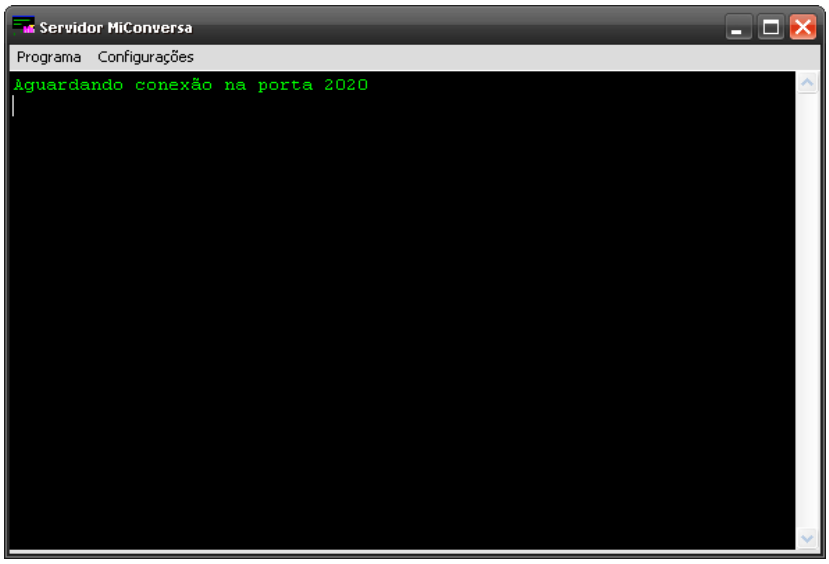

Figura 4.3 – Servidor MiConversa
(Michelle Ribeiro Abuchahin em 31/10/2008).

Nesta tela tem como default a Porta para entrada 2020 e a Porta para saída 1010, o IP Saída é preenchido manualmente, para quem utilizar o software é necessário o preenchimento de dois campos: usuário e senha com confirmação da mesma.

O botão de + é utilizado para cadastrar usuário e senha e o botão de – é utilizado para remover.

A senha escolhida bem como o nome de usuário é salva num arquivo chamado usuarios.txt, e a senha passa por um processo de criptografia com a utilização de MD5. MD5 é um algortimo de hash[26] e 128 bits unidirecional desenvolvido pela RSA Data Security, muito utilizado por sofwatres com protocolo ponto-a-ponto[37].

Utilizar o MD5 garantiu ao projeto segurança para as senhas cadastradas no banco de dados, garantia de que estas senhas não seriam descobertas por instrusos, pois a senha não fica legível e uma vez que passou pelo MD5 a informação não pode ser recuperada.

26 Hash é uma seqüência de bits geradas por um algoritmo de dispersão, em geral gerada em base hexadecimal que permite a visualização em letras de (A a F) e números, representando 2 bytes cada[13].

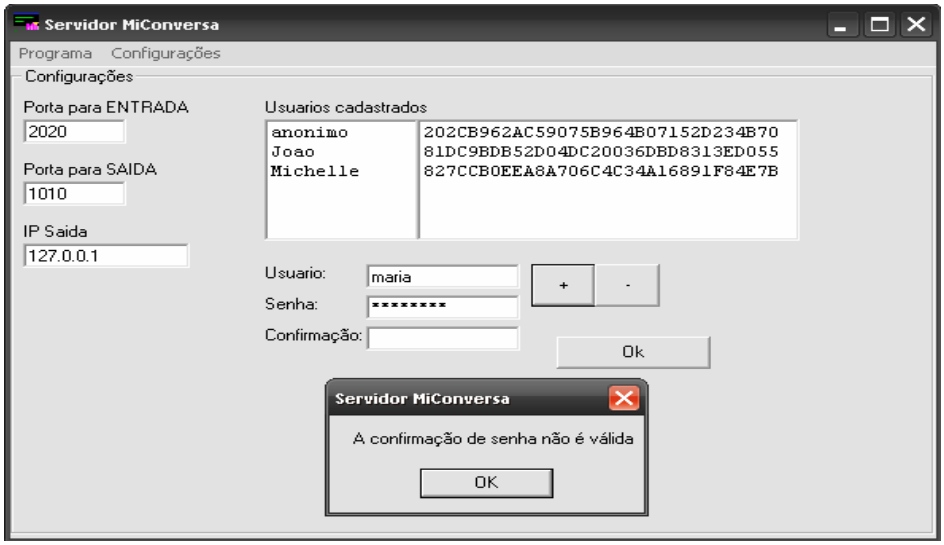

Figura 4.4 – Configuração no Servidor MiConversa
(Michelle Ribeiro Abuchahin em 31/10/2008).

Cada par usuário e senha é salvo em um arquivo de texto chamado usuarios.txt, onde os 32 primeiros caracteres referem-se ao hash da senha e os caracteres restantes ao nome de usuário.

Figura 4.5 – Usuários.txt
(Michelle Ribeiro Abuchahin em 31/10/2008).

Abaixo, temos a tela principal do MiConversa nesta tela é realizado todo o processo de comunicação. Nesta tela o usuário poderá digitar qualquer texto e o mesmo será criptografado imediatamente e enviado para o servidor ao clicar no botão Enviar.

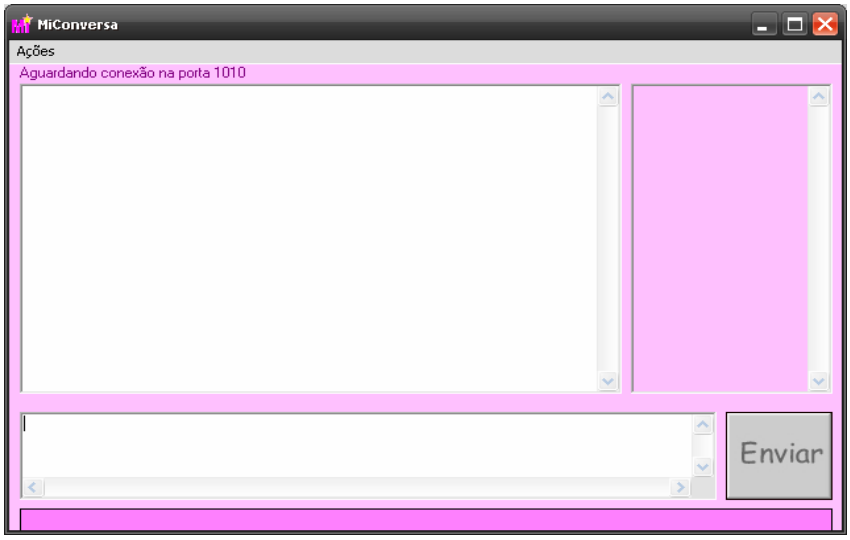

Figura 4.6 – Tela Principal MiConversa
(Michelle Ribeiro Abuchahin em 31/10/2008).

Na tela de Configurações é informado o nome de usuário e senha que já está cadastrado no Servidor MiConversa nesta mesma tela é configurado o IP do servidor para requisição de conexão e a Porta saida e Porta entrada. Ou seja, Porta saida porta de solicitação de conexão com o servidor e Porta entrada porta de espera para a conexão.

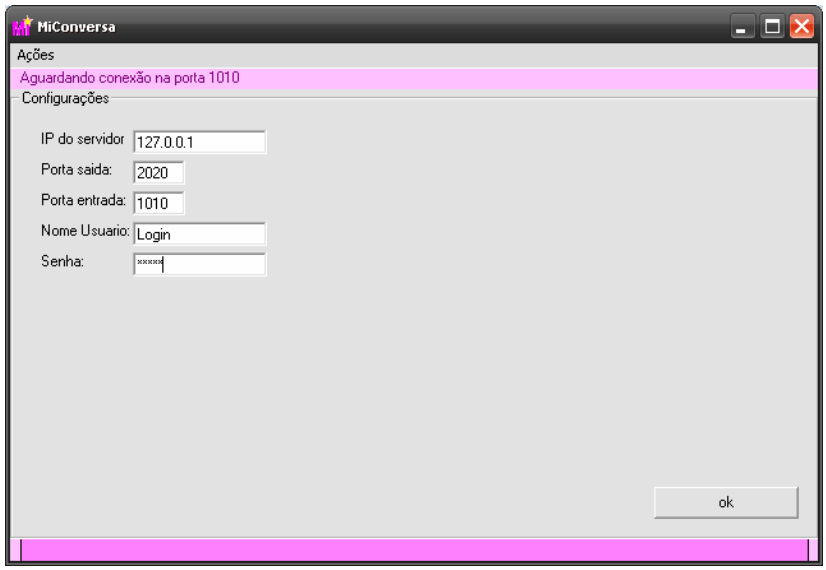

Figura 4.7 – Função Configurar em MiConversa
(Michelle Ribeiro Abuchahin em 31/10/2008).

Na confecção do software foi utilizada uma chave de 32 caracteres de 0 a 9 e de A a F, com a intenção de parecer um hash de 128 bits, gerada aleatoriamente e enviada a ambos os usuários pelo servidor, utilizando-se do seguinte algoritmo.

4.4 Metodologia de Chaves e Criptografia em MiConversa

O algoritmo de geração de chave aleatória segue o seguinte esquema:

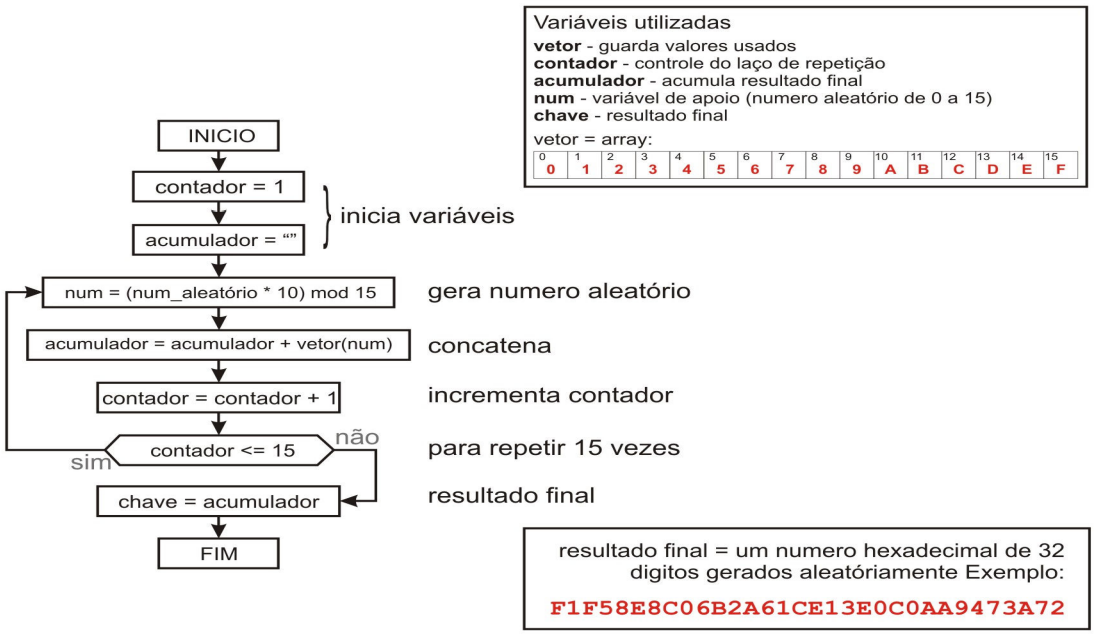

Figura 4.8 – Algoritmo de Criação de Chave Aleatória
(Michelle Ribeiro Abuchahin em 07/11/2008).

Após a geração de chave aleatória pelo servidor, o mesmo envia as chaves para ambos os usuários. Cada usuário utiliza a chave de forma distinta que é dividida em duas chaves de 16 bytes[27]. O usuário solicitante de conexão utiliza os primeiros 16 bytes para criptografar e os últimos 16 bytes para descriptografar. O usuário que aceitou o estabelecimento de conexão faz o inverso, ou seja, utiliza os primeiros 16 bytes para descriptografar e os últimos 16 bytes para criptografar. Criando uma chave diferente para ambos os sentidos de tráfego da informação.

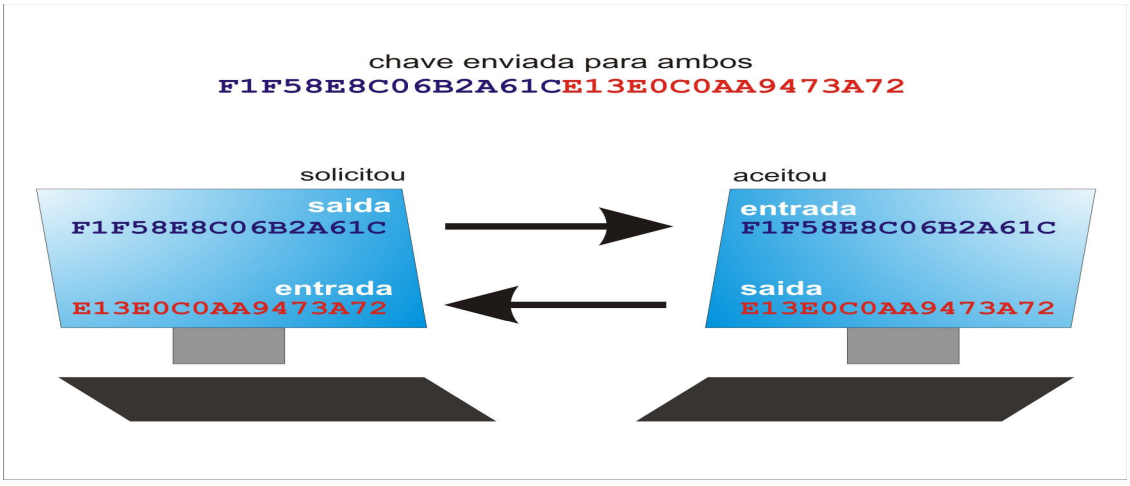

Figura 4.9 – Forma de Utilização da Chave no MiConversa
(Michelle Ribeiro Abuchahin em 07/11/2008).

26 Bytes é usado com freqüência para especificar o tamanho da memória ou da capacidade de armazenamento de um computador, independente do tipo de dados armazenados[27].

Para o processo de criptografia utilizou a seguinte estrutura:

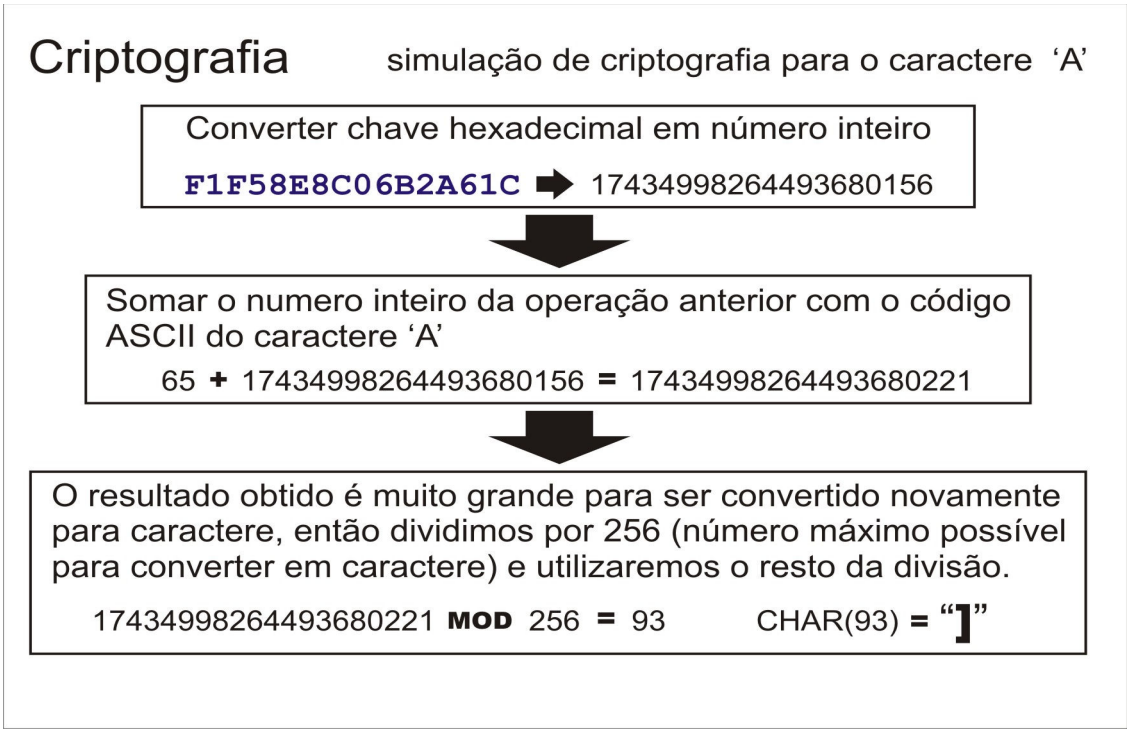

Figura 4.10 – Modelo Esquemático de Criptografia em MiConversa (Michelle Ribeiro Abuchahin em 07/11/2008).

Para conferirmos se o CHAR(93) = "]" e CHAR(65) = "A" basta que seja consultada a tabela ASCII, ou seja, código numérico usado para representar caracteres em arquivos texto em computadores e dispositivos de armazenamento eletrônico de dados[38].

Dec	Char	Dec	Char	Dec	Char	Dec	Char	Dec	Char	Dec	Char
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	|
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	_
48	0	64	@	80	P	96	`	112	p		

Figura 4.11 – Tabela ASCII (http://www.computerhope.com/jargon/a/ascii.htm, acesso em 07/11/2008).

E assim sendo para o processo de descriptografia utilizou-se a seguinte estrutura:

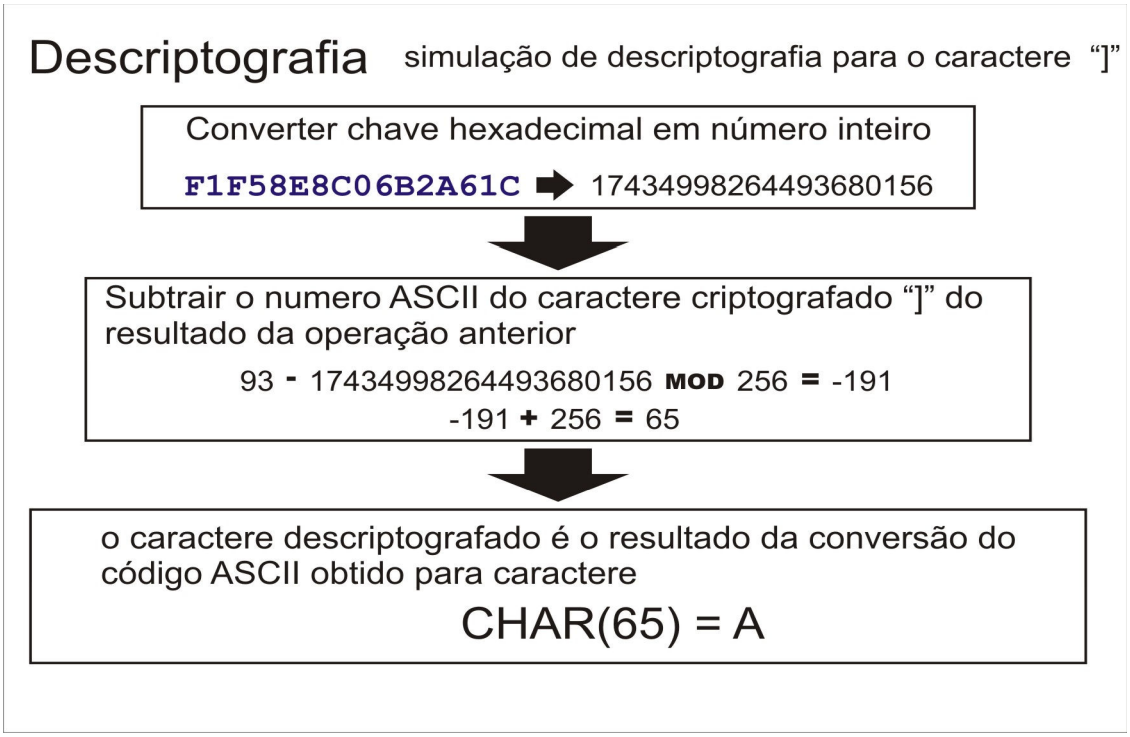

Figura 4.12 – Modelo Esquemático de Descriptografia em MiConversa (Michelle Ribeiro Abuchahin em 07/11/2008).

Capítulo 5 –Testes e Resultados

Este capítulo mostra os testes e resultados da execução do programa MiConversa e Servidor MiConversa.

Também são citados os problemas que foram ocorrendo durante o desenvolvimento do sistema.

5.1 Simulações e Testes

Após a confecção do software de mensageria instantânea cheguei na fase de simulações e testes que serão explicados a seguir.

5.1.1 Conexão

Foi criada uma rede local com três máquinas utilizando-se da topologia estrela. Uma máquina foi designada como o “Usuário 1”, a outra como “Usuário 2” e a outra como servidor.

Foi configurado o IP do servidor no “Usuário 1” e o IP do “Usuário 2” no servidor.

O “Usuário 1” solicita a conexão e ao ser estabelecida a conexão envia nome de usuário e senha se possuir cadastro no Servidor MiConversa o mesmo envia uma chave gerada única e exclusivamente utilizada nesta conexão e solicita conexão com “Usuário 2”. Ao ser estabelecida a conexão com o “Usuário 2” o mesmo envia seu usuário e senha para o Servidor MiConversa, caso esteja cadastrado o Servidor MiConversa envia ao “Usuário 2” a mesma chave enviada ao “Usuário 1”.

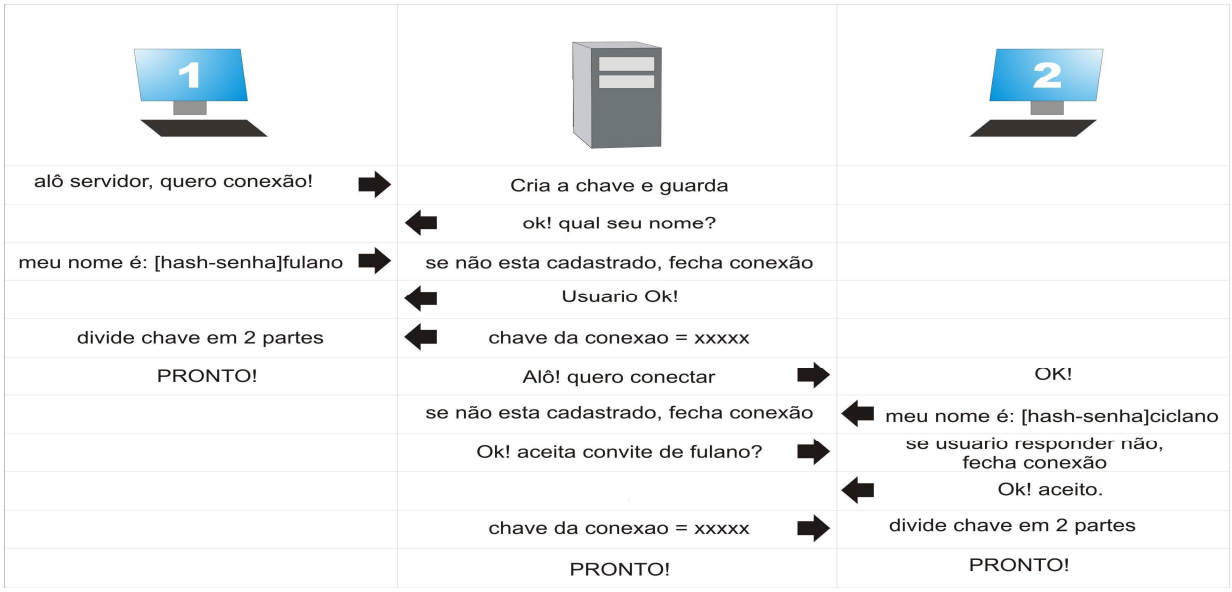

Figura 5.1 – Negociação da Conexão
(Michelle Ribeiro Abuchahin em 07/11/2008).

A partir deste momento toda mensagem enviada pelo “Usuário 1” ao Servidor MiConversa é repassada ao “Usuário 2” e vice-versa.

O Servidor MiConversa não possui a função de descriptografia apesar de ter a chave utilizada na conversação. O mesmo só envia a chave para o usuário após a confirmação de existência do nome de usuário e senha recebido pelo usuário no banco de dados.

A figura 5.2 mostra o Servidor MiConversa durante o processo de conexão e troca de mensagens entre os usuários previamente cadastrados João e Michelle.

A tela abaixo comprova que apenas após a validação do usuário é que o servidor envia a chave utilizada na conversa valendo salientar que a mensagem está trafegando “ilegível”, ou seja, criptografada.

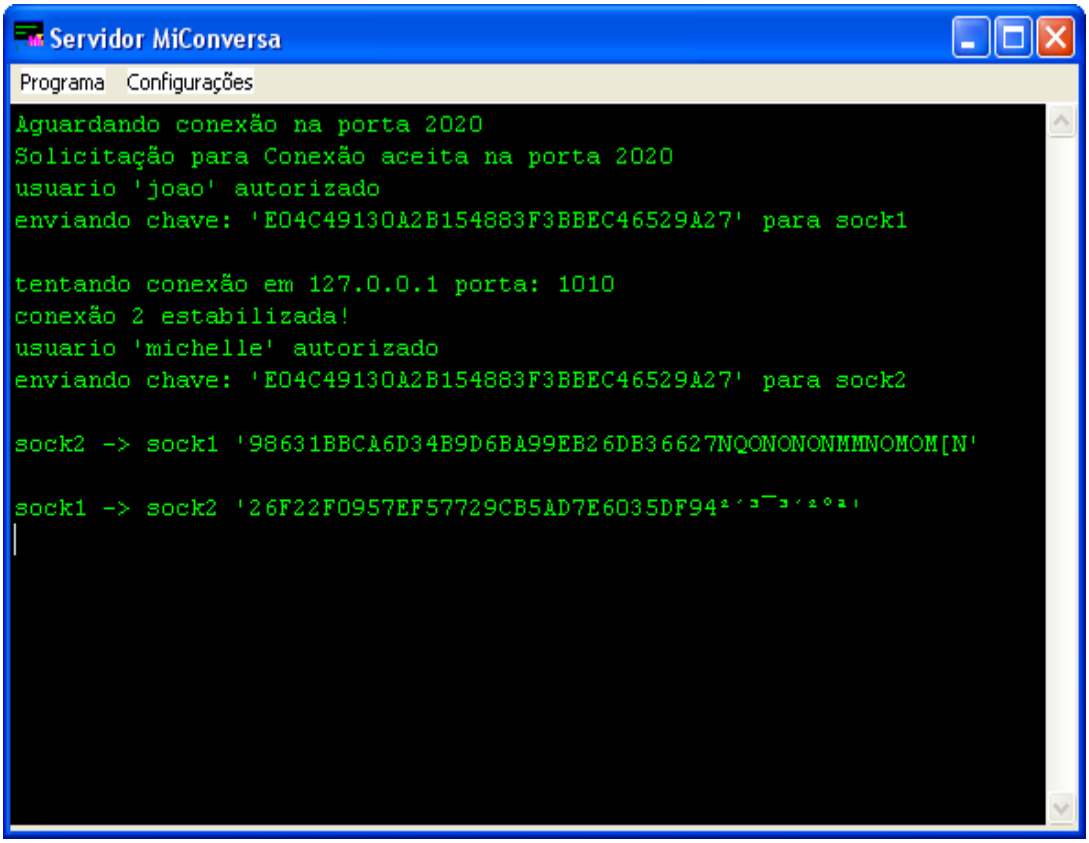

Figura 5.2 – Servidor MiConversa em Atividade

5.1.2 Processo de Troca de Mensagens

Estabelecida a conexão temos a seguinte estruturação:

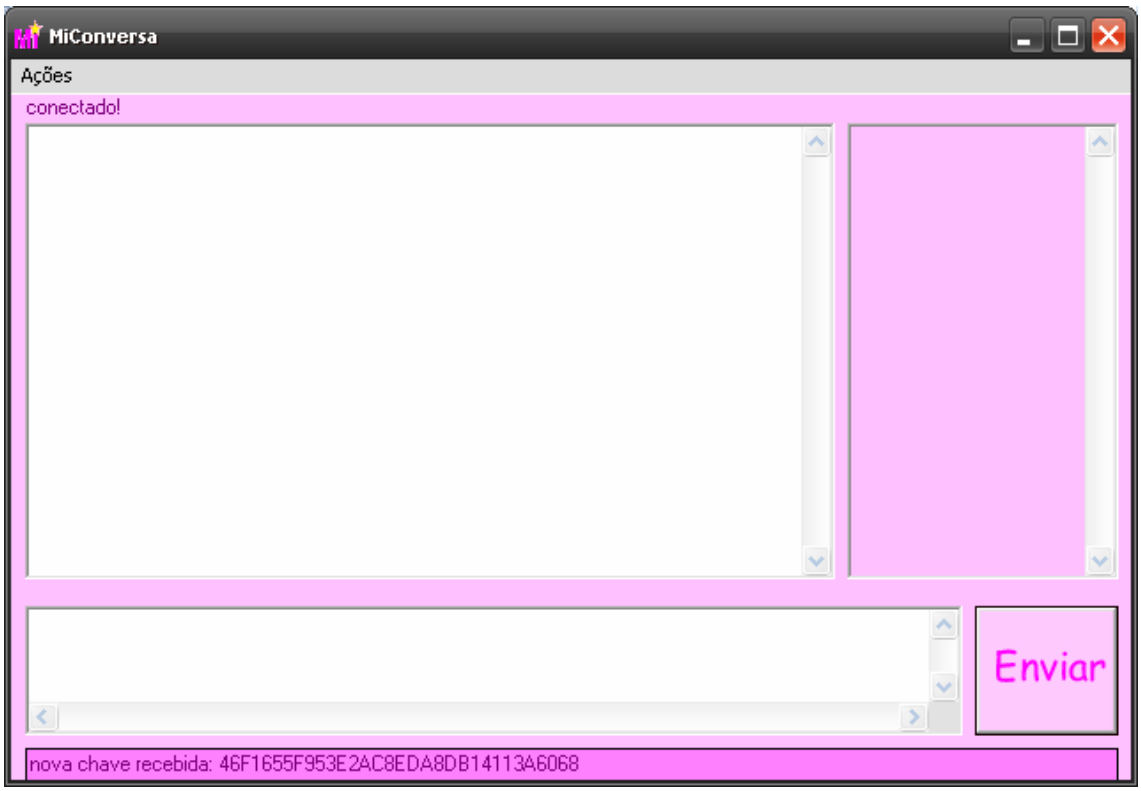

Figura 5.3 – Usuário Requisitante de Conexão

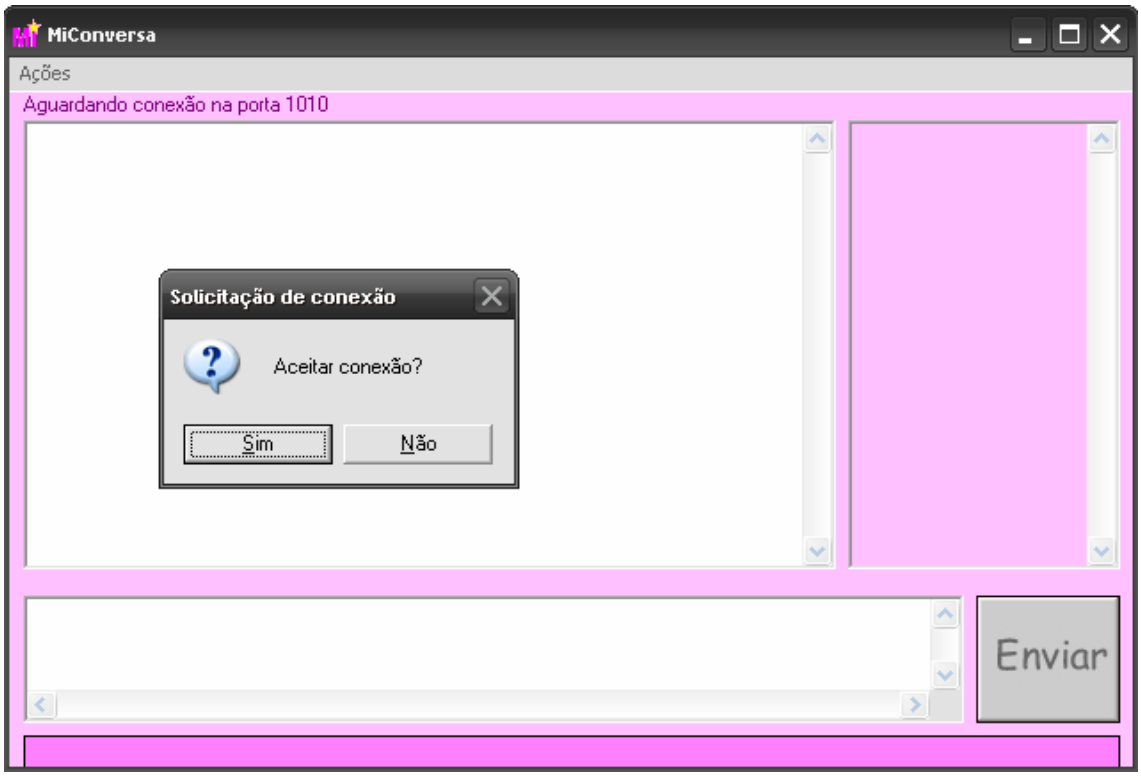

Figura 5.4 – Usuário Aceita ou Não Conexão

Se o usuário não aceitar a conexão a mesma será encerrada para ambas as partes.

Caso o usuário aceite estabelecer a conexão temos as seguintes tela:

O “Usuário 1” troca mensagens com o “Usuário 2” e vice-versa, no campo de texto de entrada o usuário digita a mensagem e clica no botão Enviar ou pressiona a tecla Enter durante este processo observasse que a mensagem digitada no campo já aparece criptografada no campo de texto em rosa. Ao efetuar o comando de envio, é criado um hash da mensagem utilizando a mensagem original e criptografada, o mesmo é adicionado no início da mensagem criptografada e enviado para o servidor. O servidor repassa a mensagem ao usuário destinatário que receberá a mensagem separando-se os primeiros 32 caracteres referentes ao hash, descriptografando a mensagem, usando a chave conhecida, criando-se outro hash com o resultado da descriptografia e a mensagem criptografada recebida, no qual comparasse o novo hash gerado com o hash recebido, se for igual mostra no campo de texto de saída o resultado da mensagem descriptografada. Se for diferente significa que a mensagem não está integra, que falta parte da mensagem ou a mesma sofreu alguma manipulação, e será mostrada a mensagem: “A mensagem não passou no teste de integridade”.

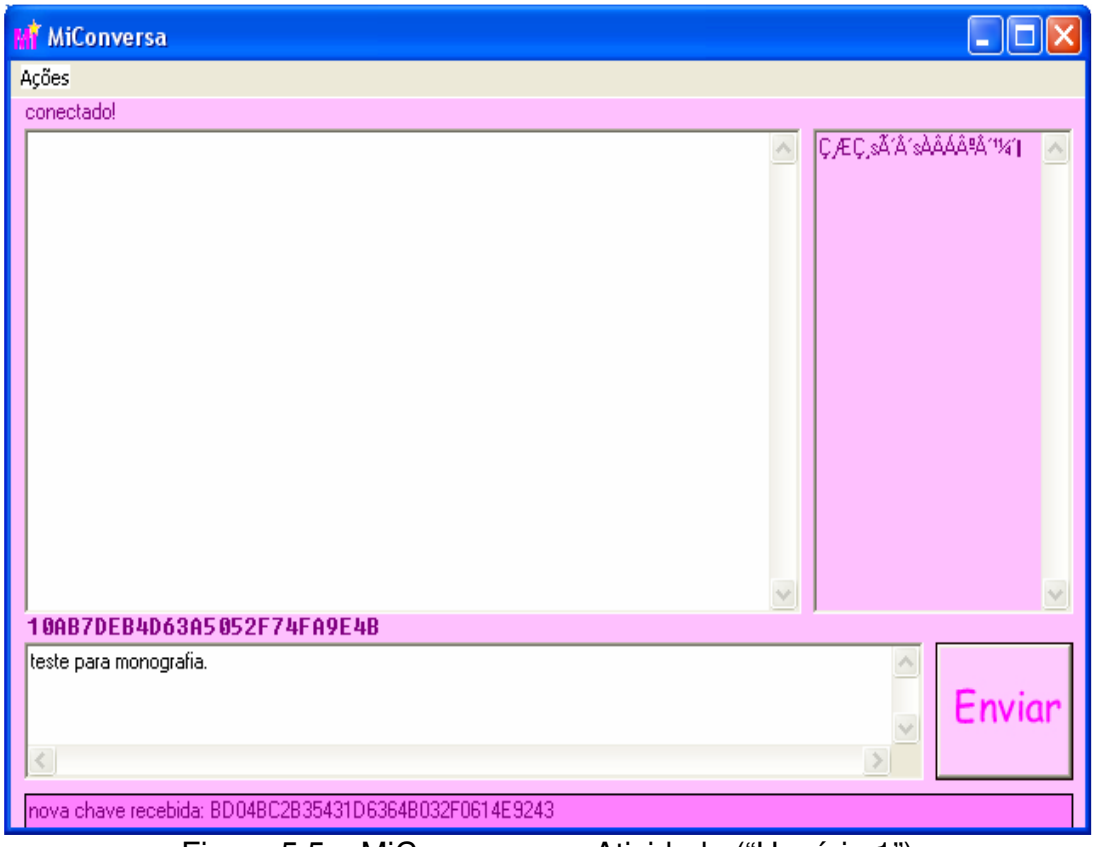

Figura 5.5 – MiConversa em Atividade (“Usuário 1”)

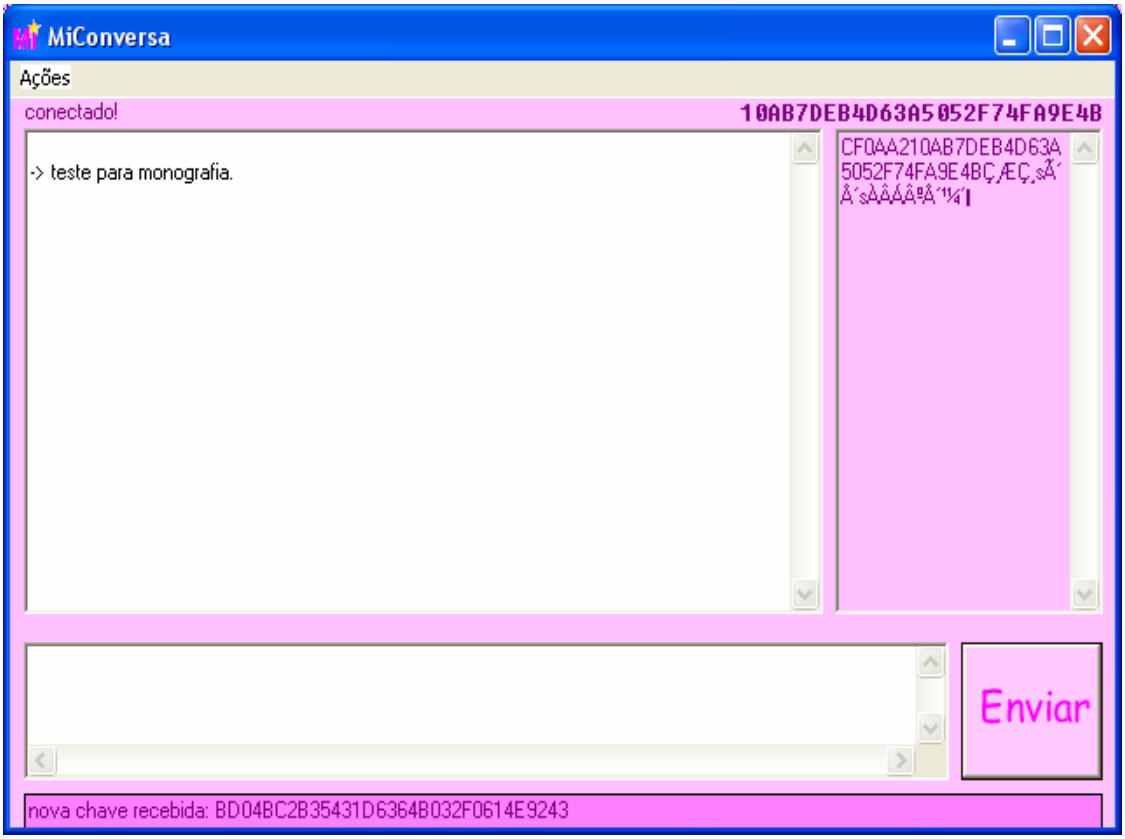

Figura 5.6 – MiConversa em Atividade (“Usuário 2”)

5.2 Críticas ao Sistema

Nesta fase menciona-se um erro premeditado que inviabiliza o cadastramento de um usuário.

Caso o administrador do servidor tente cadastrar um usuário previamente cadastrado é apresentada a seguinte mensagem:

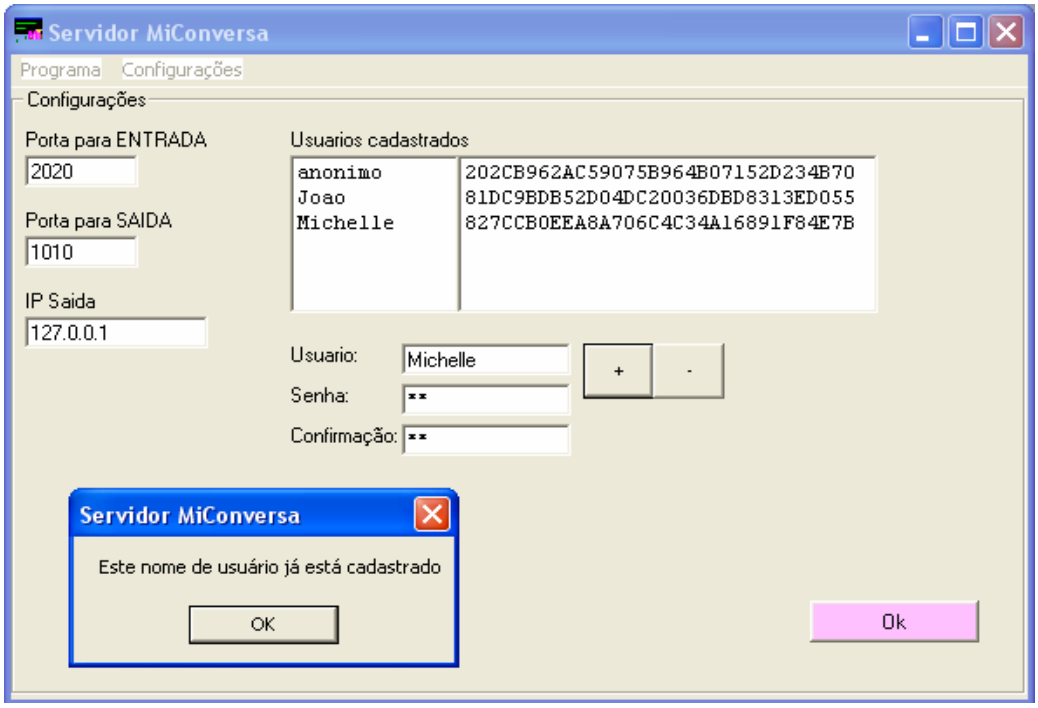

Figura 5.7 – Crítica em MiConversa

5.3 Assinatura Digital

Antes de enviar uma mensagem o usuário observa que a mensagem está sendo criptografada e gera uma assinatura digital que será o hash da soma da mensagem descriptografada concatenada com a mensagem criptografada.

Essa assinatura digital é incluída no início da mensagem criptografada, ao chegar no seu destino, a mensagem é dividida em duas partes:

Os primeiros 32 caracteres é a assinatura digital então a mensagem é descriptografada e cria-se um hash dessa mensagem concatenada com a criptografada e compara-se com a assinatura digital recebida. Se for diferente, o programa avisa que a mensagem não passou no teste de integridade.

5.4 Testes

Nesta fase mencionam-se os testes realizados no MiConversa.

5.4.1 Tentativa de Acesso Para Usuário Não Cadastrado

Foi realizado um teste para um usuário não cadastrado, ou seja, um usuário que não tem permissão para participar do processo de conversação.

O usuário conforme figura abaixo, tenta se conectar no MiConversa para possível estabelecimento de troca de mensagens com outro usuário.

Caso este usuário intruso não esteja previamente cadastrado no Servidor MiConversa não é permitido sua entrada para o estabelecimento de conexão, mesmo que o intruso descubra um nome previamente cadastrado terá ainda que descobrir a senha do usuário.

Figura 5.8 – Tentativa de Acesso Para Usuário Não Cadastrado

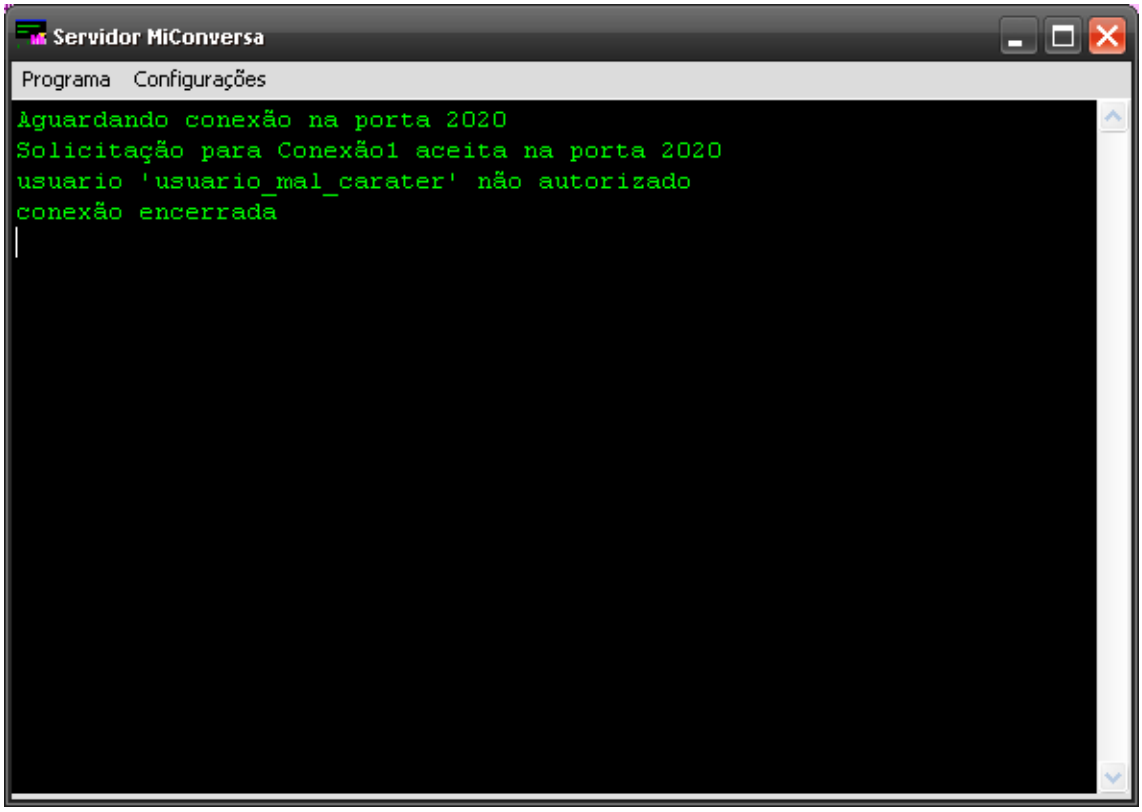

Figura 5.9 – Servidor MiConversa Para Tentativa de Acesso Para Usuário Não Cadastrado

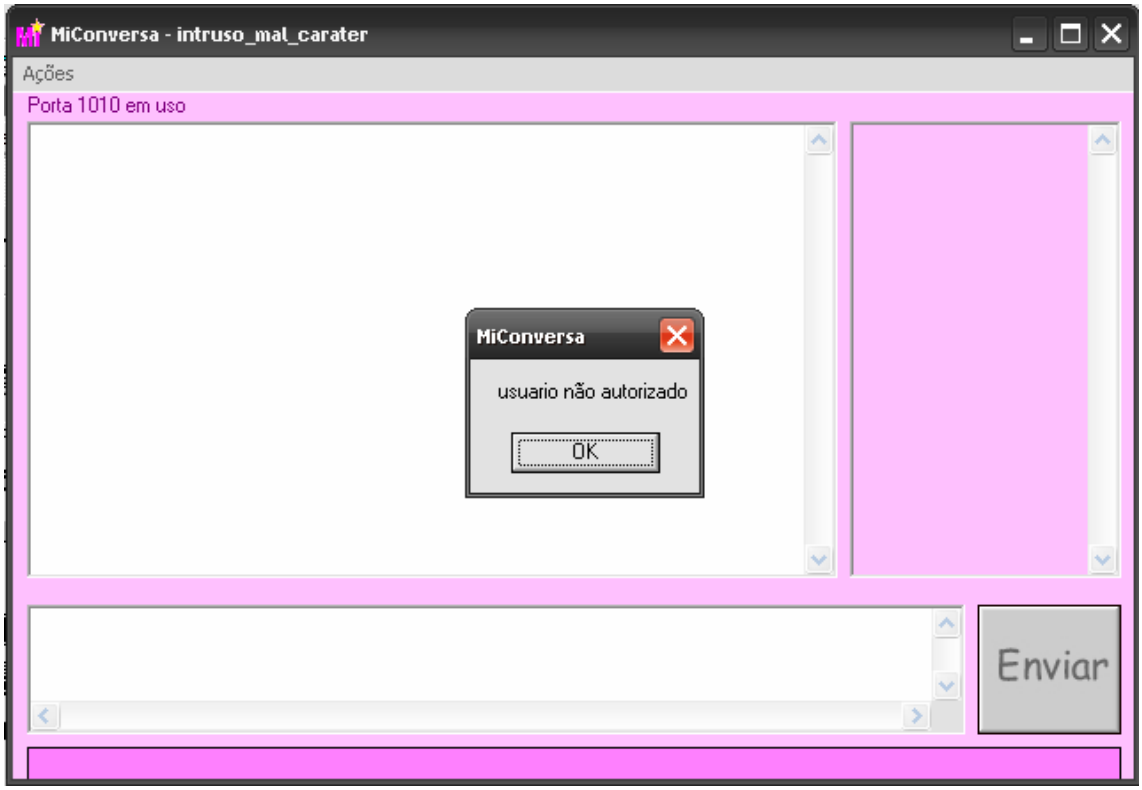

Figura 5.10 – MiConversa Para Tentativa de Acesso Para Usuário Não Cadastrado

5.4.2 Tentativa de Reaproveitamento de Chave

Para este teste comprovasse que não existe a possibilidade de reaproveitamento de uma chave já utilizada em processo de troca de mensagens.

Para isto foi criada uma função no sistema que permite alterar a chave durante a conversão, então salvasse a chave já utilizada em uma conexão, encera e reativa a mesma.

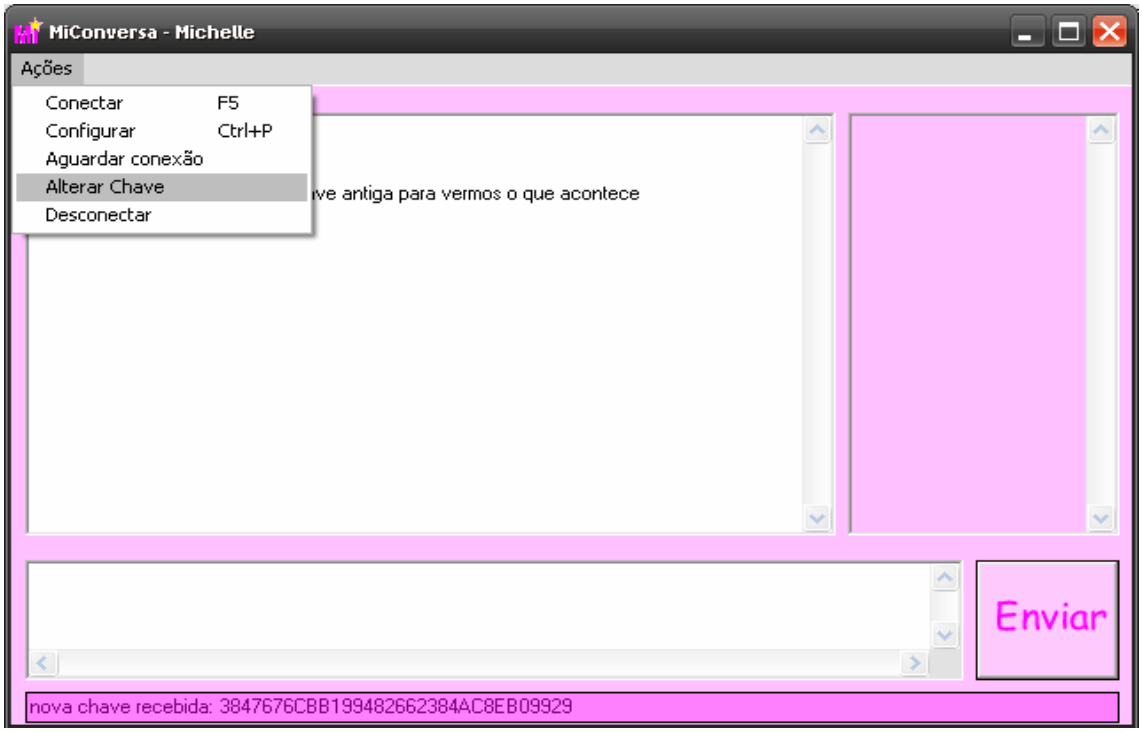

Figura 5.11 – Tentativa de Reaproveitamento de Chave

Foi digitada a chave no campo: “Digite a nova chave ou cancele”.

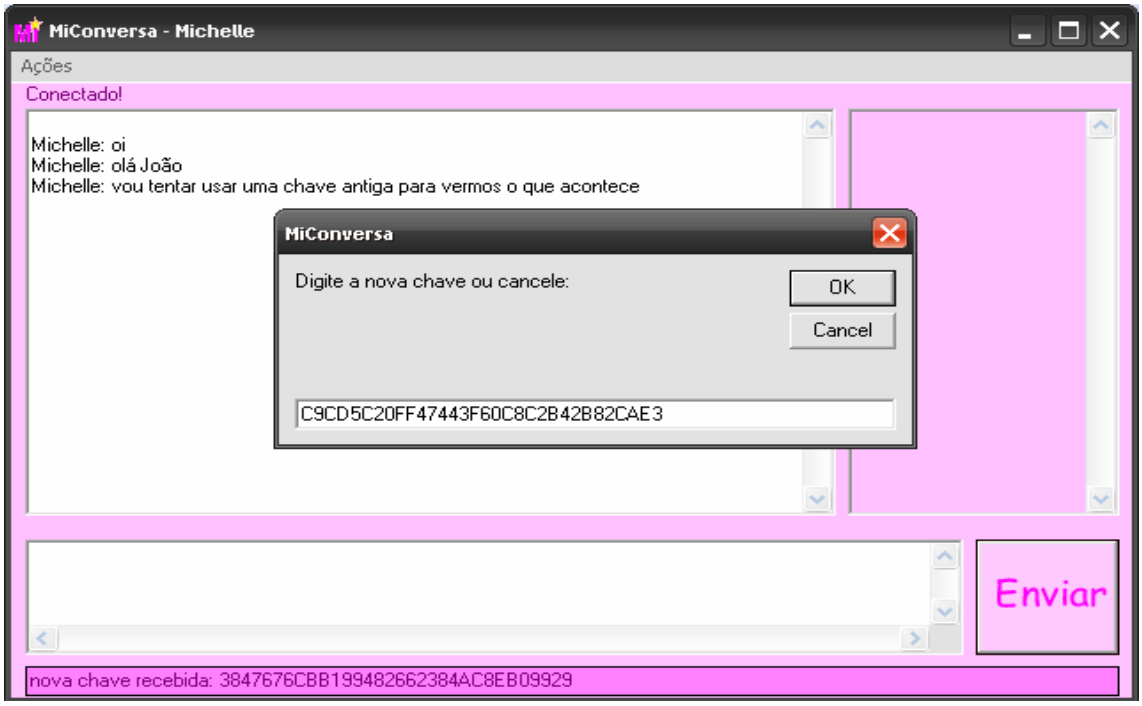

Figura 5.12 – Digitando a Chave

E podemos observar que após ser digitada a chave, imaginasse que estamos participando da conversação, mas receberemos a seguinte mensagem:

“A mensagem não passou no teste de integridade”, a mensagem não está integridade na conversação, ou seja, burlou o pré requisito de segurança.

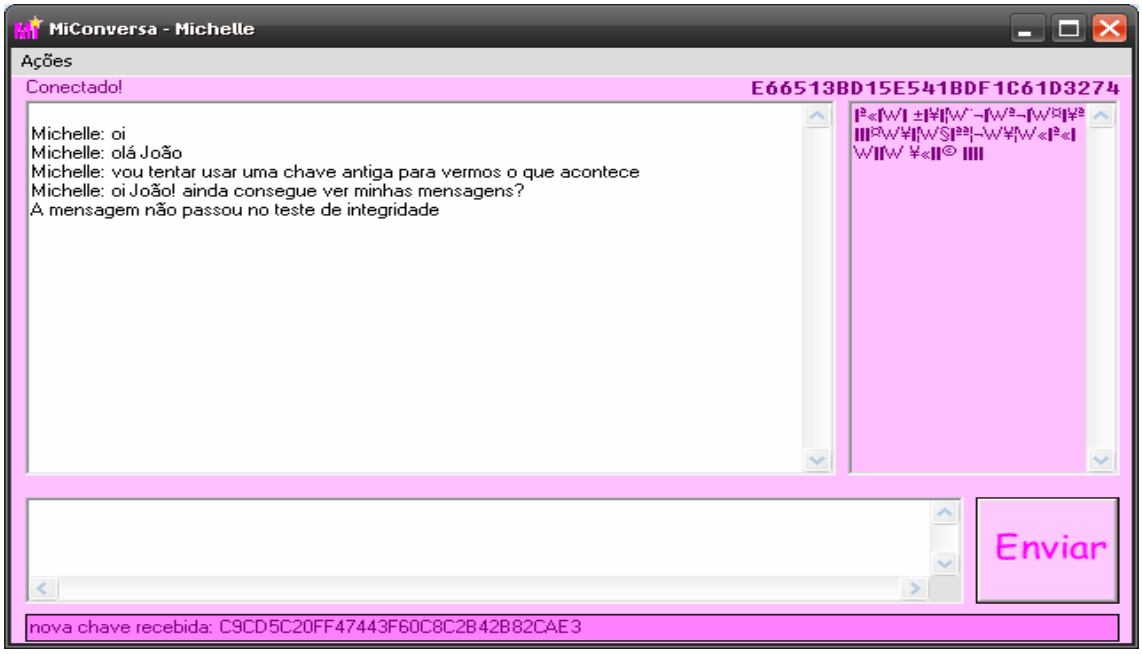

Figura 5.13 – Resposta a Tentativa de Reaproveitamento de Chave

Isto ocorre porque as mensagens recebidas tornam-se ilegíveis e não passa no teste de integridade, pois ele testa a mensagem recebida juntamente com sua assinatura digital.

É importante ressaltar que o Servidor MiConversa possui a função de repassar as mensagens recebidas e não descriptografa as mensagens para garantir a privacidade dos usuários.

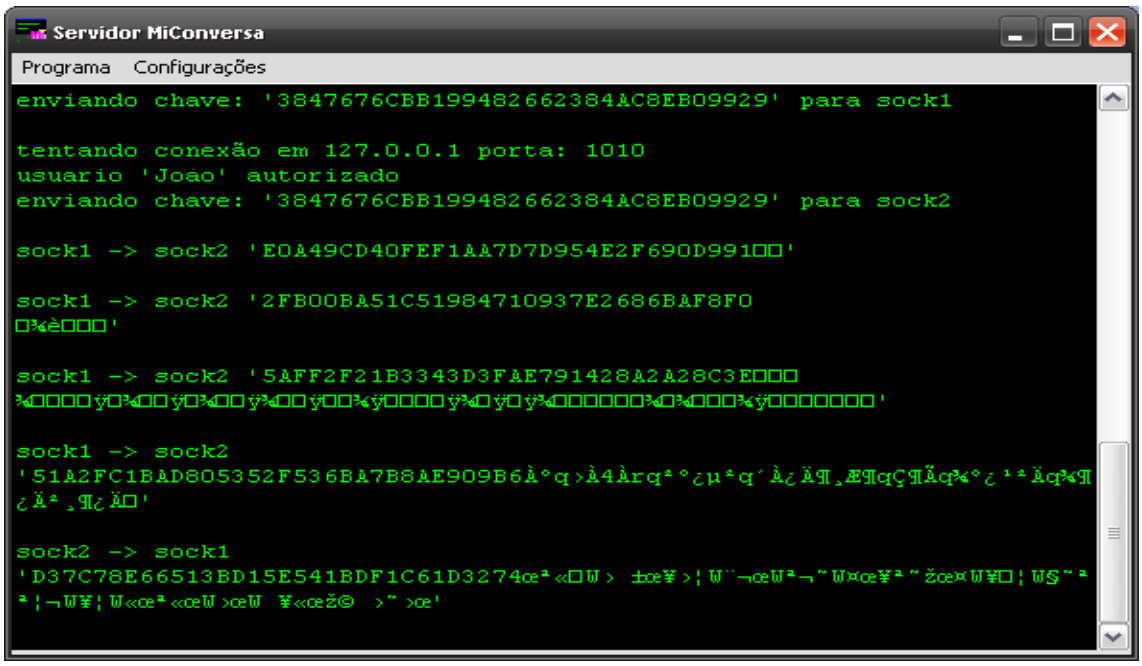

Figura 5.14 – Privacidade no Servidor MiConversa

5.4.3 Outros Testes Realizados

O teste mais importante realizado foi comprovar que a emissão e recepção das mensagens estão trafegando na rede de maneira criptografada e com isto garantindo confidencialidade dos dados para os usuários.

Para esta comprovação foi instalado um software para captura de pacotes que estão trafegando numa rede, chamado WIRESHARK. Para sua utilização o mesmo pode ser obtido no link http://baixaki.ig.com.br/download/wireshark.htm acesso em 20/09/2008.

A mensagem escolhida para ser enviada foi um trecho de um livro de DEEPAK CHOPRA:

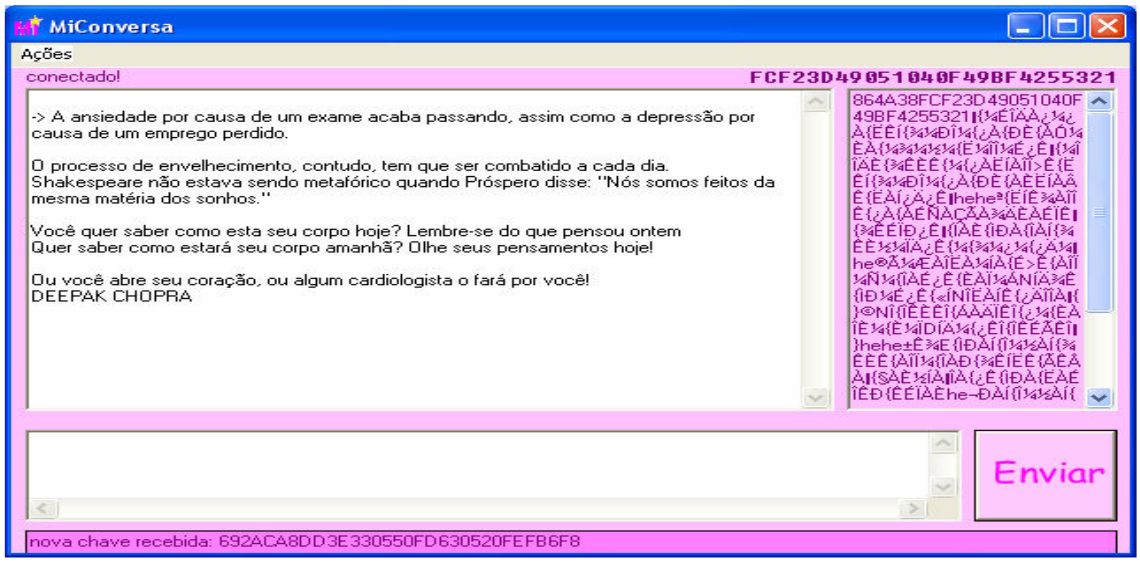

Figura 5.15 – Mensagem Enviada Criptografada

O software WIRESHARK estava capturando os pacotes no exato momento em que a mensagem chegou ao destinatário e as informações podem ser provadas conforme figura abaixo:

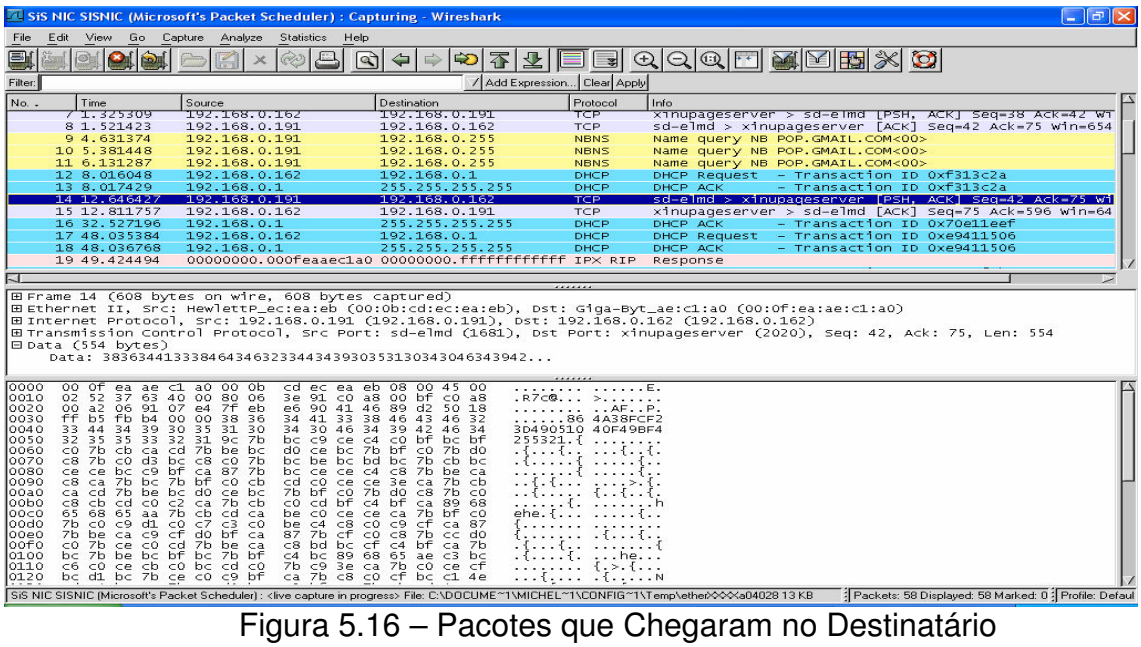

Figura 5.16 – Pacotes que Chegaram no Destinatário

E ainda comprovasse que a mensagem chegou no usuário de destino bem como passou pelo servidor criptografada:

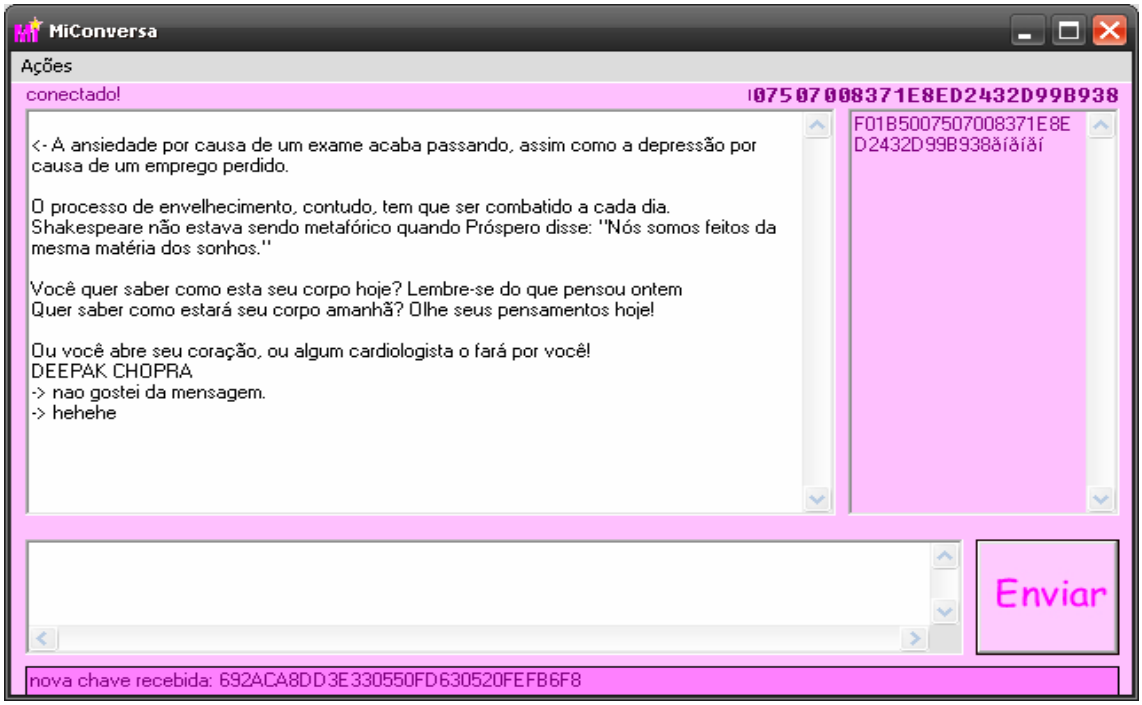

Figura 5.17 – Mensagem e Resposta

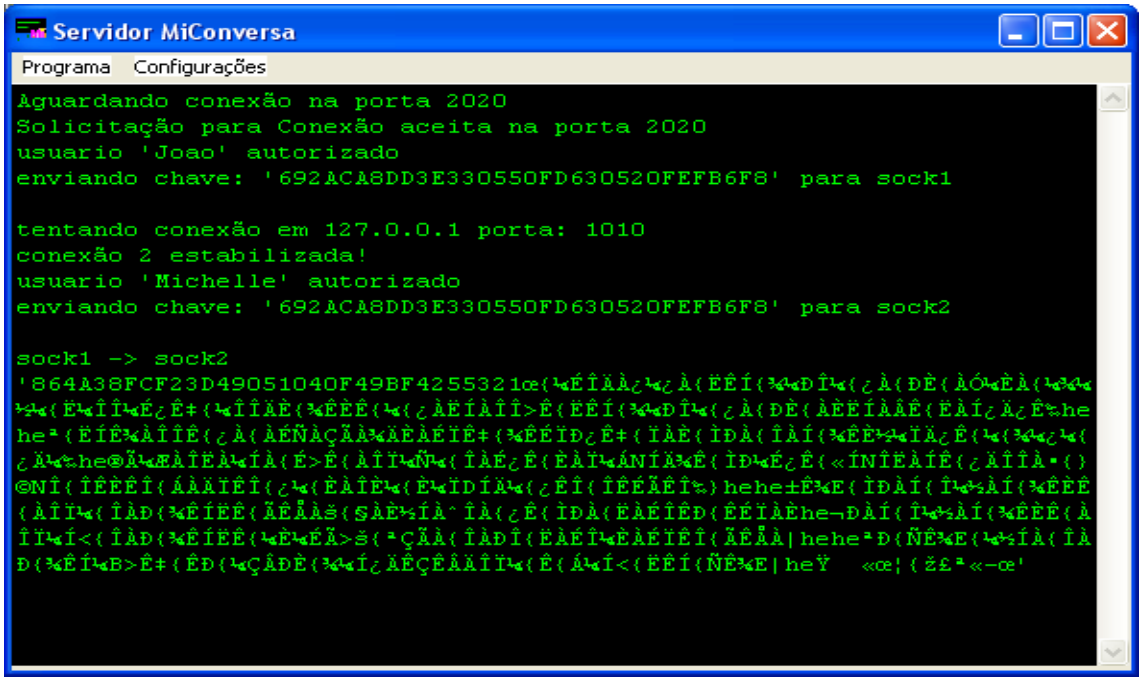

Figura 5.18 – Mensagem e Resposta no Servidor MiConversa

Após a localização da mensagem e verificação que ela está totalmente codificada, é possível afirmar que as informações estão sendo trafegadas de forma segura. E assim, se alguém mal intencionado interceptasse a mensagem enviada, não iria identificar o conteúdo das informações.

Este teste confirma que o desenvolvimento do MiConversa aconteceu de acordo com a implementação descrita e proposta para projeto.

5.5 Dificuldades

Após a definição do protótipo, apresentou certa dificuldade na escolha da linguagem de programação a ser utilizada bem como o aprendizado da mesma.

A implementação do algoritmo de criptografia mostrou-se dificultosa em ser realizada e requereu um estudo aprofundando.

A necessidade de existência de autenticação na conexão dificultou o desenvolvimento do software.

A escolha de um método de geração de chave aleatória que fosse seguro e simples de implementar, mas que mostrasse bastante dificultoso, uma vez que a finalidade do projeto é acadêmico.

A conexão com socket é apontada como um dos principais problemas, pois apresentou diversos erros que tiveram que ser resolvidos.

A escolha do método de criptografia foi importante e ao mesmo tempo tentei de todas as formas simplificar para dar ao software um dinamismo e facilidade de compreensão para os usuários.

5.6 Demonstração da Implementação

São demonstradas as funcionalidades do MiConversa que vieram para suprir os problemas levantados na proposta do projeto. Os testes realizados serão detalhados da seguinte forma:

- Serão utilizados três computadores conectados diretamente em uma rede de topologia estrela;
- Com a inicialização do Servidor MiConversa em uma máquina e do MiConversa em duas máquinas, serão cadastrados dois usuários;
- O “Usuário 1” adicionará o endereço de IP do “Usuário 2” e vice-versa;
- Após este processo caso os usuários aceitem estabelecer a conexão será iniciado o processo de troca de mensagens, demonstrando as funcionalidades do MiConversa;
- Para finalizar como procedimento de comprovação de mensagens criptografadas em rede, utilizarei o WIRESHARK para demonstração de que os pacotes trafegados na rede estão criptografados, provando desta forma a segurança do sistema.

5.7 Considerações finais

Foram realizados testes e simulações das funcionalidades do comunicador de mensageria instantâneo descrito no capítulo 4, obtendo sucesso em seus resultados finais.

No decorrer dos testes e implementação decidi usar uma chave de 32 caracteres aonde o usuário que requisitou a conexão os primeiros 16 caracteres foram utilizados para criptografar a mensagem e os 16 últimos para descriptografar. Já no usuário que aceita a conexão os últimos 16 caracteres foram utilizados para descriptografar a mensagem e os 16 primeiros para criptografar.

Isto fez com que mesmo que os dois usuários digitem a mesma mensagem o resultado da criptografia e conseqüentemente o hash são diferentes.

Esta utilização é considerada importante, pois se por exemplo, você digitar a senha do seu cartão para o receptor e mesmo que o intruso saiba o resultado criptográfico de cada caractere, ele não consegue desvendar, pois o resultado da criptografia não será o mesmo para emissor e receptor.

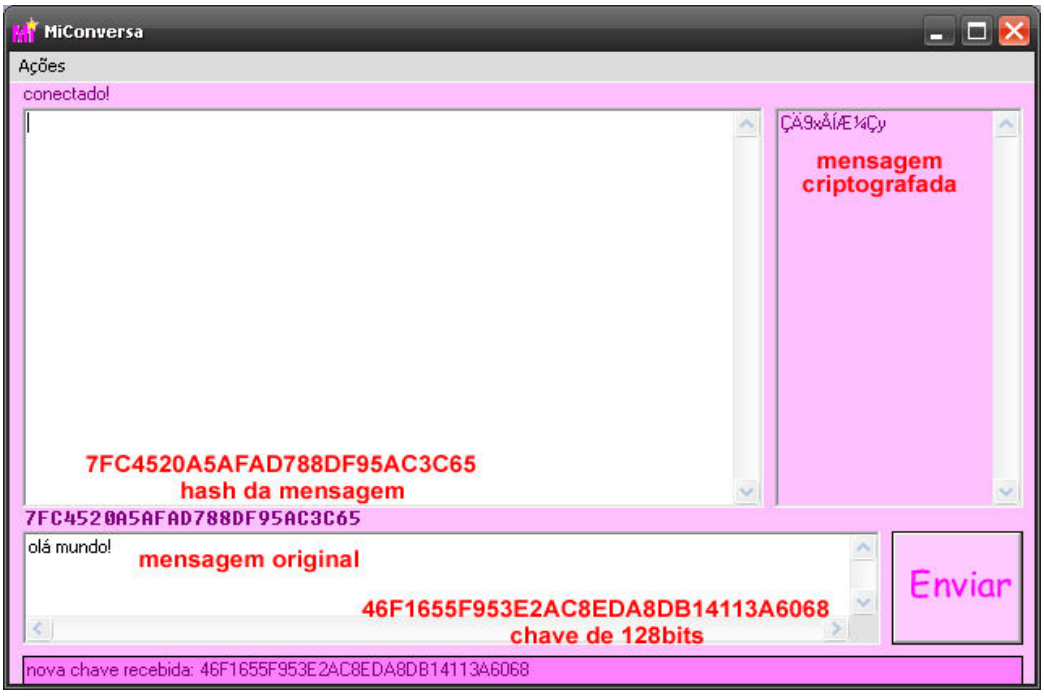

Figura 5.19 – Mensagem olá mundo! (Usuário1)

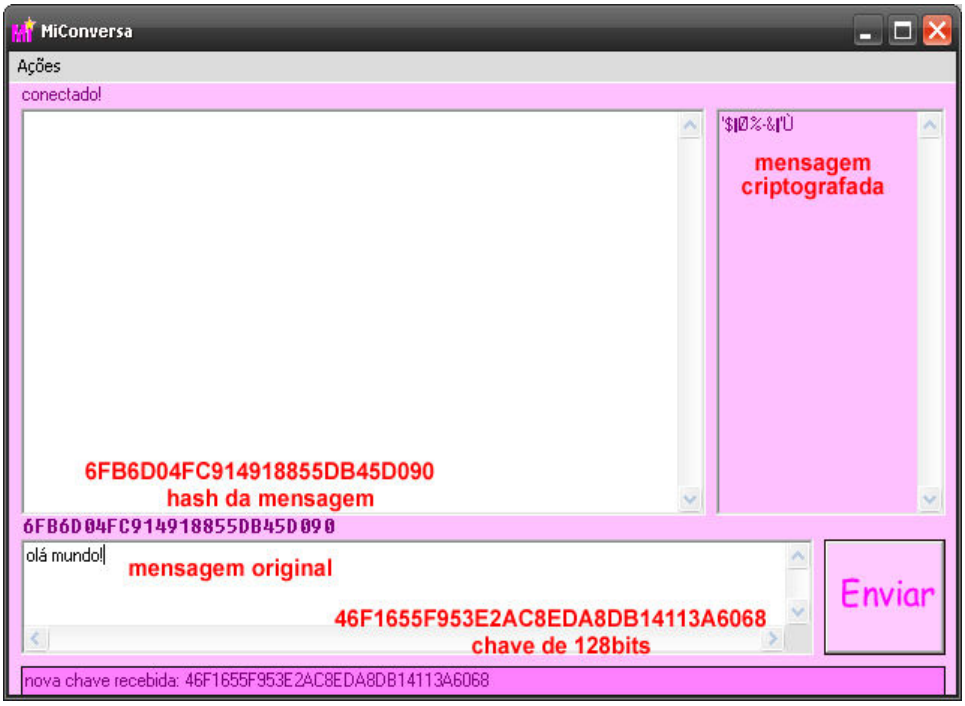

Figura 5.20 – Mensagem olá mundo! (“Usuário 2”)

Capítulo 6 – Conclusão

O estudo de comunicadores instantâneos apresenta o fato de que a segurança não é um ponto primordial levado em consideração para o desenvolvimento destas ferramentas. O ponto de partida do projeto foi a busca por desenvolvimento de um software com finalidade acadêmica que garante a segurança para os usuários no processo de troca de mensagens.

A principal preocupação é desenvolver um software em uma linguagem de programação que fosse de fácil entendimento até mesmo para os programadores iniciantes e que tivesse um dinamismo uma vez que o foco não era em um software comercial.

O objetivo principal é alcançado com o desenvolvimento do MiConversa, ou seja, um software que garante a integridade dos dados e principalmente garante que as mensagens trafegadas estão trafegando criptografadas.

Portanto, o MiConversa é de grande importância para os usuários que querem garantir a privacidade de suas informações ao utilizaram-se um comunicador instantâneo.

6.1 Projetos Futuros

As sugestões para evolução do projeto são:

- Aprimoramento do comunicador instantâneo, implementando transferência de arquivos, transferência de imagens, transferência de áudio bem como outras funcionalidades existentes nos comunicadores instantâneos utilizados atualmente;

- Transformação do MiConversa para que o mesmo perdesse a finalidade acadêmica e virasse um software de comunicação instantânea comercial.

REFERÊNCIAS BIBLIOGRÁFICAS

[1]http://www.viuisso.com.br/2005/03/21/brasil-e-lider-no-uso-de-messenger, acesso em 01/09/2008.

[2]http://www.e-commerce.org.br/STATS.htm#C, Dados Estatísticos Sobre a Internet e Comercio Eletrônico, acesso em 12/09/2008.

[3]BARBER, Joe, 2004, Development of an instant menssaging system to enable secure business communication.

[4]http://twiki.im.ufba.br/pub/GESI/WebHome/SeguranaemSistemasDistribudos.pdf, acesso em 12/09/2008.

[5]http://www2.agence.com.br/novidades-sobre-mercado-web.php?news=219, acesso em 12/09/2008.

[6]COMER, Douglas E., STEVENS, David L., 1999, Interligação em Rede com TCP/IP – Ed. Elsevier.

[7]http://www.millenniumbcp.pt/multimedia/archive/00368/newsletter_368118a.pdf, acesso em 12/09/2008.

[8]TORRES, Gabriel., 2001, Redes de Computadores Curso Completo – ed. Axcel Boooks.

[9]http://www.plugmasters.com.br/sys/materias/875/1/Sockets, acesso em 12/09/2008.

[10]http://www1.fatecsp.br/aguiar/sistemasdistribuidos.htm, 13/09/2008.

[11]http://www.conectiva.com/doc/livros/online/10.0/usuario/pt_BR/ch03s03.html, acesso em 16/09/2008.

[12]http://www.cadcobol.com/ww.htm, acesso em 16/09/2008.

[13]TERADA, Routo., 2000, Segurança de Dados Criptografia em Redes de Computador – ed. Edgard Blucher Ltda.

[14]http://www.infowester.com/firewall.php, acesso em 20/09/2008.

[15]http://gsisic.serpro.gov.br/noticias/Trojan/20070731_01, acesso em 20/09/2008.

[16]http://webmessenger.msn.com/?mkt=pt-br, acesso em 20/09/2008.

[17]http://www.windowslive.com.br/index_msn.html, acesso em 20/09/2008.

[18]http://www.nlnp.net/lazer.htm, acesso em 20/09/2008.

[19]http://www.icq.com.br/, acesso em 20/09/2008.

[20]http://br.messenger.yahoo.com/, acesso em 20/09/2008.

[21]http://www1.folha.uol.com.br/folha/informatica/ult124u333297.shtml, acesso em 20/09/2008.

[22]http://www.symantec.com/pt/br/security_response/writeup.jsp?docid=2002-082718-3007-99, acesso em 20/09/2008.

[23]http://www.wordreference.com/enpt/Payload, acesso em 28/09/2008.

[24]http://www.superdicas.com.br/infovir/anti-virus.asp, acesso em 28/09/2008.

[25]http://www.ziggi.com.br/artigos/detalhes/defendase-das-ameacas-em-mensagens-instantaneas, acesso em 28/09/2008.

[26]http://www.infowester.com/criptografia.php, acesso em 23/09/2008.

[27]TANENBAUM, Andrew S., 2003, Sistemas Operacionais Modernos – ed. Campus.

[28]http://securityresponse.symantec.com/region/br/enterprisesecurity/content/expert/BR_1341.html, acesso em 27/09/2008.

[29]http://www.terra.com.br/informatica/2002/02/14/006.htm, acesso em 17/10/2008.

[30]http://www.trendmicro.com/br/about/news/pr/archive/2005/pr060205.htm, acesso em 17/10/2008.

[31]http://www.criarweb.com/javascript/, acesso em 17/10/2008.

[32]http://www.forumpcs.com.br/noticia.php?b=146515, acesso em 17/10/2008.

[33]http://idgnow.uol.com.br/seguranca/2007/11/20/idgnoticia.2007-11-20.4716028516/, acesso em 17/10/2008.

[34]http://www.symantec.com/region/br/home_homeoffice/library/im_risks.html, acesso em 17/10/2008.

[35]http://www.forumpcs.com.br/noticia.php?b=146515, acesso em 17/10/2008.

[36]http://www.linhadefensiva.org/2007/07/spim-bloqueio-msn/, acesso em 31/10/2008.

[37]www.bullzip.com, acesso em 31/10/2008.

[38]http://tumitus.com/glossario.php, acesso em 07/11/2008.

APÊNDICES

Como o MiConversa é um software com finalidade acadêmica exponho todos os códigos fontes utilizados na criação do mesmo, assim como do Servidor MiConversa.

APÊNDICE A

Software MiConversa.

```
Public privateKey As String 'variável pública dentro do form. será visivel por todas as funções
Public privateKeyd As String
Public encryptKey As Integer
Public decriptKey As Integer
Public meunome As String
Public meuamigo As String
Dim md5Test As MD5

Private Sub btnenviar_Click()

   If (Winsock1.State = sckConnected) Then
      Winsock1.SendData (UCase(lblhash.Caption) & txtcrypt.Text) ' EncryptData(txtentrada.Text))
      txtsaida.Text = txtsaida.Text & vbCrLf & meunome & ": " & txtentrada.Text
      txtentrada.Text = ""
   ElseIf (sockListen.State = sckConnected) Then
      sockListen.SendData (UCase(lblhash.Caption) & txtcrypt.Text) '& EncryptData(txtentrada.Text))
      txtsaida.Text = txtsaida.Text & vbCrLf & meunome & ": " & txtentrada.Text
      txtentrada.Text = ""
   End If


End Sub


Private Sub cmdconfigok_Click()

   mnuconfigurar_Click

End Sub

Private Sub Form_Activate()

   txtentrada.SetFocus
   meunome = txtnomeusuario.Text
   Me.Caption = "MiConversa - " & meunome

End Sub

Private Sub mnuaguardarconexao_Click()

   aguardarConexao

End Sub

Private Sub mnualterarchave_Click()
Dim strInput As String

strInput = InputBox("Digite a nova chave ou cancele:")

If (strInput <> "") Then
   If (Len(strInput) = 32) Then
      privateKey = Mid(strInput, 1, 16)
      privateKeyd = Mid(strInput, 17, 16)
      encryptKey = keyToNum(privateKey)
      decriptKey = keyToNum(privateKeyd)
      txtTemp.Text = "nova chave recebida: " & strInput
   Else
```

```
        MsgBox "chave inválida"
    End If
End If

End Sub

Private Sub mnuconectar_Click()

    If (txtnomeusuario.Text = "") Then
        MsgBox ("Você precisa especificar um nome para conexão")
        mnuconfigurar_Click
        Exit Sub
    End If

    If Winsock1.State <> sckClosed Then
        Winsock1.Close
    End If

    Winsock1.Connect txtip.Text, txtport.Text

End Sub

Private Sub Form_Load()

    ' CONFIGURAÇÕES GRÁFICAS INICIAIS
        frmconfig.Left = 0
        Me.Height = 6600
        lblstatus.Caption = ""

    'função interna que faz com que o sock aguarde uma conexão
        aguardarConexao

    'cria objeto md5Test como MD5
        Set md5Test = New MD5

End Sub

Private Sub mnuconfigurar_Click()

    frmconfig.Visible = Not frmconfig.Visible

    If (Not frmconfig.Visible) Then
        aguardarConexao
        meunome = txtnomeusuario.Text
        Me.Caption = "MiConversa - " & meunome
    End If

End Sub


Private Sub aguardarConexao()

    On Error GoTo erro

    privateKey = ""
    privateKeyd = ""
    txtTemp.Text = ""
    Winsock1.Close
    sockListen.Close
    sockListen.LocalPort = txtportlisten.Text
    sockListen.Listen
    privateKey = 0
    lblstatus.Caption = "Aguardando conexão na porta " & sockListen.LocalPort
```

```
    GoTo fim


erro:
        lblstatus.Caption = "Porta " & sockListen.LocalPort & " em uso"
fim:

End Sub



Private Sub mnusair_Click()
sockListen.Close
Winsock1.Close
aguardarConexao
privateKey = ""
privateKeyd = ""
txtentrada.Text = ""
txtcrypt.Text = ""
txtsaida.Text = ""
txtTemp.Text = ""
End Sub

Private Sub sockListen_Close()

    lblstatus.Caption = "conexão encerrada!"
    btnenviar.Enabled = False
    privateKey = 0
    aguardarConexao

End Sub


Private Sub sockListen_ConnectionRequest(ByVal requestID As Long)

    Dim hash As String

    'iRet = MsgBox("Aceitar conexão?", vbQuestion + vbYesNo, "Solicitação de conexão")

    'If (iRet = vbYes) Then
        sockListen.Close
        sockListen.LocalPort = Val(txtport.Text) + 5
        sockListen.Accept requestID
        'lblstatus.Caption = "conectado!"
        'btnenviar.Enabled = True
        hash = md5Test.DigestStrToHexStr(txtsenha.Text)
        sockListen.SendData ("<nom>" & hash & txtnomeusuario.Text)
    'End If

End Sub

Private Sub txtentrada_Change()

    txtcrypt.Text = EncryptData(txtentrada.Text)
    lblhash.Caption = md5Test.DigestStrToHexStr(txtentrada.Text & txtcrypt.Text)
    If (txtcrypt.Text = "") Then lblhash.Caption = ""

End Sub

Private Sub txtsaida_Change()

    txtsaida.SelStart = Len(txtsaida)   'rolar automatico
```

```
End Sub

Private Sub Winsock1_Close()

    lblstatus.Caption = "conexão encerrada!"
    btnenviar.Enabled = False
    privateKey = 0
    aguardarConexao

End Sub



Private Sub sockListen_DataArrival(ByVal bytesTotal As Long)

    Dim entrada As String
    Dim comando As String
    Dim mensagem As String
    Dim hash As String
    Dim msgCrypt As String
    entrada = ""

    sockListen.GetData entrada
    'MsgBox entrada
    comando = Left(entrada, 5)

    Select Case comando
      Case "<err>":
          MsgBox Mid(entrada, 6, Len(entrada) - 5)
          sockListen.Close
      Case "<usr>":
          'pega o que veio depois do comando
          meuamigo = Mid(entrada, 6, Len(entrada) - 5)
          'pergunta se o usuario aceita a conexão
          iRet = MsgBox("Aceitar conexão de " & meuamigo & "?", vbQuestion + vbYesNo, "Solicitação
de conexão")

          If (iRet = vbYes) Then
             'se aceitou, envia o comando pro servidor dizendo que aceitou "<act>"
             lblstatus.Caption = "conectado!"
             btnenviar.Enabled = True
             sockListen.SendData "<act>"
          Else
             'se não, fecha a conexão e aguarda uma próxima
             sockListen.Close
             Winsock1.Close
             aguardarConexao
          End If

      Case "<key>":       'recebemos uma nova chave

          privateKey = Mid(entrada, 22, 16)
          privateKeyd = Mid(entrada, 6, 16)
          encryptKey = keyToNum(privateKey)
          decriptKey = keyToNum(privateKeyd)
          txtTemp.Text = "nova chave recebida: " & Mid(entrada, 6, 32) ' & " [" & encryptKey & ":" &
decriptKey & "]"
          lblstatus.Caption = "Contectado!"
          btnenviar.Enabled = True
          sockListen.SendData "<ok2>"

       Case Else           'se não for comando, então é mensagem recebida
          hash = UCase(Mid(entrada, 1, 32))
          msgCrypt = Mid(entrada, 33, Len(entrada) - 32)
```

```
            mensagem = DecryptData(msgCrypt)
            lblHashIn.Caption = hash

            If (hash <> md5Test.DigestStrToHexStr(mensagem & msgCrypt)) Then
               txtcrypt.Text = msgCrypt
               txtsaida.Text = txtsaida.Text & vbCrLf & "A mensagem não passou no teste de integridade"
& vbCrLf
            Else
               txtsaida.Text = txtsaida.Text & vbCrLf & meuamigo & ": " & mensagem
               'txtcrypt.Text = entrada
               txtcrypt.Text = entrada
            End If


      End Select

End Sub

Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)

      Dim entrada As String
      Dim comando As String
      Dim hash As String
      Dim msgCrypt As String

      Winsock1.GetData entrada
      comando = Left(entrada, 5)

      Select Case comando
         Case "<err>":
            MsgBox Mid(entrada, 6, Len(entrada) - 5)
            Winsock1.Close
         Case "<usr>":
            meuamigo = Mid(entrada, 6, Len(entrada) - 5)
         Case "<ok1>":   'caso conexão ok
            hash = md5Test.DigestStrToHexStr(txtsenha.Text)
            Winsock1.SendData ("<nom>" & hash & txtnomeusuario.Text)
         Case "<key>": 'recebemos uma nova chave
            privateKey = Mid(entrada, 6, 16)
            privateKeyd = Mid(entrada, 22, 16)
            encryptKey = keyToNum(privateKey)
            decriptKey = keyToNum(privateKeyd)
            txtTemp.Text = "nova chave recebida: " & Mid(entrada, 6, 32) '& " [" & encryptKey & ":" &
decriptKey & "]"
            btnenviar.Enabled = True
            lblstatus.Caption = "Conectado!"
         Case Else   'se não for comando, então é mensagem recebida
            hash = UCase(Mid(entrada, 1, 32))
            msgCrypt = Mid(entrada, 33, Len(entrada) - 32)
            mensagem = DecryptData(msgCrypt)
            lblHashIn.Caption = hash

            If (hash <> md5Test.DigestStrToHexStr(mensagem & msgCrypt)) Then
               txtcrypt.Text = msgCrypt
               txtsaida.Text = txtsaida.Text & vbCrLf & "A mensagem não passou no teste de integridade"
& vbCrLf
            Else
               txtsaida.Text = txtsaida.Text & vbCrLf & meuamigo & ": " & mensagem
               txtcrypt.Text = entrada
            End If
      End Select

End Sub
```

```
Public Function EncryptData(Text As String) As String

    Dim Counter As Integer  'contador para o laço while
    Dim DayKey As Integer   'chave
    Dim retorno As String   'o que será retornado
    Dim Encrypt As String   'caractere criptografado
    Dim TextLen As Long     'tamanho do caractere

    'se não tiver nada pra fazer, para por aqui
    If Text = "" Then
       EncryptData = ""
       Exit Function
    End If

    'pega valor da privatekey
       DayKey = encryptKey

    'inicia variáveis
       retorno = ""              'limpa saída
       TextLen = Len(Text)       'tamanho do texto
       Counter = 1               'contador inicia em 1

    'criptografa caractere por caractere
    While Counter <= TextLen
       DoEvents
       Encrypt = (Chr((Asc(Mid(Text, Counter, 1)) + DayKey) Mod 256))
       retorno = retorno & Encrypt     'acumulador
       Counter = Counter + 1           'para o próximo caractere
    Wend

    EncryptData = retorno

End Function

Public Function DecryptData(Text As String) As String

    Dim Counter As Integer      'contador para o laço while
    Dim DayKey As Integer       'chave
    Dim RetData As String       'o que será retornado
    Dim Decrypt As String       'caractere descriptografado
    Dim DecryptNum As Integer   'código ascii do caractere descriptografado
    Dim TextLen As Long         'tamanho do texto

    'se não tem nada pra fazer, sair!
       If Text = "" Then
          Exit Function
       End If

    'pega o valor da variável privatekey
       DayKey = decriptKey


    TextLen = Len(Text)         'tamanho total do texto
    Counter = 1                 ' dessa vez vamos começar do primeiro caractere

    While Counter <= TextLen    'caminha por todo o texto
       DoEvents                 'poupar processador

       DecryptNum = (Asc(Mid(Text, Counter, 1)) - DayKey) Mod 256
       If DecryptNum < 0 Then
          DecryptNum = 255 + DecryptNum
       Else
          DecryptNum = DecryptNum Mod 256
       End If
```

```
        Decrypt = Right(Chr(DecryptNum), 1)
        RetData = RetData & Decrypt 'acumulador

        Counter = Counter + 1
    Wend

    DecryptData = RetData

End Function

Public Function keyToNum(key As String) As Integer

    Dim cod As String
    Dim Num As Long

    key = "&h" & key

    Num = Val(key)
    If (Num < 0) Then Num = Num * -1
    Num = Num Mod 6000 '65536 = (256 * 256) = tamanho máximo de uma variavel integer

    'MsgBox num
    keyToNum = Num

End Function
```

APÊNDICE B

Servidor MiConversa.

```
' declaração de variáveis públicas utilizadas no programa

Public chaveA As String 'utilizada para gerar a primeira parte da chave de criptografia
Public chaveB As String 'utilizada para gerar a segunda parte da chave de criptografia
Public user1 As String  'utilizada para armazenar o nome do primeiro usuário da conversa
Public user2 As String  'utilizada para armazenar o nome do segundo usuário da conversa
Dim md5Test As MD5      'utilizada para gerar hash MD5


Private Sub btnaddusers_Click()     'executado quando o usuário clica no botão de adicionar usuários


   ' se não digitou nome de usuario:
   If (txtadduser.Text = "") Then
      MsgBox ("Você precisa digitar um nome de usuário")  'exibe mensagem solicitando pra digitar
nome de usuário
      txtadduser.SetFocus                    'ajusta o foco para o txtadduser
      Exit Sub                               'sai da função sem fazer nada mais
   End If

   ' se o usuário já existe:
   If (usuarioExiste(txtadduser.Text)) Then
      MsgBox ("Este nome de usuário já está cadastrado")  'exibe mensagem dizendo que o usuário já
existe
      txtadduser.SetFocus                    'ajusta o foco para txtadduser
      Exit Sub                               'sai da função sem fazer nada mais
   End If

   ' se o campo senha está em branco:
   If (txtsenha.Text = "") Then
      MsgBox "Digite uma senha"              'exibe mensagem dizendo que precisa digitar uma
senha
      txtsenha.SetFocus                      'ajusta o foco para o txtsenha
      Exit Sub                               'sai da função sem fazer nada mais
   End If

   ' se confirmação de senha não bate:
   If (txtsenha.Text <> txtsenha2.Text) Then
      MsgBox "A confirmação de senha não é válida"  'exibe mensagem dizendo que a
confirmação de senha não é igual à senha
      txtsenha2.Text = ""                    'limpa a confirmação de senha
      txtsenha2.SetFocus                     'ajusta o foco para o txtsenha2 (confirmação de senha)
      Exit Sub                               'sai da função sem fazer nada mais
   End If

   'a execução da função chegará a esta linha caso passe em todos os testes acima
   lstusers.AddItem (txtadduser.Text)        'leva o nome digitado para a lista de usuários
cadastrados
   lstpass.AddItem (md5Test.DigestStrToHexStr(txtsenha.Text))  'lista de senhas (escondida para o
usuário) recebe hash da senha
   txtadduser.Text = ""                      'limpa txtadduser
   txtsenha.Text = ""                        'limpa txtsenha
   txtsenha2.Text = ""                       'limpa txtsenha2 (confirmação de senha)
   txtadduser.SetFocus                       'ajusta o foco para txtadduser (campo de nome de
usuário)

End Sub
```

```
Private Sub btnremoveusers_Click()  'executa sempre que o usuário clicar no botão de remover usuários

   If (lstusers.ListIndex <> -1) Then            'se ítem selecionado <> -1 (alguém foi selecionado)
      lstusers.RemoveItem (lstusers.ListIndex)    'remove o ítem selecionado da lista de usuários
      lstpass.RemoveItem (lstpass.ListIndex)      'remove o ítem selecionado da lista de senhas
   Else                                           'se ítem selecionado for igual a -1 (nada foi selecionado)
      MsgBox ("você precisa selecionar um usuário na lista")  'exibe mensagem dizendo que é necessário selecionar alguém
   End If

End Sub

Private Sub cmdConfigOk_Click() 'quando o usuário clicar no botão "OK" na página de configuração


   If mnuaguardarconexao.Checked = True Then 'se já estiver aguardando, fecha e abre de novo
      mnuaguardarconexao_Click        'uma vez para fechar (chamando a função click do menu mnuaguardarconexao)
      mnuaguardarconexao_Click    'mais uma vez para abrir
   Else    'se não, apenas abre
      mnuaguardarconexao_Click
   End If

   gravaUsers                  'chama função que grava os nomes das listas no banco de dados

   frmConfig.Visible = False     'esconde frmconfig (propriedade "visible" do frame de configuração recebe "false"

End Sub

Private Sub Form_Load()        'executado sempre que o programa é iniciado

   loadUsers                   'carregar usuarios do banco de dados para a lista de usuários cadastrados
   mnuaguardarconexao_Click    'chama função do botão de aguardar conexão responsável por fazer o programa aguardar conexão de um cliente
   frmConfig.Left = 0          'posição X do frame recebe 0 (canto esquerdo da janela)
   chaveA = ""                 'inicia variáveis chaveA com valor vazio
   chaveB = ""                 'inicia variáveis chaveB com valor vazio
   Set md5Test = New MD5       'cria objeto da classe MD5 para usarmos quando for necessário criar hash MD5

End Sub

Private Sub Form_Resize()      'executado toda vez que se redimensiona o formulário, na inicialização do programa e quando é minimizado/restaurado

   On Error Resume Next        'em caso de erro, continue assim mesmo!

   'reajusta tamanho do campo de texto de saída e frame de configurações
   txtstatus.Width = Me.Width - 130        'ajusta sua largura para a largura atual da janela - 130
   txtstatus.Height = Me.Height - 850      'ajusta sua altura para a altura atual da janela - 850
   frmConfig.Width = Me.Width - 130        'ajusta sua largura para a largura atual da janela - 130
   frmConfig.Height = Me.Height - 850      'ajusta sua altura para a altura atual da janela - 850

   cmdConfigOk.Left = Me.Width - 2000      'posição X do botão de OK da configuração será a largura da janela - 2000
   cmdConfigOk.Top = Me.Height - 1700      'posição Y do botão de OK da configuração será a altura da janela - 1700



End Sub
```

```
Private Sub lstpass_KeyDown(KeyCode As Integer, Shift As Integer)  'executado toda vez que uma tecla é pressionada sobre a lista de senhas

lstusers.ListIndex = lstpass.ListIndex  'usuário selecionado será o de mesma posição da senha selecionada

End Sub

Private Sub lstpass_KeyUp(KeyCode As Integer, Shift As Integer)  'executado toda vez que uma tecla é solta sobre a lista de senhas

lstusers.ListIndex = lstpass.ListIndex  'usuário selecionado será o de mesma posição da senha selecionada

End Sub

Private Sub lstpass_MouseMove(Button As Integer, Shift As Integer, x As Single, Y As Single) 'executado toda vez que moverem o mouse sobre a lista de senhas

lstusers.ListIndex = lstpass.ListIndex  'usuário selecionado será o de mesma posição da senha selecionada

End Sub

Private Sub lstusers_KeyDown(KeyCode As Integer, Shift As Integer) 'executado toda vez que pressionarem uma tecla sobre a lista de usuários

lstpass.ListIndex = lstusers.ListIndex  'senha selecionada será a de mesma posição do usuário selecionado

End Sub

Private Sub lstusers_KeyUp(KeyCode As Integer, Shift As Integer) 'executado toda vez que soltarem uma tecla sobre a lista de usuários

lstpass.ListIndex = lstusers.ListIndex  'senha selecionada será a de mesma posição do usuário selecionado

End Sub

Private Sub lstusers_MouseDown(Button As Integer, Shift As Integer, x As Single, Y As Single) 'executado toda vez que presionarem uma tecla do mouse sobre a lista de usuários

lstpass.ListIndex = lstusers.ListIndex  'senha selecionada será a de mesma posição do usuário selecionado

End Sub


Private Sub lstpass_MouseDown(Button As Integer, Shift As Integer, x As Single, Y As Single) 'executado toda vez que presionarem uma tecla do mouse sobre a lista de senhas

lstusers.ListIndex = lstpass.ListIndex  'usuário selecionado será o de mesma posição da senha selecionada

End Sub

Private Sub lstusers_MouseMove(Button As Integer, Shift As Integer, x As Single, Y As Single) 'executado toda vez que moverem o mouse sobre a lista de nomes

lstpass.ListIndex = lstusers.ListIndex  'senha selecionada será a de mesma posição do usuário selecionado

End Sub
```

```
Private Sub mnuaguardarconexao_Click()  'executado toda vez que o usuário clica no menú de aguardar conexão

   'se propriedade "checked" do menu aguardarconexao for true (certamente está aguardando conexão)
   If (mnuaguardarconexao.Checked = True) Then
      Winsock1.Close                'fecha winsock1
      mnuaguardarconexao.Checked = False  'ajusta "checked" de aguardarconexao como false
      status ("conexão encerrada")          'diz ao usuário que a conexão foi encerrada chamando a função "status"
      Exit Sub                      'sai da função sem fazer nada mais
   End If

   'inverte valor de "checked"
   mnuaguardarconexao.Checked = Not mnuaguardarconexao.Checked

   On Error GoTo erro1     'se der erro vai para linha erro1: e mostra mensagem

   chaveA = ""            'limpa chaveA
   chaveB = ""            'limpa chaveB
   Winsock2.Close         'fecha winsock1
   Winsock1.Close         'fecha winsock2
   Winsock1.LocalPort = txtport.Text   'ajusta porta local de winsock1
   Winsock1.Listen                'diz ao winsock1 que deve aguardar uma conexão
   status ("Aguardando conexão na porta " & txtport.Text)  'diz ao usuário que estamos aguardando conexão

GoTo fim                  'se não deu erro nenhum, vai para fim: e termina função
erro1:
   'mensagem de erro caso tenha ocorrido algum erro ao tentar fazer winsock1 escutar
   MsgBox "Porta " & Winsock1.LocalPort & "em uso", vbCritical, "Porta Ocupada"
fim:

End Sub

Private Sub status(mensagem As String)      'função status, é executada sempre que for chamada no código fonte

   ' coloca uma mensagem na saida para usuario ler
   'propriedade text de txtstatus recebe ela mesma mais a mensagem recebida como parâmetro
   txtstatus.Text = txtstatus.Text & mensagem & vbCrLf

End Sub


Private Sub mnuConfigConexao_Click()    'executa sempre que o usuário clicar no menu "conexão"

   'faz frmconfig (objeto que contem os controles de conexão) ficar visivel ao usuário
   frmConfig.Visible = True

End Sub


Private Sub txtstatus_Change()          'executada sempre que txtstatus (usado para mostrar mensagens ao administrador do servidor) receber um texto novo

   'rola barra de rolagem para a última linha mostrada
   txtstatus.SelStart = Len(txtstatus)

End Sub


Private Sub Winsock1_Close()          'executada sempre que winsock1 perde a conexão
```

```
    'apenas para mostrar o que aconteceu (winsock1 fechou conexão)
    status ("Conexão encerrada!")

    'volta a aguardar conexão
    If mnuaguardarconexao.Checked = True Then 'se já estiver aguardando, fecha e abre de novo
        mnuaguardarconexao_Click    'uma vez pra fechar
        mnuaguardarconexao_Click    'mais uma vez pra abrir
    Else    'se não, apenas abre
        mnuaguardarconexao_Click    'só uma pra abrir pois já estava fechada
    End If

End Sub

Private Sub Winsock1_Connect()      'executada sempre que o winsock1 estabelece uma conexão

    'apenas para mostrar o que aconteceu (uma conexão foi feita no winsock1)
    status ("Conexão1 Estabilizada")

End Sub

Private Sub Winsock1_ConnectionRequest(ByVal requestID As Long) 'executada sempre que
winsock1 recebe solicitação de conexão

    'gera as chaves
    chaveA = GenerateKey    'gera chave de 16Bytes e armazena em ChaveA
    chaveB = GenerateKey    'gera chave de 16Bytes e armazena em ChaveB

    ' VAMOS AGORA TENTAR CONECTAR NO CLIENTE "B"
    If Winsock2.State <> sckClosed Then 'se estado de winsock2 for diferente de fechado (aberto)
então:
        Winsock2.Close              'fecha conexão em winsock2
    End If


    Winsock1.Close                  'reinicia conexão em winsock1
    Winsock1.LocalPort = Val(txtport.Text) + 5  'reajusta porta local do winsock1 para a porta que o
usuário escolheu + 5
    Winsock1.Accept requestID           'aceita a conexão
    Winsock1.SendData ("<ok1>")         'envia ao winsock1 o comando "<ok1>"

    'mostra mensagem na janela de saída para o administrador do servidor ver o que aconteceu
    status ("Solicitação para Conexão1 aceita na porta " & Winsock1.LocalPort)

End Sub

Private Sub Winsock1_DataArrival(ByVal bytesTotal As Long)  'executada sempre que o winsock1
recebe alguma informação
    'declaração de variáveis
    Dim entrada As String   'usada para armazenar o que foi recebido
    Dim comando As String   'utilizada para armazenar os 5 primeiros caracteres a fim de verificar se é
um comando conhecido
    Dim nome As String      'utilizada para armazenar um nome de usuário caso receba um

    Winsock1.GetData entrada    'recupera o que foi recebido pelo winsock1
    comando = Left(entrada, 5)  'pega os 5 primeiros caracteres a fim de que se verifique se é um
comando

    Select Case comando         'caso comando seja:
        Case "<nom>":           'se o comando for o nome de usuario
            'captura o que estiver após o "<key>"
                nome = Mid(entrada, 6, Len(entrada) - 5)
            'guarda o nome na variavel user1
                user1 = Mid(nome, 33, Len(nome) - 32)
```

```
        'verifica se o usuario está cadastrado pela função "permitido"
            If (permitido(nome)) Then    'chama a função "permitido" para verificar se o "nome" está cadastrado no banco de dados. Se tiver cadastrado:
                'deixa uma mensagem na caixa de saída para o administrador do servidor avisando
                status ("usuario '" & user1 & "' autorizado")
                'envia para o winsock1 o comando <key> junto com a chaveA e chaveB
                Winsock1.SendData ("<key>" & chaveA & chaveB) 'devolve nova chave
                'mostra mensagem para o administrador do servidor que foi enviado a chave
                status "enviando chave: '" & chaveA & chaveB & "' para sock1"
                'pede conexão ao winsock2 pelo ip e porta especificada pelo administrador do servidor
                Winsock2.Connect txtip.Text, txtport2.Text
                'mostra mensagem avisando que estamos tentando conectar no outro cliente
                status (vbCrLf & "tentando conexão em " & txtip.Text & " porta: " & txtport2.Text)
            Else    'se o usuário não estiver autorizado:
                'mostra mensagem para o administrador dizendo que o usuário não está autorizado
                status ("usuario '" & Mid(nome, 33, Len(nome) - 32) & "' não autorizado")
                'envia ao winsock1 o comando "<err>" junto com a mensagem "usuário não autorizado"
                Winsock1.SendData ("<err>usuario não autorizado")
                'fecha a conexão
                status ("conexão encerrada")
            End If

            'se o estado do winsock2 for "sckConnected" (conectado):
            If Winsock2.State = sckConnected Then
                Winsock2.SendData ("<key>" & chaveA & chaveB)   'envia a nova chave para o winsock2
            End If

        Case Else  'caso não for nenhum comando:
            'mostra na tela a mensagem recebida
            status (vbCrLf & "sock1 -> sock2 '" & entrada & "'")

            'se winsock2 estiver conectado, repassa a informação recebida para o mesmo
            If (Winsock2.State = sckConnected) Then
                Winsock2.SendData (entrada) 'repassa para o winsock2 a mensagem recebida (pelo winsock1)
            End If

    End Select

End Sub

Private Sub Winsock1_Error(ByVal Number As Integer, Description As String, ByVal Scode As Long, ByVal Source As String, ByVal HelpFile As String, ByVal HelpContext As Long, CancelDisplay As Boolean)
'é executado sempre que houver um erro no winsock1

    status ("Erro conexão") 'manda a mensagem para a caixa de saída dizendo que houve um erro de conexão

End Sub


Private Sub Winsock2_DataArrival(ByVal bytesTotal As Long)  'é executado sempre que o winsock2 recebe uma informação
    'declaração de variáveis
    Dim entrada As String   'string usada para armazenar a informação recebida
    Dim comando As String   '5 primeiros caracteres da entrada para ver se é um comando
    Dim nome As String      'usada para armazenar o nome de usuário caso seja recebido um


    Winsock2.GetData entrada    'recupera o que foi recebido e armazena na variável "entrada"
    comando = Left(entrada, 5)  'pega os 5 primeiros caracteres para ver se é um comando

    Select Case comando         'caso o comando (5 primeiros caracteres) seja:
```

```
      Case "<ok2>":          '<ok2>
         'envia para o winsock1 (pra quem solicitou a conexão) o comando "<usr>" junto com o nome
do usuario que aceitou a conexão
         Winsock1.SendData "<usr>" & user2
      Case "<act>":          '"comando <act> significa que o usuário 2 aceitou a conexão
         'envia para o winsock2 o comando "<key>" junto da chaveA e chaveB
         Winsock2.SendData ("<key>" & chaveA & chaveB) 'devolve nova chave
         'insere uma mensagem na tela de saída avisando ao administrador do servidor
         status "enviando chave: '" & chaveA & chaveB & "' para sock2"
      Case "<nom>":          'comando "<nom>" significa que o nome do cliente foi recebido
         'captura o que estiver após o "<nom>"
            nome = Mid(entrada, 6, Len(entrada) - 5)
         'guarda na variável user2
            user2 = Mid(nome, 33, Len(nome) - 32)
         'verifica se o usuario está cadastrado chamando a função "permitido"
            If (permitido(nome)) Then   'se estiver cadastrado no sistema:
               'mostra mensagem para administrador do servidor
               status ("usuario '" & user2 & "' autorizado")
               'envia para winsock2 o comando <usr> seguido do nome do cliente que solicitou a
conexão a fim de
               'que o cliente 2 aceite ou não
               Winsock2.SendData "<usr>" & user1
            Else    'se não estiver cadastrado
               'avisa o administrador do servidor que o usuário não está autorizado e denuncia o nome
não autorizado
               status ("usuario '" & Mid(nome, 33, Len(nome) - 32) & "' não autorizado")
               'envia para o winsock2 o comando "<err>" (erro) seguido da mensagem de erro
               Winsock2.SendData ("<err>usuario não autorizado")
               'fecha conexão com o winsock1, a conexão com o winsock2 será feita logo após ele
receber o erro (comando <err>)
               Winsock1.Close
               'avisa ao administrador que a conexão foi encerrada
               status ("conexão encerrada")
            End If

         'manda a nova chave para a outra parte
            If (Winsock1.State = sckConnected) Then 'se estado do winsock1 for conectado:
               'envia para o winsock1 o comando "<key>" seguido de chaveA e chaveB
               Winsock1.SendData ("<key>" & chaveA & chaveB)
            End If

      Case Else 'se não for nenhum comando, então é uma mensagem e deve ser repassada
         'mostra na tela a mensagem recebida
            status (vbCrLf & "sock2 -> sock1 '" & entrada & "'")
         'se winsock2 estiver conectado, repassa a informação recebida para o mesmo
            If (Winsock1.State = sckConnected) Then Winsock1.SendData (entrada)
   End Select

End Sub

Private Function permitido(nome As String) As Boolean    'executada sempre que chamada por
alguma linha de comando
   'função que verifica se um nome se encontra na lista de nomes permitidos
   'declaração de variáveis
   Dim cont As Integer 'será usada no laço for

   'anda linha por linha no listbox
   For cont = 0 To lstusers.ListCount
      'se valor recebido em letras maiúsculas (UCase) for igual a senha e usuário na posição "cont"
em letras maiúsculas:
      If (UCase(lstpass.List(cont) & lstusers.List(cont)) = UCase(nome)) Then
         'retorna verdadeiro
         permitido = True
         'sai da função
```

```
            Exit Function
        End If
    Next

    'se não encontrar, vai chegar nessa linha e será retornado false
    permitido = False

End Function

Private Function usuarioExiste(nome As String) As Boolean    'executada sempre que chamada por alguma linha de comando
    'função que verifica se um nome é existente no banco de dados
    'declaração de variáveis
    Dim cont As Integer 'será usada no laço for

    'procura linha por linha do listbox
    For cont = 0 To lstusers.ListCount
        'se valor recebido em letras maiúsculas (UCase) for igual a usuário na posição "cont" em letras maiúsculas:
        If (UCase(lstusers.List(cont)) = UCase(nome)) Then
            'se encontrar, retorna true
            usuarioExiste = True
            'sai da função
            Exit Function
        End If
    Next

    'se não encontrar, vai chegar nessa linha e será retornado false
    usuarioExiste = False

End Function

Public Sub gravaUsers() 'é executada toda vez que o usuário clica no botão "ok" na tela de configuração
    'esta função salva os nomes no banco de dados (arquivo de texto)
    Dim iARQ As Integer    'variável usada como ponteiro de arquivo

    iARQ = FreeFile 'evita um erro
    Open "usuarios.txt" For Output As iARQ  'abre arquivo para gravação
        'caminha por toda a lista de nomes
        For cont = 0 To lstusers.ListCount - 1
            Print #iARQ, lstpass.List(cont) + lstusers.List(cont)    'salva a senha junto o nome no final, em cada linha do arquivo
        Next

    'fecha arquivo
    Close iARQ


End Sub

Function loadUsers() As String  'função executada no carregamento do programa para recuperar os valores do banco de dados
    'declaração de variáveis
    Dim iARQ As Integer         'ponteiro de arquivo
    Dim sLinha As String        'linha recuperada do arquivo

    iARQ = FreeFile 'evita um erro
    Open "usuarios.txt" For Input As iARQ   'abre arquivo para leitura

    Do While Not EOF(iARQ)  'faça enquanto não é o fim do arquivo "iARQ" (EOF = end of file)
        Line Input #iARQ, sLinha        'lê linha do arquivo e armazena na variável sLinha
        'adiciona à lista de usuários a sub string de sLinha começando do caractere nº 33 (pulando os 32 caracteres
```

```
        'que são o hash da senha) e pegando o numero de caracteres da linha - 32 (tamanho da linha sem o hash da senha)
        lstusers.AddItem (Mid(sLinha, 33, Len(sLinha) - 32))
        'adiciona à lista de senhas a sub string da variável sLinha que começa do caractere 1 e de tamanho 32 caracteres
        '(os primeiros 32 caracteres sempre serão o hash da senha)
        lstpass.AddItem (Mid(sLinha, 1, 32))
    Loop

    'Fecha o arquivo
    Close iARQ

End Function

Public Function GenerateKey() As String 'função chamada toda vez que precisa gerar uma chave de criptografia
    'declaração de variáveis
    Dim randNum As Integer  'variável usada para receber o número aleatório
    Dim acum As String      'variável usada para acumular o resultado final
    Dim cont As Integer     'variável usada para controle de voltas do laço for
    Dim vet                 'vetor que armazenará os valores de 0 a 9 e A a F (para gerar uma chave hexadecimal)

    'variável vet recebe todos os valores separados por ','
    vet = "0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F"
    'transforma esses valores separados por ',' em valores do vetor e armazena na variável "vet" usando a função "Split"
    vet = Split(vet, ",")

    Randomize       'inicia a geração de números aleatórios
    acum = ""       'limpa a variável "acum"
    'laço for executa 16 iterações para acumular 16 caracteres dos que estão no vetor "vet"
    For cont = 1 To 16 'chave de 16 bytes
        'gera um numero aleatório, multiplica por 100, divide seu resultado por 16
        'e acumula o resto na variável randNum
        randNum = Rnd() * 100 Mod 16 'base 16 (hexadecimal)
        'acumula o valor de "vet" na posição (acum)
        acum = acum & vet(randNum)
    Next

    'retorna valor de acum pela função
    GenerateKey = acum

End Function
```

APÊNDICE C

MD5 utilizado tanto para o Servidor MiConversa como para o MiConversa, o mesmo não foi de minha autoria sendo retirado do link: www.bullzip.com, acesso em 10/08/2008.

```
Option Explicit

'/**************************************************************************
' *  Copyright (C) 2000 by Robert Hubley.                                  *
' *  All rights reserved.                                                  *
' *                                                                        *
' *  This software is provided ``AS IS'' and any express or implied        *
' *  warranties, including, but not limited to, the implied warranties of  *
' *  merchantability and fitness for a particular purpose, are disclaimed. *
' *  In no event shall the authors be liable for any direct, indirect,     *
' *  incidental, special, exemplary, or consequential damages (including, but  *
' *  not limited to, procurement of substitute goods or services; loss of use, *
' *  data, or profits; or business interruption) however caused and on any  *
' *  theory of liability, whether in contract, strict liability, or tort    *
' *  (including negligence or otherwise) arising in any way out of the use of  *
' *  this software, even if advised of the possibility of such damage.      *
' *                                                                        *
' **************************************************************************
'
'  CLASS: MD5
'
'  DESCRIPTION:
'    This is a class which encapsulates a set of MD5 Message Digest functions.
'    MD5 algorithm produces a 128 bit digital fingerprint (signature) from an
'    dataset of arbitrary length.  For details see RFC 1321 (summarized below).
'    This implementation is derived from the RSA Data Security, Inc. MD5 Message-Digest
'    algorithm reference implementation (originally written in C)
'
'  AUTHOR:
'    Robert M. Hubley 12/1999
'
'
'  NOTES:
'     Network Working Group                          R. Rivest
'     Request for Comments: 1321     MIT Laboratory for Computer Science
'                                  and RSA Data Security, Inc.
'                                          April 1992
'
'
'                       The MD5 Message-Digest Algorithm
'
'     Summary
'
'        This document describes the MD5 message-digest algorithm. The
'        algorithm takes as input a message of arbitrary length and produces
'        as output a 128-bit "fingerprint" or "message digest" of the input.
'        It is conjectured that it is computationally infeasible to produce
'        two messages having the same message digest, or to produce any
'        message having a given prespecified target message digest. The MD5
'        algorithm is intended for digital signature applications, where a
'        large file must be "compressed" in a secure manner before being
'        encrypted with a private (secret) key under a public-key cryptosystem
'        such as RSA.
```

```
'
'       The MD5 algorithm is designed to be quite fast on 32-bit machines. In
'       addition, the MD5 algorithm does not require any large substitution
'       tables; the algorithm can be coded quite compactly.
'
'       The MD5 algorithm is an extension of the MD4 message-digest algorithm
'       1,2]. MD5 is slightly slower than MD4, but is more "conservative" in
'       design. MD5 was designed because it was felt that MD4 was perhaps
'       being adopted for use more quickly than justified by the existing
'       critical review; because MD4 was designed to be exceptionally fast,
'       it is "at the edge" in terms of risking successful cryptanalytic
'       attack. MD5 backs off a bit, giving up a little in speed for a much
'       greater likelihood of ultimate security. It incorporates some
'       suggestions made by various reviewers, and contains additional
'       optimizations. The MD5 algorithm is being placed in the public domain
'       for review and possible adoption as a standard.
'
'       RFC Author:
'       Ronald L.Rivest
'       Massachusetts Institute of Technology
'       Laboratory for Computer Science
'       NE43 -324545    Technology Square
'       Cambridge, MA  02139-1986
'       Phone: (617) 253-5880
'       EMail:    Rivest@ theory.lcs.mit.edu
'
'
'
'  CHANGE HISTORY:
'
'    0.1.0  RMH    1999/12/29      Original version
'
'


'=
'= Class Constants
'=
Private Const OFFSET_4 = 4294967296#
Private Const MAXINT_4 = 2147483647

Private Const S11 = 7
Private Const S12 = 12
Private Const S13 = 17
Private Const S14 = 22
Private Const S21 = 5
Private Const S22 = 9
Private Const S23 = 14
Private Const S24 = 20
Private Const S31 = 4
Private Const S32 = 11
Private Const S33 = 16
Private Const S34 = 23
Private Const S41 = 6
Private Const S42 = 10
Private Const S43 = 15
Private Const S44 = 21


'=
'= Class Variables
'=
Private State(4) As Long
Private ByteCounter As Long
```

```
Private ByteBuffer(63) As Byte


'=
'= Class Properties
'=
Property Get RegisterA() As String
   RegisterA = State(1)
End Property

Property Get RegisterB() As String
   RegisterB = State(2)
End Property

Property Get RegisterC() As String
   RegisterC = State(3)
End Property

Property Get RegisterD() As String
   RegisterD = State(4)
End Property


'=
'= Class Functions
'=

'
' Function to quickly digest a file into a hex string
'
Public Function DigestFileToHexStr(FileName As String) As String
   Open FileName For Binary Access Read As #1
   MD5Init
   Do While Not EOF(1)
      Get #1, , ByteBuffer
      If Loc(1) < LOF(1) Then
         ByteCounter = ByteCounter + 64
         MD5Transform ByteBuffer
      End If
   Loop
   ByteCounter = ByteCounter + (LOF(1) Mod 64)
   Close #1
   MD5Final
   DigestFileToHexStr = GetValues
End Function

'
' Function to digest a text string and output the result as a string
' of hexadecimal characters.
'
Public Function DigestStrToHexStr(SourceString As String) As String
   MD5Init
   MD5Update Len(SourceString), StringToArray(SourceString)
   MD5Final
   DigestStrToHexStr = GetValues
End Function

'
' A utility function which converts a string into an array of
' bytes.
'
Private Function StringToArray(InString As String) As Byte()
   Dim I As Integer
   Dim bytBuffer() As Byte
```

```
    ReDim bytBuffer(Len(InString))
    For I = 0 To Len(InString) - 1
        bytBuffer(I) = Asc(Mid(InString, I + 1, 1))
    Next I
    StringToArray = bytBuffer
End Function

'
' Concatenate the four state vaules into one string
'
Public Function GetValues() As String
    GetValues = LongToString(State(1)) & LongToString(State(2)) & LongToString(State(3)) &
LongToString(State(4))
End Function

'
' Convert a Long to a Hex string
'
Private Function LongToString(Num As Long) As String
        Dim a As Byte
        Dim b As Byte
        Dim c As Byte
        Dim d As Byte

        a = Num And &HFF&
        If a < 16 Then
            LongToString = "0" & Hex(a)
        Else
            LongToString = Hex(a)
        End If

        b = (Num And &HFF00&) \ 256
        If b < 16 Then
            LongToString = LongToString & "0" & Hex(b)
        Else
            LongToString = LongToString & Hex(b)
        End If

        c = (Num And &HFF0000) \ 65536
        If c < 16 Then
            LongToString = LongToString & "0" & Hex(c)
        Else
            LongToString = LongToString & Hex(c)
        End If

        If Num < 0 Then
            d = ((Num And &H7F000000) \ 16777216) Or &H80&
        Else
            d = (Num And &HFF000000) \ 16777216
        End If

        If d < 16 Then
            LongToString = LongToString & "0" & Hex(d)
        Else
            LongToString = LongToString & Hex(d)
        End If

End Function

'
' Initialize the class
'   This must be called before a digest calculation is started
'
Public Sub MD5Init()
```

```
    ByteCounter = 0
    State(1) = UnsignedToLong(1732584193#)
    State(2) = UnsignedToLong(4023233417#)
    State(3) = UnsignedToLong(2562383102#)
    State(4) = UnsignedToLong(271733878#)
End Sub

'
' MD5 Final
'
Public Sub MD5Final()
    Dim dblBits As Double

    Dim padding(72) As Byte
    Dim lngBytesBuffered As Long

    padding(0) = &H80

    dblBits = ByteCounter * 8

    ' Pad out
    lngBytesBuffered = ByteCounter Mod 64
    If lngBytesBuffered <= 56 Then
        MD5Update 56 - lngBytesBuffered, padding
    Else
        MD5Update 120 - ByteCounter, padding
    End If


    padding(0) = UnsignedToLong(dblBits) And &HFF&
    padding(1) = UnsignedToLong(dblBits) \ 256 And &HFF&
    padding(2) = UnsignedToLong(dblBits) \ 65536 And &HFF&
    padding(3) = UnsignedToLong(dblBits) \ 16777216 And &HFF&
    padding(4) = 0
    padding(5) = 0
    padding(6) = 0
    padding(7) = 0

    MD5Update 8, padding
End Sub

'
' Break up input stream into 64 byte chunks
'
Public Sub MD5Update(InputLen As Long, InputBuffer() As Byte)
    Dim II As Integer
    Dim I As Integer
    Dim J As Integer
    Dim K As Integer
    Dim lngBufferedBytes As Long
    Dim lngBufferRemaining As Long
    Dim lngRem As Long

    lngBufferedBytes = ByteCounter Mod 64
    lngBufferRemaining = 64 - lngBufferedBytes
    ByteCounter = ByteCounter + InputLen
    ' Use up old buffer results first
    If InputLen >= lngBufferRemaining Then
        For II = 0 To lngBufferRemaining - 1
            ByteBuffer(lngBufferedBytes + II) = InputBuffer(II)
        Next II
        MD5Transform ByteBuffer

        lngRem = (InputLen) Mod 64
```

```
        ' The transfer is a multiple of 64 lets do some transformations
        For I = lngBufferRemaining To InputLen - II - lngRem Step 64
            For J = 0 To 63
                ByteBuffer(J) = InputBuffer(I + J)
            Next J
            MD5Transform ByteBuffer
        Next I
        lngBufferedBytes = 0
    Else
      I = 0
    End If

    ' Buffer any remaining input
    For K = 0 To InputLen - I - 1
        ByteBuffer(lngBufferedBytes + K) = InputBuffer(I + K)
    Next K

End Sub

'
' MD5 Transform
'
Private Sub MD5Transform(Buffer() As Byte)
    Dim x(16) As Long
    Dim a As Long
    Dim b As Long
    Dim c As Long
    Dim d As Long

    a = State(1)
    b = State(2)
    c = State(3)
    d = State(4)

    Decode 64, x, Buffer

    ' Round 1
    FF a, b, c, d, x(0), S11, -680876936
    FF d, a, b, c, x(1), S12, -389564586
    FF c, d, a, b, x(2), S13, 606105819
    FF b, c, d, a, x(3), S14, -1044525330
    FF a, b, c, d, x(4), S11, -176418897
    FF d, a, b, c, x(5), S12, 1200080426
    FF c, d, a, b, x(6), S13, -1473231341
    FF b, c, d, a, x(7), S14, -45705983
    FF a, b, c, d, x(8), S11, 1770035416
    FF d, a, b, c, x(9), S12, -1958414417
    FF c, d, a, b, x(10), S13, -42063
    FF b, c, d, a, x(11), S14, -1990404162
    FF a, b, c, d, x(12), S11, 1804603682
    FF d, a, b, c, x(13), S12, -40341101
    FF c, d, a, b, x(14), S13, -1502002290
    FF b, c, d, a, x(15), S14, 1236535329

    ' Round 2
    GG a, b, c, d, x(1), S21, -165796510
    GG d, a, b, c, x(6), S22, -1069501632
    GG c, d, a, b, x(11), S23, 643717713
    GG b, c, d, a, x(0), S24, -373897302
    GG a, b, c, d, x(5), S21, -701558691
    GG d, a, b, c, x(10), S22, 38016083
    GG c, d, a, b, x(15), S23, -660478335
    GG b, c, d, a, x(4), S24, -405537848
    GG a, b, c, d, x(9), S21, 568446438
```

```
    GG d, a, b, c, x(14), S22, -1019803690
    GG c, d, a, b, x(3), S23, -187363961
    GG b, c, d, a, x(8), S24, 1163531501
    GG a, b, c, d, x(13), S21, -1444681467
    GG d, a, b, c, x(2), S22, -51403784
    GG c, d, a, b, x(7), S23, 1735328473
    GG b, c, d, a, x(12), S24, -1926607734

    ' Round 3
    HH a, b, c, d, x(5), S31, -378558
    HH d, a, b, c, x(8), S32, -2022574463
    HH c, d, a, b, x(11), S33, 1839030562
    HH b, c, d, a, x(14), S34, -35309556
    HH a, b, c, d, x(1), S31, -1530992060
    HH d, a, b, c, x(4), S32, 1272893353
    HH c, d, a, b, x(7), S33, -155497632
    HH b, c, d, a, x(10), S34, -1094730640
    HH a, b, c, d, x(13), S31, 681279174
    HH d, a, b, c, x(0), S32, -358537222
    HH c, d, a, b, x(3), S33, -722521979
    HH b, c, d, a, x(6), S34, 76029189
    HH a, b, c, d, x(9), S31, -640364487
    HH d, a, b, c, x(12), S32, -421815835
    HH c, d, a, b, x(15), S33, 530742520
    HH b, c, d, a, x(2), S34, -995338651

    ' Round 4
    II a, b, c, d, x(0), S41, -198630844
    II d, a, b, c, x(7), S42, 1126891415
    II c, d, a, b, x(14), S43, -1416354905
    II b, c, d, a, x(5), S44, -57434055
    II a, b, c, d, x(12), S41, 1700485571
    II d, a, b, c, x(3), S42, -1894986606
    II c, d, a, b, x(10), S43, -1051523
    II b, c, d, a, x(1), S44, -2054922799
    II a, b, c, d, x(8), S41, 1873313359
    II d, a, b, c, x(15), S42, -30611744
    II c, d, a, b, x(6), S43, -1560198380
    II b, c, d, a, x(13), S44, 1309151649
    II a, b, c, d, x(4), S41, -145523070
    II d, a, b, c, x(11), S42, -1120210379
    II c, d, a, b, x(2), S43, 718787259
    II b, c, d, a, x(9), S44, -343485551


    State(1) = LongOverflowAdd(State(1), a)
    State(2) = LongOverflowAdd(State(2), b)
    State(3) = LongOverflowAdd(State(3), c)
    State(4) = LongOverflowAdd(State(4), d)

'  /* Zeroize sensitive information.
'*/
'  MD5_memset ((POINTER)x, 0, sizeof (x));

End Sub

Private Sub Decode(Length As Integer, OutputBuffer() As Long, InputBuffer() As Byte)
    Dim intDblIndex As Integer
    Dim intByteIndex As Integer
    Dim dblSum As Double

    intDblIndex = 0
    For intByteIndex = 0 To Length - 1 Step 4
        dblSum = InputBuffer(intByteIndex) + _
```

```
                        InputBuffer(intByteIndex + 1) * 256# + _
                        InputBuffer(intByteIndex + 2) * 65536# + _
                        InputBuffer(intByteIndex + 3) * 16777216#
        OutputBuffer(intDblIndex) = UnsignedToLong(dblSum)
        intDblIndex = intDblIndex + 1
    Next intByteIndex
End Sub

'
' FF, GG, HH, and II transformations for rounds 1, 2, 3, and 4.
' Rotation is separate from addition to prevent recomputation.
'
Private Function FF(a As Long, _
                    b As Long, _
                    c As Long, _
                    d As Long, _
                    x As Long, _
                    s As Long, _
                    ac As Long) As Long
    a = LongOverflowAdd4(a, (b And c) Or (Not (b) And d), x, ac)
    a = LongLeftRotate(a, s)
    a = LongOverflowAdd(a, b)
End Function

Private Function GG(a As Long, _
                    b As Long, _
                    c As Long, _
                    d As Long, _
                    x As Long, _
                    s As Long, _
                    ac As Long) As Long
    a = LongOverflowAdd4(a, (b And d) Or (c And Not (d)), x, ac)
    a = LongLeftRotate(a, s)
    a = LongOverflowAdd(a, b)
End Function

Private Function HH(a As Long, _
                    b As Long, _
                    c As Long, _
                    d As Long, _
                    x As Long, _
                    s As Long, _
                    ac As Long) As Long
    a = LongOverflowAdd4(a, b Xor c Xor d, x, ac)
    a = LongLeftRotate(a, s)
    a = LongOverflowAdd(a, b)
End Function

Private Function II(a As Long, _
                    b As Long, _
                    c As Long, _
                    d As Long, _
                    x As Long, _
                    s As Long, _
                    ac As Long) As Long
    a = LongOverflowAdd4(a, c Xor (b Or Not (d)), x, ac)
    a = LongLeftRotate(a, s)
    a = LongOverflowAdd(a, b)
End Function

'
' Rotate a long to the right
'
Function LongLeftRotate(value As Long, bits As Long) As Long
```

```
    Dim lngSign As Long
    Dim lngI As Long
    bits = bits Mod 32
    If bits = 0 Then LongLeftRotate = value: Exit Function
    For lngI = 1 To bits
       lngSign = value And &HC0000000
       value = (value And &H3FFFFFFF) * 2
       value = value Or ((lngSign < 0) And 1) Or (CBool(lngSign And _
             &H40000000) And &H80000000)
    Next
    LongLeftRotate = value
End Function

'
' Function to add two unsigned numbers together as in C.
' Overflows are ignored!
'
Private Function LongOverflowAdd(Val1 As Long, Val2 As Long) As Long
    Dim lngHighWord As Long
    Dim lngLowWord As Long
    Dim lngOverflow As Long

    lngLowWord = (Val1 And &HFFFF&) + (Val2 And &HFFFF&)
    lngOverflow = lngLowWord \ 65536
    lngHighWord = (((Val1 And &HFFFF0000) \ 65536) + ((Val2 And &HFFFF0000) \ 65536) +
lngOverflow) And &HFFFF&
    LongOverflowAdd = UnsignedToLong((lngHighWord * 65536#) + (lngLowWord And &HFFFF&))
End Function

'
' Function to add two unsigned numbers together as in C.
' Overflows are ignored!
'
Private Function LongOverflowAdd4(Val1 As Long, Val2 As Long, val3 As Long, val4 As Long) As
Long
    Dim lngHighWord As Long
    Dim lngLowWord As Long
    Dim lngOverflow As Long

    lngLowWord = (Val1 And &HFFFF&) + (Val2 And &HFFFF&) + (val3 And &HFFFF&) + (val4 And
&HFFFF&)
    lngOverflow = lngLowWord \ 65536
    lngHighWord = (((Val1 And &HFFFF0000) \ 65536) + _
                ((Val2 And &HFFFF0000) \ 65536) + _
                ((val3 And &HFFFF0000) \ 65536) + _
                ((val4 And &HFFFF0000) \ 65536) + _
                lngOverflow) And &HFFFF&
    LongOverflowAdd4 = UnsignedToLong((lngHighWord * 65536#) + (lngLowWord And &HFFFF&))
End Function

'
' Convert an unsigned double into a long
'
Private Function UnsignedToLong(value As Double) As Long
       If value < 0 Or value >= OFFSET_4 Then Error 6 ' Overflow
       If value <= MAXINT_4 Then
         UnsignedToLong = value
       Else
         UnsignedToLong = value - OFFSET_4
       End If
     End Function

'
' Convert a long to an unsigned Double
```

```
'
Private Function LongToUnsigned(value As Long) As Double
        If value < 0 Then
          LongToUnsigned = value + OFFSET_4
        Else
          LongToUnsigned = value
        End If
End Function
```