Centro Universitário de Brasília – UniCEUB
Faculdade de Exatas e Tecnologia - FAET
Curso de Engenharia da Computação

Francis Vilaça Santos

Projeto de autenticidade em um sistema de comunicação ponto a ponto

Brasília
2006

Francis Vilaça Santos

Projeto de autenticidade em um sistema de comunicação ponto a ponto

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 2006

Dedicatória

À Deus,
Por ter me dado força nesta caminhada.
À Minha família,
Pelo apoio e estimulo que me ofereceram.
À Engenheira Priscila Cortez,
Por ter acreditado e lutado comigo.
Em memória a Joabe Teles,
Por sua amizade e vontade de me ver formado.
Dedico-lhes essa conquista como gratidão.

Agradecimentos

Aos meus pais, Manoel e Fátima, que me deram a oportunidade de chegar até aqui, priorizando a minha educação durante todos esses anos e me ajudando com todo apoio possível na elaboração deste projeto.

Ao meu irmão Felippe Vilaça, por ter tido paciência e me ajudado em tudo que foi possível.

Não tenho palavras para agradecer a pessoa mais importante na conquista dessa etapa da minha vida, minha mulher Priscila Cortez. Muito obrigado por todo o seu amor, carinho, companheirismo e apoio, nas revisões e noites mal dormidas. Obrigado por tudo, essa conquista é nossa.

Ao meu professor orientador Marco Antônio de O. Araújo, pelo direcionamento, clareza e comprometimento.

Muito obrigado ao Júlio Lessa, que me deu auxilio de suma importância no desenvolvimento deste projeto.

A minha amiga Ana Paula, que sempre me ajudou e lutou comigo no decorrer de todos os anos na faculdade.

Obrigado aos pingüins, aos praguinhas, aos “mais bebi” e a todos os meus amigos que me apoiaram nesta longa jornada. Agradeço a todos que compreenderam a minha ausência durante a elaboração deste projeto, que me deram ajuda e estimulo para continuar a batalha em busca do diploma de engenheiro.

“O sucesso só vem antes do trabalho no dicionário.”

(Albert Einstein)

RESUMO

A comunicação instantânea, também conhecida por *chat* ou “bate-papo”, trata-se de um recurso de rede para troca de informações *online* e em tempo real, sendo, normalmente, utilizada em conversas formais ou informais entre pessoas. Os *softwares* de comunicação instantânea presentes no mercado, como o *Microsoft Messenger*, o *Yahoo Messenger* ou o *Google Talk*, não são considerados um meio seguro para o tráfego de informações, pois, na maioria dos casos, não oferecem mecanismos para garantir a confidencialidade, a integridade e a autenticidade, que são pilares da segurança, nas mensagens trocadas entre as partes envolvidas na comunicação. Desta forma, é possível que outras pessoas possam ter conhecimento do teor das mensagens e até mesmo modificá-las, tendo em vista que as mesmas transitam pela rede de computadores como um texto explícito para leitura. Além disso, o receptor não tem certeza de que a mensagem recebida condiz com a mensagem enviada e nem pode afirmar qual a origem da mesma. Para amenizar estes riscos, esse projeto visa aumentar o nível de segurança para troca de mensagens, criando um sistema que utiliza técnicas como a criptografia, que é o ato de codificar dados em informações aparentemente sem sentido, para que a mensagem não seja interpretada por terceiros. Uma função matemática irá criar um sumário da mensagem, para que se possa garantir a integridade das informações trafegadas, podendo afirmar que não sofreram modificações ao longo do seu trajeto. A técnica principal do projeto será a utilização da assinatura digital, que implementará autenticidade nos dados enviados, permitindo assim, que o destinatário tenha certeza que a mensagem recebida foi originada realmente pela fonte anunciada.

Palavras-chave: Sistemas Ponto a Ponto, Segurança, Criptografia, Função *Hash*, Comunicação Instantânea.

ABSTRACT

The instant communication, also known as chat, is a network resource used for online information transfers; it is commonly used in formal and informal conversations. Some instant communication softwares existents in the market, such as *Microsoft Messenger*, *Yahoo Messenger* and *Google Talk*, are not considered a safe way for trafficking of information because most of the time they don't offer necessary tools to guarantee the confidentiality, the integrity and the authenticity that are main roles at the security of the messages changed between the parts involved at the communication. This lack of guarantee, grants other people the possibility to know the content of the messages and even to modify them, these messages run free at computer networks as explicit texts for reading. Besides that, the receiver does not know for sure if the received message matches with the sent one and cannot even affirm its origin. To reduce these risks, this Project will increase the security level between the messages changes, developing a system that uses techniques such as encryption, which is the act of encoding data in information that are apparently meanless securing that the message won't be interpreted by third. A math function will create a message summary to guarantee the integrity of the information, been able to affirm that they had not suffer any modification along their way. The main technique of this project is the use of digital signatures, that implements authenticity of the sent data, making possible for the receiver to know, for certain, that the received message was originated by the announced source.

Key words: Peer-to-peer, Security, Encryption, Hash Function, Instant Communication.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

Capítulo 1 – Introdução

1.1 MOTIVAÇÃO

A comunicação instantânea se resume a um aplicativo de intercâmbio de informações em tempo real, que tem sido cada vez mais difundido no meio social e profissional. Segundo pesquisa do Ibope/NetRatings realizada em 23 de março de 2005, o Brasil é líder no uso de comunicadores instantâneos com 6,6 milhões de usuários, seguido pela Espanha, que ocupa a segunda colocação. Segundo levantamento realizado em fevereiro de 2005, o volume de usuários no Brasil representa 60,2% dos usuários residenciais ativos, o que soma um total de 11 milhões de usuários[1]. “No mundo dos negócios os aplicativos de comunicação instantânea têm se revelado uma excelente ferramenta para comunicação, principalmente quando os membros de uma determinada empresa estão distribuídos geograficamente” [2].

Na Figura 1.1, pode-se observar o crescimento do número de usuários da internet no Brasil, que resulta em um crescimento proporcional ao número de usuários de comunicadores instantâneos [3].

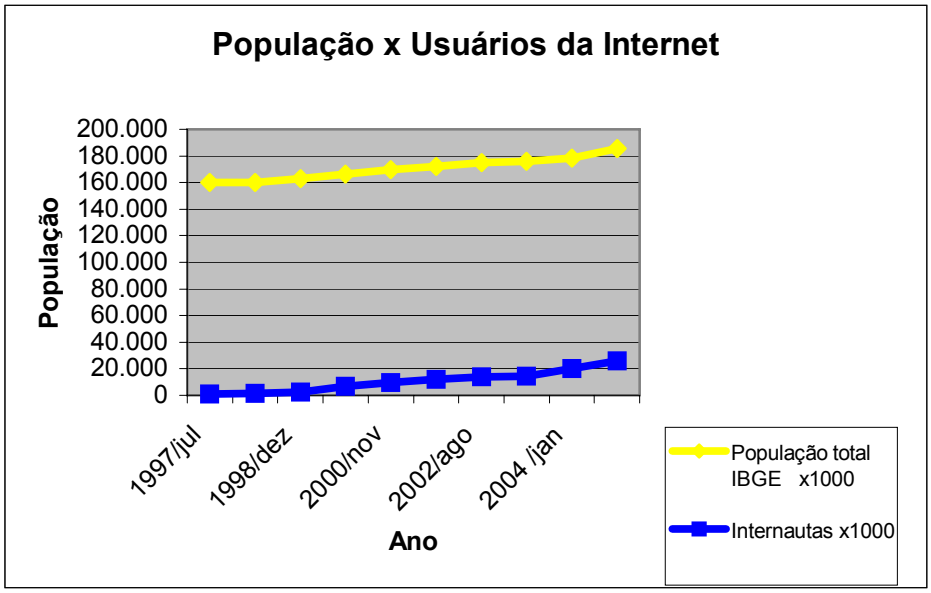

Figura 1.1 – População do Brasil x Número de usuários da internet.
(fonte: http://www.e-commerce.org.br/STATS.htm#D em 01/09/2006)

Com o crescimento visualizado na Figura 1.1 segundo fontes de pesquisa, o Brasil já ocupa a décima colocação no ranking de países com maior número de usuários da internet, onde aproximadamente 26.6% destes fazem uso de aplicativos de comunicação instantânea. A Tabela 1.1, que mostra os países detentores do maior número de usuários da internet, revela a posição de destaque do Brasil ocupando a décima colocação no ranking [3].

Tabela 1.1 – Os países com o maior número de usuários da internet
(http://www.e-commerce.org.br/STATS.htm#D em 01/09/2006)

Posição	**País**	**Usuários Internet**	**População (est. 2006)**	**Adoção da Internet**	**Mundo (%)**
1	Estados Unidos	203.824,428	299.093,237	68,1%	20%
2	China	111.000,000	1.306.724,067	8,5%	10,9%
3	Japão	86.300,000	128.389,000	67,2%	8,5%
4	Índia	50.600,000	1.112.225,812	4,5%	5%
5	Alemanha	48.722,055	82.515,988	59%	4,8%
6	Reino Unido	37.800,000	60.139,274	62,9%	3,7%
7	Coréia do Sul	33.900,000	50.633,265	67%	3,3%
8	Itália	28.870,000	59.115,261	48,8%	2,8%
9	França	26.214,174	61.004,840	43%	2,6%
10	**Brasil**	**25.900,000**	**184.284,898**	**14,1%**	**2,5%**
14	Espanha	17.142,198	44.351,186	38,7 %	1,7 %

O processo de comunicação instantânea da grande maioria dos softwares presentes no mercado pode ser descrito tomando como base um usuário que faz uso da internet e que deseja conversar com outros usuários. Cada usuário adiciona na sua lista pessoal várias pessoas com quem deseja se comunicar, e esta ação recíproca permite que cada contato apareça ativo e disponível para a conversação. Por questão de privacidade, os usuários ali adicionados podem ou não aceitar a conversa, quando a conversa é aceita o tráfego das mensagens é iniciado [4].

Independente da forma como é utilizado, este tipo de comunicação deve garantir a segurança da informação, tendo como principal objetivo dificultar o risco da revelação ou alteração do conteúdo trafegado por pessoas não autorizadas.

As intranets[1], extranets[2] e a grande rede internet são os meios mais utilizados para o uso dos aplicativos de comunicação instantânea, sendo este último o que possui um maior número de usuários destes *softwares*. No entanto, como a internet não foi concebida, originalmente, como um meio seguro de transferência de informações, o sigilo e a confidencialidade deste tipo de comunicação está prejudicada, a mercê de ataques maliciosos que podem vir a acarretar prejuízos aos participantes da comunicação [5].

O estudo "Internet Security Threat Report" é um relatório sobre ameaças da segurança na internet que foi realizado pela empresa americana Symantec, especializada em segurança de informação, no período entre 01/01/2005 a 30/06/2005, e que revela o aumento dos ataques virtuais[3] na América Latina. Neste estudo o Brasil é considerado, após os Estados Unidos, como o maior gerador de ataques maliciosos na internet (Tabela 1.2 – Estatística de ataques na América Latina). A pesquisa foi realizada com base em dados coletados através de detectores da empresa instalados em 24 mil computadores espalhados por 180 países [6].

Tabela 1.2 – Estatística de ataques na América Latina

Colocação	Total (%)	País
1	45	Estados Unidos
2	19	Brasil
3	9	México
4	4	Argentina
5	23	Outros

A Tabela 1.2 revela a posição de destaque do Brasil frente aos demais países, apresentando uma grande vantagem percentual frente ao México, o país que segue o Brasil na relação.

O estudo ainda revela que "cerca de 74% dos códigos maliciosos[4] disseminados na América Latina têm o objetivo de roubar informações sigilosas", quer seja pessoal quer seja corporativa. No relatório do ano de 2004 esse número não ultrapassava 50% (Figura 1.2 – Percentual de ataques a

1 Rede de computadores privativa que utiliza as mesmas tecnologias que são utilizadas na Internet [7].
2 Rede de computadores com tecnologia Internet que mantém comunicação com a empresa, mas está situada fora dela. Em geral, usada para conectar a empresa com seus fornecedores e clientes [8].
3 Ataque intencional realizado por indivíduos mal intencionados utilizando-se de meios computacionais para a realização do mesmo [6].
4 Conhecidos como *malwares*, são programas especificamente desenvolvidos para executar ações danosas em um computador. São exemplos de *malware:* vírus, *worms*, *spyware* [9].

internet a América Latina), percebe-se então, um aumento preocupante, resultando um maior número de pessoas mal intencionadas usufruindo das falhas de segurança na internet. As falhas nos comunicadores instantâneos são consideráveis e já aparecem em estatísticas como um dos principais alvos de *Hackers* [5] [6].

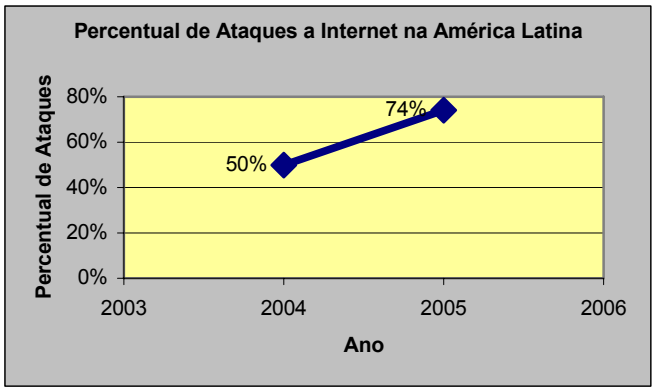

Figura 1.2 – Percentual de ataques a internet na América Latina

1.2 OBJETIVO

Devido ao desenvolvimento da comunicação instantânea, este tipo de *software* atingiu tamanha proporção de uso, que hoje se destaca entre os principais alvos de ataques virtuais. A proposta principal deste projeto é desenvolver um *software* que permita os usuários se comunicarem através de mensagens instantâneas de forma segura, garantindo a autenticidade das informações trafegadas. Desta forma, podemos diminuir o risco da interceptação e posterior leitura das informações trafegadas na comunicação, assim como o problema principal da integridade destes dados. Para garantir a solução dos problemas mencionados será utilizado um algoritmo criptográfico e uma função *Hash*, associados a um *software* de comunicação instantânea, que possuirá algumas funcionalidades dos *softwares* já existentes e mais utilizados no mercado, porém com o incremento da confidencialidade e integridade dos dados trafegados.

5 Programadores maliciosos e ciberpiratas que agem com o intuito de violar ilegal ou imoralmente sistemas de computação [10].

1.3 ESTRUTURA DA MONOGRAFIA

No capítulo 1 foi apresentada a motivação que levou ao desenvolvimento deste projeto além do seu objetivo proposto.

No capítulo 2 será apresentado o referencial teórico que dará sustentação ao desenvolvimento do projeto. Onde serão citadas as formas existentes de métodos e técnicas para a implementação deste projeto, assim como demonstradas as justificativas das metodologias utilizadas, com seus respectivos locais de pesquisa.

No capítulo 3 serão apresentadas as premissas utilizadas no projeto, o desenvolvimento do projeto envolvendo a parte do software, e como ocorre em sua estrutura o procedimento de troca de mensagens instantâneas e toda a parte de segurança integrada no programa.

No capítulo 4 serão apresentados os testes e simulações realizadas com o comunicador instantâneo, assim como as dificuldades enfrentadas. Também é demonstrado como os objetivos foram atingidos na interceptação da mensagem e no procedimento onde não ocorra esse processo. Para efeito de demonstração do projeto serão utilizados dois computadores interligados por uma rede ponto a ponto, utilizando o programa desenvolvido para a comunicação instantânea segura entre os mesmos.

Finalmente, no capítulo 5 são apresentadas as conclusões e as perspectivas de evolução do projeto.

Capítulo 2 – Referencial Teórico

Este capítulo mostrará os referenciais teóricos que foram utilizados para o desenvolvimento deste projeto.

O embasamento teórico desse projeto está dividido em quatro tópicos distintos: Sistema Ponto a Ponto, Mensagem Instantânea, Criptografia, Autenticidade e Função *Hash*. Serão abordados conceitos, objetivos, características e algumas particularidades de cada item.

Também fará uma explicação a respeito da escolha de cada opção dos itens para compor o projeto.

2.1 SISTEMA PONTO A PONTO

Este será o tipo de sistema utilizado neste projeto, e sua definição se dá pela forma na qual ele está distribuído e interligado entre os computadores que fazem o seu uso. Assim, para esclarecer o que significa esse tipo de sistema devemos definir, detalhar, e mostrar as características de uma rede local de computadores interligada ponto a ponto.

2.1.1 Tipos de rede

Quanto à aplicação podemos ter dois tipos de redes de computadores [11]: cliente-servidor e ponto a ponto.

2.1.1.1 Cliente-Servidor

Uma máquina ou um pequeno grupo de máquinas centraliza os serviços da rede oferecidos às demais estações, tais como aplicativos e filas de impressão. As máquinas que requerem esses serviços são chamadas de clientes, e as máquinas que os fornecem são chamadas de servidores.

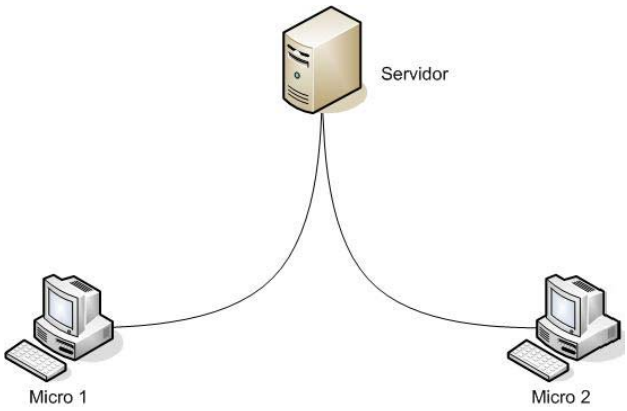

Figura 2.1.1 - Rede Cliente-Servidor

2.1.1.2 Rede ponto a ponto

Caracterizada por ser pequena e simples de ser estruturada, é uma rede independente de organização central ou servidores, além de dispor às máquinas integrantes as mesmas capacidades e responsabilidades dentro da arquitetura. Através dessa tecnologia qualquer dispositivo pode acessar diretamente os recursos de outro, sem nenhum controle centralizado.

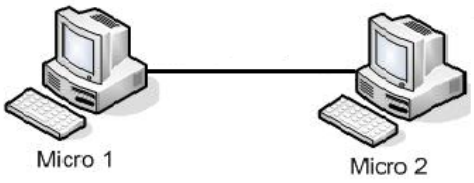

Figura 2.1.2 - Rede Ponto a Ponto

As redes ponto a ponto descentralizadas apresentam diversas vantagens sobre as redes tradicionais de cliente-servidor, pois têm infinita capacidade de expansão sem necessidade de recursos centralizados e dispendiosos. Sua estrutura funcional garante o acesso ágil e contínuo às informações, onde cada computador conectado funciona de maneira independente, permitindo a exploração, o acesso e a localização de dados de interesse do usuário [11].

Sendo o projeto baseado em um sistema de comunicação ponto a ponto e a simulação para troca de mensagens acontecendo somente entre duas

máquinas, será utilizada uma rede ponto a ponto para atender a necessidade do desenvolvimento.

2.2 MENSAGEM INSTANTÂNEA

Mensagem instantânea é uma forma popular de comunicação que oferece a capacidade de trocar mensagens entre pessoas em tempo real por uma rede de computadores.

Nos últimos anos ocorreu uma grande mudança de comportamento, e hoje as pessoas estão deixando de usar o telefone para se comunicar pela internet através de mensagem instantânea. Desde o ano de 1996 esta forma de se comunicar revolucionou a vida das pessoas, no início, surgiu um comunicador instantâneo chamado de *ICQ - I seek you* (Eu Procuro Você), da empresa Mirabilis, e em seguida surgiram diversos outros, entre eles, o *MSN Messenger* da empresa Microsoft que hoje detêm o maior número de usuários deste tipo de software [12].

A possibilidade de poder observar as pessoas entrando na rede e estabelecer contato através de uma mensagem em tempo real com as mesmas, tornou as aplicações de mensagem instantânea uma das mais populares da Internet.

Os sistemas de mensagem instantânea fornecem entrega imediata das mensagens ao destinatário, diferentemente do correio eletrônico, em que a mensagem enviada pelo emissor é armazenada na caixa postal do destinatário e entregue apenas quando este verifica se existem novas mensagens no servidor.

Em alguns sistemas de comunicação instantânea, se o usuário destino não estiver disponível para comunicação, a mensagem pode ser armazenada até que o mesmo se torne disponível, ou ela pode ser simplesmente descartada. Para evitar esta incerteza na entrega, a maioria dos sistemas fornece uma lista de contatos com um mecanismo capaz de identificar um usuário e determinar o seu estado, por exemplo, ativo, inativo ou ocupado. Na figura 2.2 estão os tipos de estados que um usuário do MSN *Messenger* pode assumir.

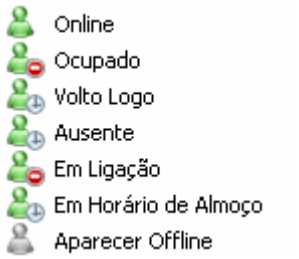

Figura 2.2 – Alguns exemplos de estados de usuários do *MSN Messenger*.

O correio eletrônico revolucionou o mundo e as empresas, mas está sendo gradativamente substituído pelas mensagens instantâneas nos locais que necessitam de praticidade e velocidade nas comunicações. O mesmo processo de substituição se aplica ao telefone, que possui, entre suas desvantagens, o alto custo.

2.2.1 Riscos Existentes

A seguir serão apresentados alguns problemas relacionados aos aplicativos públicos de comunicação instantânea presentes no mercado:

- Os aplicativos que permitem a comunicação instantânea criam canais de comunicação que o *firewall*[6] das empresas não consegue filtrar, permitindo então que terceiros localizados fora da rede corporativa tenham acesso às informações trafegadas [4].

- As informações trocadas através das mensagens podem ser armazenadas localmente na máquina de um usuário ou mesmo em um servidor e este conteúdo pode ser revelado a qualquer momento em uma investigação pessoal ou mesmo em ataques maliciosos, ferindo a privacidade dos usuários destes *softwares* [4].

- Alguém mal intencionado pode revelar a localização de origem de um usuário durante uma comunicação ou mesmo na transmissão de um arquivo, fazendo uso do endereço para realizar uma invasão ao computador ou até mesmo a rede de computadores na qual o usuário esteja conectado [4].

6 *Firewall* é um sistema de rede que tem por função regular o tráfego de rede entre redes distintas e impedir a transmissão de dados nocivos ou não autorizado de uma rede a outra [13].

- Os comunicadores instantâneos são uma porta de entrada para vírus que podem prejudicar o computador do usuário ou danificar toda a sua rede de computadores [4].

Uma pesquisa realizada em outubro de 2004 pelo grupo de segurança Sans Institute revela que os programas de mensagens instantâneas e aplicativos para a troca de mensagens estão entre os principais causadores de vulnerabilidades ao computador ou rede em que o usuário esteja conectado, resultando em uma porta de entrada para programas maliciosos e pessoas mal intencionadas [14].

Microsoft Messenger, *Yahoo Messenger*, *Google Talk*, *AOL Messenger e ICQ* são nomes de alguns dos comunicadores instantâneos públicos mais conhecidos e utilizados na internet. Já é fato que a grande maioria dos usuários dos *softwares* citados não estão cientes das vulnerabilidades destes aplicativos. Podemos ver no gráfico abaixo os comunicadores mais vulneráveis devido a sua utilização.

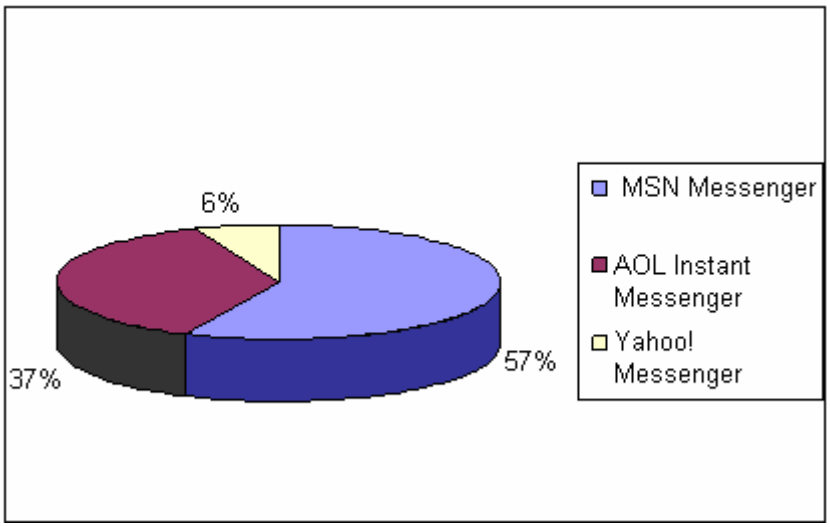

Figura 2.2.1 – Número de ataques por comunicadores instantâneos.
(Fonte: http://www.forumpcs.com.br/noticia.php?b=146515)

2.2.2 Utilização nas Empresas

O conceito de mensagens instantâneas evoluiu para atender às empresas por sua praticidade e versatilidade, este meio de comunicação já

promete ser o de maior crescimento da história em ambientes corporativos. Recentemente, companhias e profissionais de diversas áreas descobriram como usar essa ferramenta para aperfeiçoar a comunicação no mundo dos negócios.

Os sistemas de mensagens instantâneas empresariais propiciam perguntas e respostas em tempo real, eliminando a espera por e-mails que demoram, às vezes, dias para serem respondidos, telefones ocupados e a necessidade de retorno de ligações.

As mensagens instantâneas proporcionam interação imediata, complementando os sistemas tradicionais de comunicação e trazendo novas vantagens. A flexibilidade deste meio representa uma nova tecnologia que quebra barreiras de custos, ao mesmo tempo em que amplia a produtividade das pessoas e o compartilhamento de informações. Com um sistema empresarial de mensagens instantâneas, as empresas passam a contar com o que há de mais revolucionário em comunicação na atualidade. Exatamente por isso, todas as análises do setor apontam para uma crescente adoção destes sistemas nas corporações.

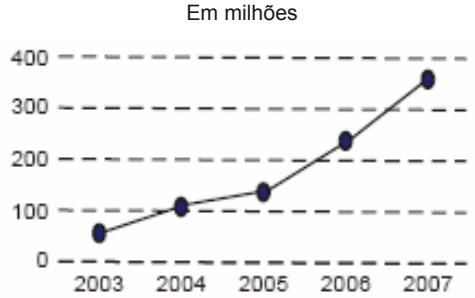

Figura 2.2.2 – Uso de mensagem instantânea em empresas de todo o mundo.
(Fonte: http://www.radicati.com/)

O Radicati Group recentemente publicou um relatório no qual foi atestado o seguinte [15]:

- 85% das empresas norte-americanas usam comunicadores instantâneos.
- 20% das empresas do mundo todo usam comunicadores instantâneos e a previsão é que este número atinja os 80% no final de 2008.

Segundo o Radicati Group, algumas motivações e resultados se mostraram constantes nas empresas:

- 44% otimizaram a comunicação entre as pessoas dentro dos escritórios.
- 33% reduziram os custos de telefonia.
- 11% melhoraram a produtividade dos funcionários.
- 11% complementaram satisfatoriamente os sistemas de comunicação já utilizados, como e-mail e telefone.

2.2.3 Benefícios dos Comunicadores Empresariais

- Amplia as oportunidades de colaboração, incluindo conferências;
- Não depende da internet para prover comunicações entre os *softwares;*
- Aumenta a segurança dos dados por trafegar em uma rede interna;
- Reduz ou elimina o uso de comunicadores instantâneos públicos, conseqüentemente, ameaças externas;
- Permite a interligação de redes, possibilitando a conexão entre pessoas e filiais.

2.3 CRIPTOGRAFIA

Com o aumento crescente de ameaças e pessoas mal intencionadas ligadas nas redes de computadores, se faz necessário assegurar a existência de ambientes confiáveis para a troca de informações de uma forma segura. Isto é possível aumentando o nível de segurança através do recurso da criptografia.

A criptografia, do Grego kryptós, "escondido", e gráphein, "escrever", existe desde a era romana e defini-se como a arte ou ciência que utiliza a matemática para escrever uma mensagem em cifra ou em código, de forma a permitir que a informação seja mantida, mas que somente o destinatário possa decifrá-la e compreender o seu conteúdo [16].

A primeira e principal garantia oferecida pela criptografia é a confidencialidade, sendo isto possível, tais dados podem ser armazenados ou transmitidos através de redes inseguras como a Internet.

Com a criptografia é possível transformar um texto original (*plain text*) ou texto claro (*cleartext*) em texto cifrado (*ciphertext*) ou codificado, que tem a aparência de um texto ilegível. Para que seja possível ler esse texto é necessário desencriptar ou decodificar a mensagem.

Figura 2.3 – Funcionamento da criptografia.
(Fonte: http://www.tdec.com.br/Parceiros/PGP/Imagens/crypto_img_1.jpg em 15/09/2006)

Tradicionalmente um sistema criptográfico integra os seguintes elementos:

- Algoritmo – Fórmula orientadora para a transformação dos dados;

- Criptograma – Texto que sofreu a transformação imposta pelo algoritmo definido;

- Chave – Parâmetro definido pelo algoritmo, responsável pela transformação do texto claro, não codificado, em criptograma (cifrar), ou pela transformação inversa, tornando o texto claro novamente (decifrar) [17].

Quando o objetivo for enviar uma mensagem de forma confidencial, o sistema tem que entregar a informação ao algoritmo de encriptação de tal modo que este a codifique numa mensagem encriptada. Na encriptação perfeita, não existe qualquer relação entre a mensagem encriptada e a original, impossibilitando assim, uma comparação entre ambas. Já no receptor, o

algoritmo de desencriptação terá que gerar a mensagem original a partir da encriptada.

Os algoritmos de criptografia são de domínio público e bem conhecidos e, portanto, se tais algoritmos fossem eficazes por si só para a encriptação e desencriptação de mensagens, qualquer atacante poderia usá-los na descodificação não autorizada de mensagens da rede, quebrando por completo as características necessárias a um sistema seguro.

Desta forma, é necessária a utilização de uma sequência de *bytes*, gerada aleatoriamente, nas operações de encriptação e desencriptação, designadas por chaves. Assim, em caso de ataque, alguém mal intencionado só decodifica eficazmente a mensagem interceptada com a chave certa.

Em um algoritmo perfeito de encriptação com chave, a mensagem encriptada não deve estar relacionada nem com o algoritmo de encriptação nem com a chave utilizada. Esta suposta ausência de relação entre a mensagem que será encriptada, a mensagem encriptada e a chave utilizada é fundamental, pois caso contrário seria possível recuperar a chave utilizada na encriptação a partir da mensagem encriptada, e proceder então à sua desencriptação.

Porém, nos algoritmos conhecidos existem essas relações, mas são de tal forma desconsideráveis, dada a atual capacidade computacional. O tempo necessário para a geração de uma chave válida a partir de uma ou mais mensagens encriptadas pode demorar centenas de anos [18].

2.3.1 Principais Tipos

Existem dois tipos principais de criptografia: a de chaves simétricas e a de chaves públicas, também conhecida como assimétrica.

Este item da monografia irá descrever um breve conceito sobre criptografia de chaves simétricas e um detalhamento mais específico sobre a criptografia de chaves públicas, pois esta será a utilizada neste projeto.

2.3.1.1 Chave simétrica

A criptografia de chave simétrica utiliza uma única chave secreta para codificação e decodificação das informações. Desta forma, tanto o emissor como o receptor devem possuir a mesma chave para o algoritmo de criptografia. Este tipo se caracteriza por usar funções matemáticas mais simples e por isso ser mais rápida, porém possui um problema, as duas partes precisam trocar as chaves por um canal seguro, pois caso alguém intercepte esta troca poderá decifrar todas as informações trafegadas.

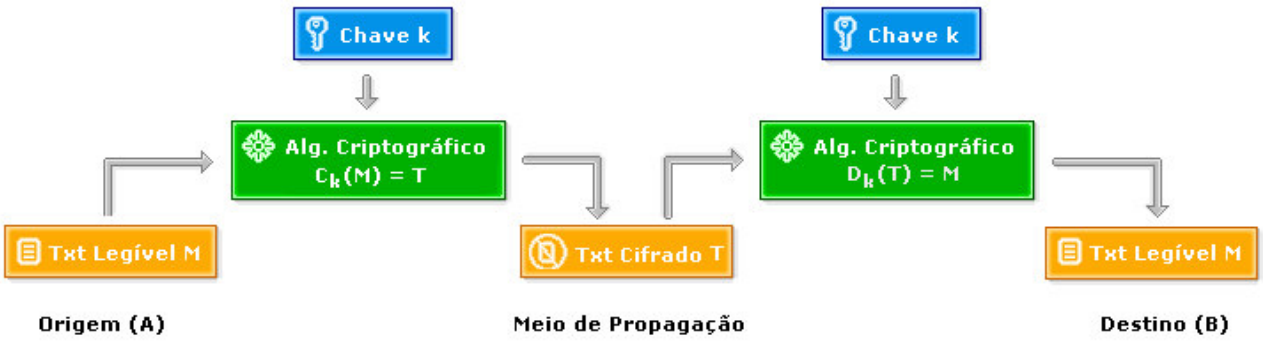

Figura 2.3.1.1 – Sistema de chave simétrica. Utiliza-se a mesma Chave K no processo.
(Fonte: http://www.inf.furb.br/~pericas/orientacoes/LogicaBAN2004.pdf em 22/09/2006)

O algoritmo simétrico de criptografia mais disseminado no mundo é o *Data Encryption Standard* (DES) que na tradução é o padrão para criptografia de dados.

2.3.1.2 Chave Assimétrica

A criptografia assimétrica surgiu para resolver a debilidade da criptografia simétrica na questão da distribuição da chave, pois nela, não é necessário que o emissor e o destinatário tenham a mesma chave única, já que existe uma chave para criptografar e uma chave diferente para descriptografar. Sua idéia foi concebida através da utilização de funções matemáticas de uma só direção, ou seja, a função sai de um estado inicial e volta a ele sem ter que passar pelo mesmo caminho.

Neste sistema de chaves assimétricas, cada pessoa deverá possuir duas chaves. A chave que criptografa a informação é a chave pública e a outra que faz a descriptografia é a chave privada. A chave pública pode ficar

disponível para qualquer pessoa que queira se comunicar com outra de modo seguro, mas a chave privada deverá ficar em poder apenas de cada titular. É somente com a chave privada que o destinatário poderá decodificar uma mensagem que foi criptografada para ele com sua respectiva chave pública. Desta forma, um intruso não conseguirá decifrar a mensagem apenas com as informações públicas, isto somente será possível com a chave que está em poder do destinatário.

Na Figura 2.3.1.2 abaixo está representada a criptografia de chave assimétrica. Onde para o usuário A enviar uma mensagem para o usuário B ele utiliza a Chave K que representa a chave pública de B. Para o destino ler a mensagem original ele utiliza a Chave S que é sua chave privada.

Figura 2.3.1.2 – Sistema de Chave Assimétrica.
(Fonte: http://www.inf.furb.br/~pericas/orientacoes/LogicaBAN2004.pdf em 22/09/2006)

A criptografia assimétrica envolve funções matemáticas de cálculos mais complexos, fazendo com que o processo de criptografar e descriptografar a mensagem seja mais demorada com relação à criptografia simétrica, porém mais segura.

Neste projeto a prioridade foi garantir um nível elevado de segurança, porém, mesmo com algoritmos de maior processamento o sistema proposto não terá sua performance na prática prejudicada, já que a troca de mensagens entre usuários não exige um tempo de resposta tão baixo.

2.3.1.2.1 Algoritmo RSA

Um algoritmo criptográfico assimétrico é uma função matemática utilizada no processo de codificação e decodificação da informação. Tais algoritmos funcionam com a combinação das chaves públicas e privadas que são utilizadas para criptografar e descriptografar, de modo que uma mesma informação criptografada com chaves diferentes produza resultados diferentes.

Neste projeto será utilizado o algoritmo de criptografia assimétrica mais famoso e implementado no mundo, o RSA. Este algoritmo foi descoberto por um grupo de pesquisadores e seu nome se deu a partir das iniciais do sobrenome dos mesmos, Ron Rivest, Adi Shamir, Len Adleman [18].

Este algoritmo é o mais bem sucedido para implementação de sistemas de chaves assimétricas, além de ser considerado um dos mais seguros por ter sobrevivido a todas as tentativas de quebrá-lo, tornando-se um algoritmo muito forte.

Com este método é possível gerar chaves de 1024 bits, que torna o sistema muito mais seguro em relação à criptografia ssimétrica, porém o processo de cifragem da informação se torna mais lento.

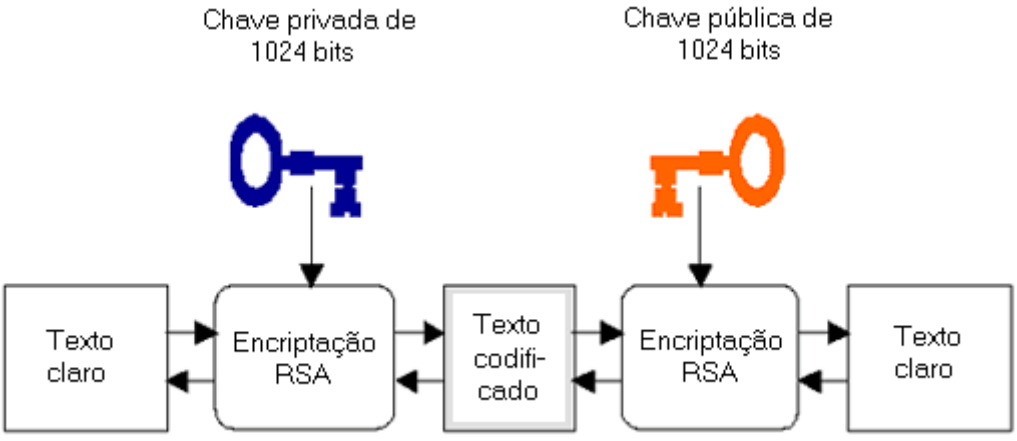

Figura 2.3.1.2.1 – Funcionamento do algoritmo RSA.
(Fonte:http://www.gta.ufrj.br/~rezende/cursos/eel879/trabalhos/vpn/conceitos_assimetrica.gif em 22/09/2006)

A segurança desse método se baseia na dificuldade de fatorar números extensos. Segundo os pesquisadores, a fatoração de um número de 200 dígitos requer 4 milhões de anos para ser processada. E para fatorar um número de 500 dígitos exige 10^{25} anos [18].

Mesmo que os computadores se tornem mais velozes, muito tempo irá passar até que se possa fatorar um número de 500 dígitos, e até lá será possível escolher a fatoração de um número ainda maior.

Para utilização do algoritmo, ambas as partes envolvidas na comunicação deverão escolher suas respectivas chaves públicas e privadas de acordo com o seguinte método [18].

1 – Escolher dois números primos distintos ***p*** e ***q***, sendo que esses números devem ser extensos e possuir em torno de uma centena de casas decimais. Quanto maior os valores, mais difícil será quebrar o RSA.

2 – Calcular:

$$n = pq$$

$$z = (p-1)(q-1)$$

3 – Selecionar um inteiro aleatório **"e"** tal que seja menor que **n** e não tenha fatores comuns com o z, exceto o 1. Assim:

$$1<e<z$$

$$m.d.c.(e,z) = 1$$

4 – Achar um número inteiro **"d"** tal que:

$$1<d<z$$

$$E*d=1(mod\ z)$$

Desta forma, a chave pública que ficará disponível será o par de números (n,e) e a chave privada que deverá ficar em posse apenas do usuário que a gerou, será o par de números (n,d).

Para o processo de criptografar a informação deve-se seguir os seguintes procedimentos:

1 – Obter a chave pública do destinatário;

2 – Dividir em blocos o texto escrito de forma clara e que será enviado para o destinatário, de modo que cada mensagem, ***M***, fique no intervalo *0*≤ M<n.

3 – Calcular:

$$\mathbf{c = M^e \bmod n}$$

4 – Enviar o texto cifrado **"c"** para o destinatário.

Para que o destinatário possa descriptografar a informação recebida deverá realizar a seguinte operação:

- Utilizar a sua chave privada **"d"** e calcular:

$$\mathbf{M = c^d \bmod n.}$$

Para visualização do processo utilizado pelo algoritmo RSA será demonstrado abaixo um exemplo de como funciona a criptografia da palavra SUZANNE [18].

Para este exemplo serão utilizados os números ***p*=3** e ***q*=11**. De acordo com o conceito de segurança desenvolvido para o RSA, estes números são muito pequenos e trazem um grande risco pelo fato de ser mais fácil para alguém mal intencionado conseguir quebrar a criptografia. Por isso, esses números devem ser adotados apenas para facilitar o exemplo proposto.

Calcula-se:

$$\boldsymbol{n = p.q = 3.11 = 33}$$

$$\mathbf{z(33) = (}\boldsymbol{p}\mathbf{ - 1)(}\boldsymbol{q}\mathbf{ - 1) = (3 - 1)(11 - 1) = 20}$$

O valor escolhido como número *d* que seja co-primo a **z(n)** deve satisfazer a equação MDC(20, d) = 1. Deste modo, ***d*=7**, pois 7 e 20 não têm fatores comuns.

A equação ***e*d* = 1 (mod z(n))*** deve ser verdadeira e para isso o **"*e*"** deverá ser um número que multiplicado por **7 (mod 20)** seja igual a 1.

Usando o método da tentativa e erro:

O valor 1 não daria certo, pois:

1.7 (mod 20) ≠ 1

O valor 2 também não daria certo, pois:

2.7 (mod 20) ≠ 1

O número 3 se identifica, pois:

3.7 (mod 20) = 1

Assim, o algoritmo já possui as duas chaves que serão utilizadas neste processo:

- A chave privada consiste em (d, n) = (7, 33)

- A chave pública consiste em (e, n) = (3, 33)

Representando cada letra do alfabeto por um número, obtém-se a tabela 2.3.1.2.1 que representa o processo de encriptar a informação.

Tabela 2.3.1 – Encriptando a informação.

Simbólico	Numérico	M^3	$C=M^3 \pmod{33}$
S	19	6859	28
U	21	9261	21
Z	26	17576	20
A	01	1	1
N	14	2744	5
N	14	2744	5
E	05	125	26

A Tabela 2.3.1.2.2 demonstra como funciona o processo inverso, de descriptografar a informação.

Tabela 2.3.2 – Descriptografia da informação.

$C=M^3 \pmod{33}$	C^7	$M=C^7 \pmod{33}$	Simbólico
28	13492928512	19	S
21	1801088541	21	U
20	1280000000	26	Z
1	1	01	A
5	78125	14	N
5	78125	14	N
26	8031810176	05	E

Como *p* e *q* escolhidos para o exemplo anterior foram pequenos, os números colocados nas tabelas não tiveram uma extensão de grande tamanho. Porém, se a grandeza escolhida fosse da ordem de 2^{512} o valor de *n* seria da ordem de 2^{1024} e desta forma as chaves poderiam ter até 1024 bits ou 128 caracteres de 8 bits, trazendo a segurança máxima do algoritmo RSA.

2.4 AUTENTICIDADE

O conceito de autenticidade é a certeza absoluta de que uma informação provém das fontes anunciadas e que não foi alvo de mutações ao longo de um processo [19]. Com outras palavras, uma informação é autêntica quando é possível garantir a sua origem, saber quem é o seu emissor e ter a certeza que ela não foi modificada ao longo do seu trajeto.

No projeto, será implementada a assinatura digital para identificar o emissor da mensagem e a função *Hash* para garantir a integridade da informação trafegada pela rede.

2.4.1 Assinatura Digital

A assinatura digital consiste no processo de identificação do emissor da mensagem enviada. Ela funciona para o meio digital assim como assinaturas comuns funcionam para documentos impressos. A assinatura é utilizada para garantir que um usuário escreveu ou está de acordo com a mensagem na qual a assinatura foi implementada.

Para que uma assinatura digital seja confiável ela precisa atender aos seguintes requisitos [20]:

1 – Ser verificável – Deve ser possível provar que uma mensagem enviada por um usuário foi realmente assinada por ele. O destinatário precisa confirmar a autoria da mensagem.

2 – Incontestável – Possibilitar a propriedade do não-repúdio, isto é, garantir que o emissor não negue a autoria do envio da mensagem ao destinatário.

3 – Não Falsificável – O destinatário não pode ter a possibilidade de modificar a mensagem recebida assinada digitalmente pelo emissor.

Na assinatura digital, a ligação entre a mensagem e o emissor é realizada através de um algoritmo de criptografia assimétrica e suas chaves.

Para assinar digitalmente uma informação, o emissor deve aplicar a sua chave privada na mensagem que deseja enviar e transmitir o resultado para o destinatário. Este receberá a mensagem assinada e irá verificar a assinatura aplicando a chave pública do emissor. Se o processo ocorrer sem falhas, pode-se afirmar que a mensagem recebida foi originada pelo emissor, já que a única forma de quebrar a segurança deste processo seria roubando a chave privada que fica em posse do usuário que irá enviar a mensagem.

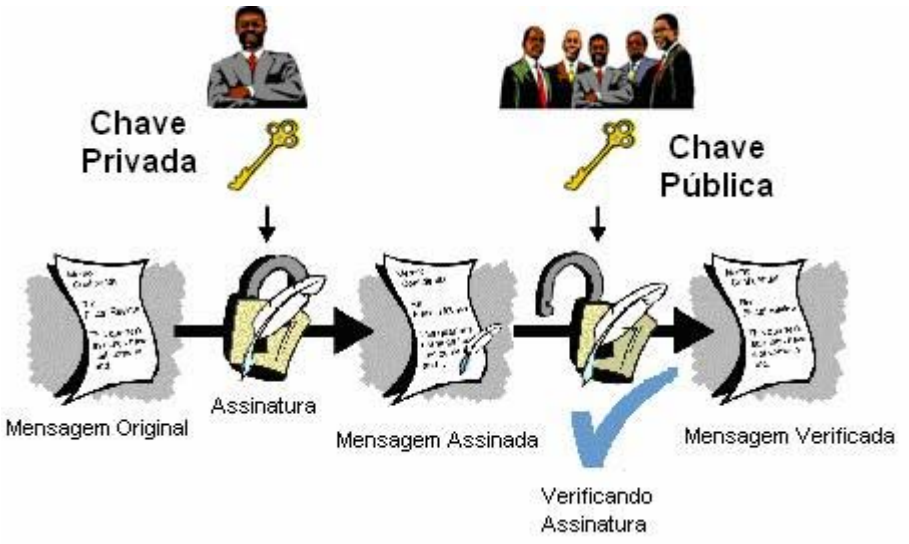

Figura 2.4.1 – Processo de Assinatura Digital e Verificação de Assinatura.
(Fonte: http://library.thinkquest.org/C0126342/a.gif em 28/09/2006)

Seria possível utilizar o algoritmo de criptografia assimétrica RSA para a assinatura digital, porém, o sistema proposto no projeto já utiliza este algoritmo para garantir a confidencialidade das informações. Além disso, existe um algoritmo, que será descrito no tópico abaixo, que foi desenvolvido somente para implementação da assinatura digital e é mais rápido que o algoritmo RSA para esta função.

2.4.1.1 Algoritmo DSA

O DSA é um algoritmo de assinatura digital (*Digital Signature Algorithm*) de chave pública, que só é utilizado para assinaturas digitais.

Este algoritmo criado pela NSA - *National Security Agency* (Agência de Segurança Nacional) nos Estados Unidos, foi patenteado pelo governo e aceito como um padrão de assinatura digital federal pelo NIST - *National Institute of Standards and Technology* (Instituto Nacional de Padrões e Tecnologia).

Sua segurança é obtida através da matemática utilizando cálculos muito diferentes do RSA, porém o algoritmo DSA é tão forte quanto o RSA. Esta segurança é semelhantemente garantida de acordo com o tamanho das chaves utilizadas no processo de encriptação e decriptação.

As chaves utilizadas pelo algoritmo DSA possuem tamanhos de 512 a 1024 bits, sendo esse máximo valor mais que suficiente para necessidades de segurança, se igualando ao RSA.

O DSA não garante a confidencialidade dos dados, pois, como foi dito anteriormente, o seu propósito é assinar digitalmente as mensagens enviadas. Já o algoritmo RSA pode ser utilizado para as duas funções, mas a sua desvantagem é ser mais lento para assinar uma mensagem, sendo que o DSA faz a função de assinar e verificar a assinatura com a mesma utilização de tempo, garantindo uma melhor performance.

O usuário ao utilizar este algoritmo precisa gerar sua chave privada para assinar digitalmente a mensagem e sua chave pública que será enviada para o destinatário da mensagem, somente esta chave poderá verificar se a assinatura do dado é confiável.

Abaixo, está sendo explicado como é o funcionamento do algoritmo DSA [21].

Para gerar as chaves é necessário obter as seguintes variáveis:

p = um número primo, tal que $2^{L-1} < p < 2^{L}$ para $512 \leq L \leq 1024$ e L múltiplo de 64.

q = primo divisor de p - 1, tal que $2^{159} < q < 2^{160}$

$g = h^{(p-1)/q} \bmod p$, tal que h é um inteiro $1 < h < p - 1$ e $h^{(p-1)/q} \bmod p > 1$

A chave privada é dada por x = um número inteiro randômico ou pseudo-randômico, tal que $0 < x < q$

Chave pública é dada por $y = g^{x} \bmod p$

k = um número inteiro randômico ou pseudo-randômico, tal que $0 < k < q$

O parâmetro k deve ser gerado a cada assinatura e ser secreto.

Para gerar a assinatura é necessário calcular o par r e s:

$$r = (g^{k} \bmod p) \bmod q$$

$$s = (k^{-1}(\text{SHA-1}(M) + xr)) \bmod q$$

Para verificar a assinatura é preciso calcular:

$$u = (s^{-1} * \text{SHA-1}(M)) \bmod q$$

$$v = (s^{-1} * r) \bmod q$$

$$r = (g^{u} * y^{v} \bmod p) \bmod q$$

O SHA-1 é uma função *hash* que tem por objetivo calcular um resumo da mensagem enviada para verificação da integridade. O funcionamento da função *hash* será explicado no próximo item.

No desenvolvimento deste projeto será utilizado o valor de SHA-1(M) como nulo, a fim de implementar separadamente um algoritmo para função *hash* e poder demonstrar o seu funcionamento nos testes e simulações do projeto.

2.5 FUNÇÃO HASH

A integridade das mensagens trafegadas pode ser garantida através de uma técnica de resumo digital, chamada de função *Hash* ou *message digest* (sumário da mensagem).

Uma função *Hash* pode ser comparada a uma impressão digital. Da mesma maneira que uma impressão digital identifica exclusivamente uma pessoa, uma função *Hash* calcula um valor de identificação para a mensagem específica.

Esta função baseada em cálculos matemáticos é responsável pela conversão de mensagens de tamanho variado em uma *string* de tamanho fixo, denominado *valor de Hash* [20].

Quando se aplica uma função *Hash* em uma mensagem M (*Hash*(M)) é necessário que o algoritmo utilizado atenda a algumas características básicas como:

- Dado um valor de entrada, ele consiga gerar um valor diferente do inicial na saída.
- Não deve haver qualquer relação entre o valor de entrada e o da saída.
- Aceitem receber valores de entrada de tamanhos diferentes e retornem valores de saída de tamanho fixo.

Para que uma função *Hash* possa ser dita segura, ela precisa atender aos seguintes requisitos:

- Deve ser computacionalmente impossível gerar o valor da entrada através do valor da saída.
- Impraticável existir duas mensagens diferentes que gerem o mesmo resumo de mensagem na saída.
- Uma mudança na entrada de até mesmo um bit deve produzir um valor de saída completamente diferente.

O grau de segurança da função é, portanto, dependente do tamanho da saída, sendo esta mais segura à medida que sejam utilizados mais bits na representação dos valores *Hash*.

Os algoritmos que implementam a função *Hash* têm como objetivo fazer com que o resumo sofra uma grande modificação se algum caractere do conteúdo da mensagem for alterado. Isso impede que um usuário mal intencionado possa modificar o conteúdo da mensagem sem que o destinatário saiba.

Para verificar se houve alteração na mensagem basta calcular o *Hash* da mensagem recebida e compará-lo com o da mensagem original, se forem iguais significa que a mensagem está íntegra e não sofreu modificações ao longo do seu trajeto.

Utiliza-se a função da seguinte forma para cálculo de integridade:

1 – O emissor calcula o *Hash* da mensagem M que ele irá enviar:

$$D = Hash(M)$$

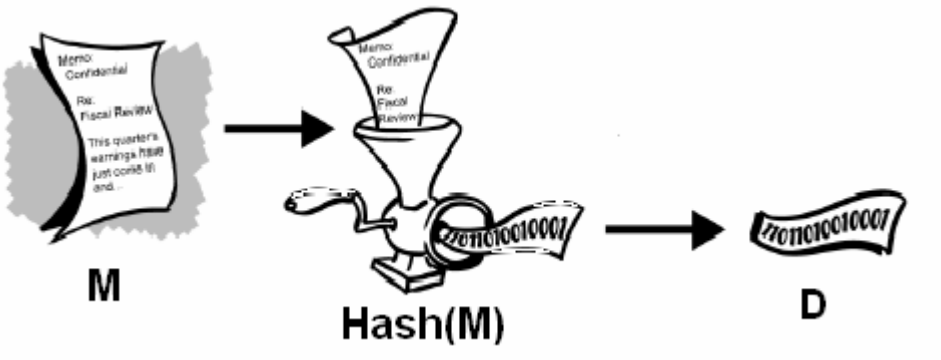

Figura 2.5 – Cálculo do *Hash* pelo emissor.
(Fonte: Autor)

2 – A mensagem M e o resumo gerado pelo Hash D são enviados ao destinatário.

Os passos que o destinatário tem que seguir para verificar a integridade da informação recebida são:

1 – Receber a mensagem M e o seu resumo gerado pela função Hash D.

2 – Calcular o hash da mensagem M, onde:

D’ = Hash(M)

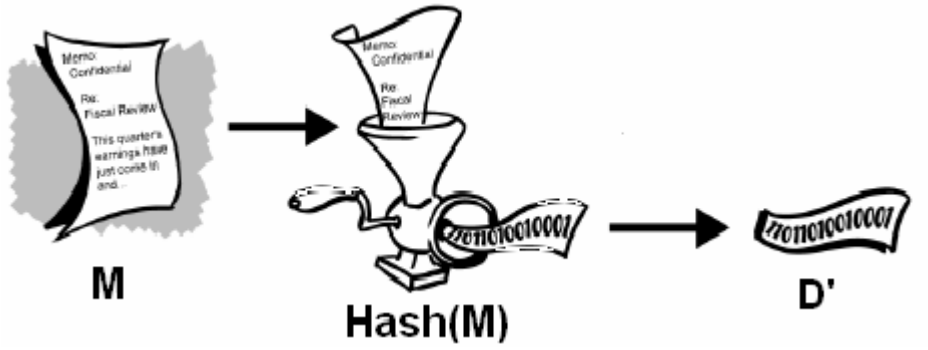

Figura 2.5.1 – Cálculo do *Hash* pelo destinatário.

(Fonte: Autor)

3 - Comparar D com D’ para ver se são iguais:

Se D = D’ → a mensagem está íntegra

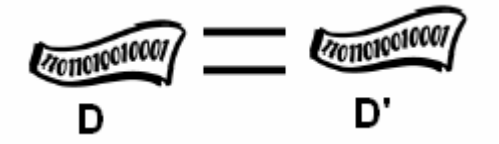

Figura 2.5.2 – Verificação do *Hash* e mensagem íntegra.

Se D ≠ D’ → a mensagem não é confiável

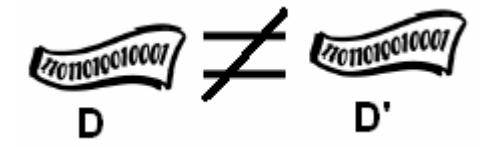

Figura 2.5.3 – Verificação do *Hash* e mensagem não confiável.

No item seguinte, serão abordados o conceito e funcionalidade do algoritmo MD5 para cálculo da função *Hash,* este algoritmo foi escolhido para desenvolver o processo de resumo das mensagens trafegadas entre os usuários, através do sistema proposto no projeto.

2.5.1 Algoritmo MD5

Desenvolvido por Ronald Rivest do MIT - *Massachusetts Institute of Technology* (Instituto Tecnológico de Massachusetts) nos Estados Unidos, o algoritmo MD5 – M*essage Digest 5* (sumário da mensagem versão 5) é um dos mais utilizados no mundo.

Por ser um algoritmo unidirecional, um resumo da mensagem gerado pelo MD5 não pode ser transformado novamente no texto que lhe deu origem.

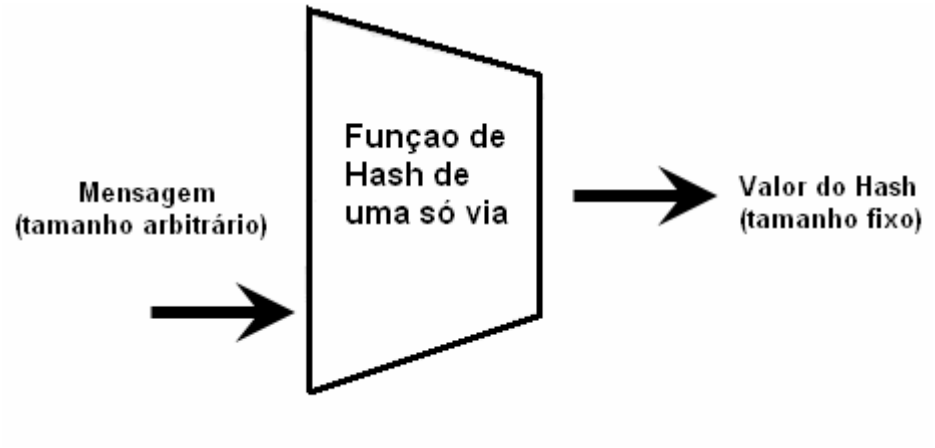

Figura 2.5.1.1 – Função Unidirecional.
(Fonte: http://www.cert-rs.tche.br/docs_html/mail3.gif em 07/10/2006)

O algoritmo MD5 produz um resumo fixo de 128 bits e a Figura 2.5.1.2 abaixo mostra como cada mensagem distinta gera um resultado diferente.

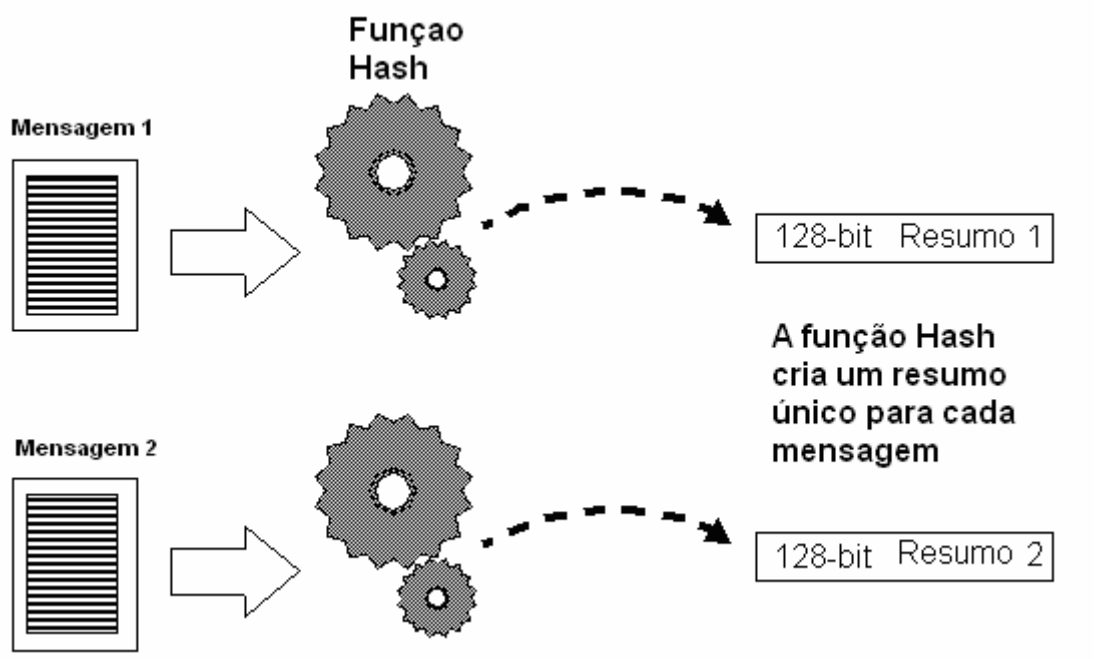

Figura 2.5.1.2 – Para cara entrada diferente a função MD5 gera uma saída distinta com valor fixo de 128bits.
(Fonte - http://www.akadia.com/img/md5.gif em 07/10/2006)

Os exemplos a seguir mostram na prática o processo da Figura 2.5.1.2 utilizando a função MD5:

Entrada: echo "There is CHF1500 in the blue box."
Saída: 2db1ff7a70245309e9f2165c6c34999d

Entrada: echo "The meeting last week was swell."
Saída: 050f3905211cddf36107ffc361c23e3d

Entrada: echo "There is CHF1100 in the blue box."
Saída: 462a8dfbfead80335053f2fa2988d276

Pode-se notar que os códigos gerados pela função MD5 são extremamente diferentes para cada mensagem. Isto acontece mesmo com a primeira e a terceira mensagem acima, que possuem diferença em apenas um caractere.

Calcular o *Hash* de cada mensagem enviada através do sistema proposto no projeto é muito importante para garantir pequenas mudanças nos

dados originados pelo emissor. Bastando para garantir a integridade da mensagem o cálculo da função MD5 na saída e no destino para efeito de comparação e garantia de que os dados originais não foram modificados acidentalmente ou propositadamente ao longo do seu trajeto.

Existe outro algoritmo chamado SHA-1 - *Secure Hash Algorithm* (Algoritmo de Hash Seguro) que está sendo muito utilizado em diversos seguimentos e substituindo o MD5, ele é mais seguro por gerar uma saída de 160bits.

Apesar desta característica importante, o MD5 foi escolhido para verificar a integridade das mensagens trafegadas pelo sistema. Ele é um algoritmo mais simples de ser implementado e possui um nível de segurança suficiente para o desenvolvimento do projeto, além de ser eficiente na geração de resumos únicos e rápidos para troca de mensagens.

Capítulo 3 – Descrição do Projeto

Este capítulo trata das especificações e toda a parte de desenvolvimento do projeto proposto.

3.1 TOPOLOGIA

A topologia da rede ponto a ponto a ser utilizada no projeto deverá seguir um dos padrões descritos a seguir:

a) A primeira opção, Figura 3.1.1, estaria ligando diretamente duas máquinas através de um cabo par trançado[7] (Figura 3.1.2), uma conexão sem fio ou qualquer outro meio de transmissão entre máquinas. Esta será a opção utilizada para demonstrar o funcionamento das técnicas de segurança aplicadas no projeto.

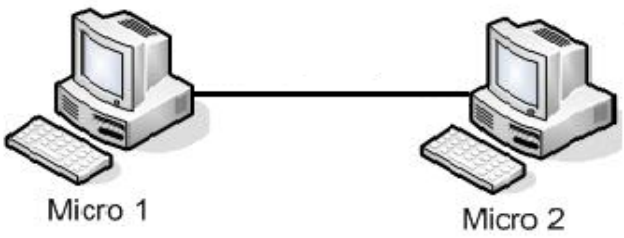

Figura 3.1.1 – Ligação direta entre duas máquinas.

Figura 3.1.2 – Cabo par trançado.
(fonte: http://www.deltatronic.com.br/img_atualizavel/foto/442f.jpg em 10/10/2006)

[7] É o tipo de cabo mais utilizado para ligar computadores em rede. Formado por pares de fios que se entrelaçam por toda a extensão do cabo, evitando assim interferências externas, ou do sinal de um dos fios para o outro [22].

b) Na segunda opção, mostrada na Figura 3.1.3, o sistema funcionará perfeitamente na comunicação de duas ou mais máquinas, interligadas entre si em uma mesma rede local, através de um dispositivo de rede, como um *Hub*[8] (figura 3.1.4).

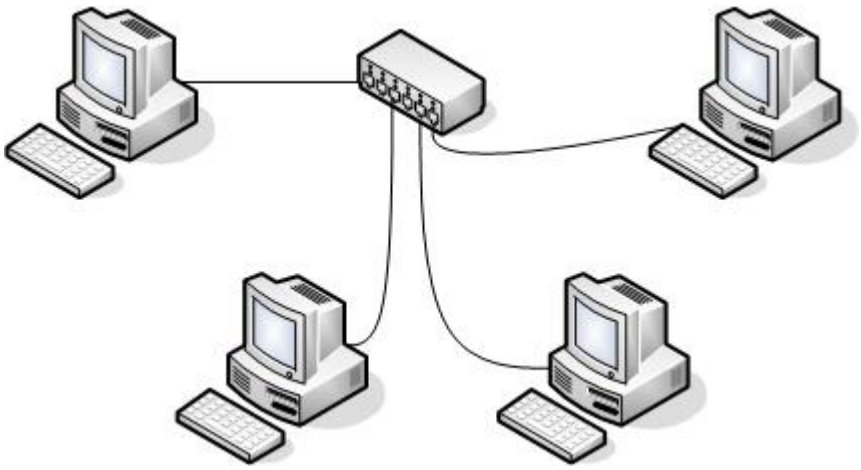

Figura 3.1.3 – Ligação entre diversas máquinas através de um dispositivo de rede.

Figura 3.1.4 – *Hub.*
(fonte: http://www2.elecom.co.jp/network/hub/100mbps/ld-csw05n/image/main.jpg em 10/10/2006)

3.2 ESPECIFICAÇÕES

Para a demonstração do propósito do projeto será utilizado um programa especificamente desenvolvido na linguagem de programação Java, que possui

[8] Trata-se de um dispositivo com diversas portas, onde cada uma delas pode ser conectada a uma máquina ou outro dispositivo. O seu principal objetivo é repassar os dados vindos de um computador de origem para o computador destino [11].

uma interface simples, de fácil entendimento e utilização, cujo nome foi denominado de MIVIL – Mensagem Instantânea Vilaça.

Como o MIVIL é em Java, foi verificado que para o MIVIL funcionar perfeitamente, é necessário que na máquina onde ele será executado possua instalado e/ou atualizado o software Java para Windows, chamado Java Runtime Environment Version 5.0, que pode ser obtido gratuitamente na internet no endereço www.java.com.

A linguagem de programação escolhida para desenvolvimento do sistema de comunicação instantânea MIVIL foi o Java, por ser uma linguagem mais nova e versátil, bastante utilizada e de muita integração com o sistema operacional Windows.

O processo de comunicação remota que acontecerá entre os comunicadores instantâneos MIVIL foi desenvolvido utilizando o RMI - *Remote Method Invocation* (Invocação de Métodos Remotos). Este recurso do ambiente de desenvolvimento Java permite a comunicação de funções entre sistemas Java que estejam em máquinas diferentes e interligadas em uma rede de computadores.

O MIVIL foi desenvolvido e testado somente em sistemas operacionais rodando na plataforma Windows. Com os recursos da linguagem utilizada no software, teoricamente, o MIVIL pode vir a ser utilizado em outras plataformas, bastando para isso, instalar a máquina virtual Java correspondente a cada tipo de sistema operacional.

3.3 INTERFACE EMISSOR / RECEPTOR

A compilação do código do MIVIL gerou um arquivo chamado MIVIL.jar. Este deve ser executado para dar início à utilização do comunicador instantâneo.

A interface inicial do MIVIL é igual para todos os usuários que desejam se comunicar, bastando apenas, personalizar a interface com seu nome e cadastrar sua lista de contatos. A tela inicial pode ser vista na Figura 3.3.2

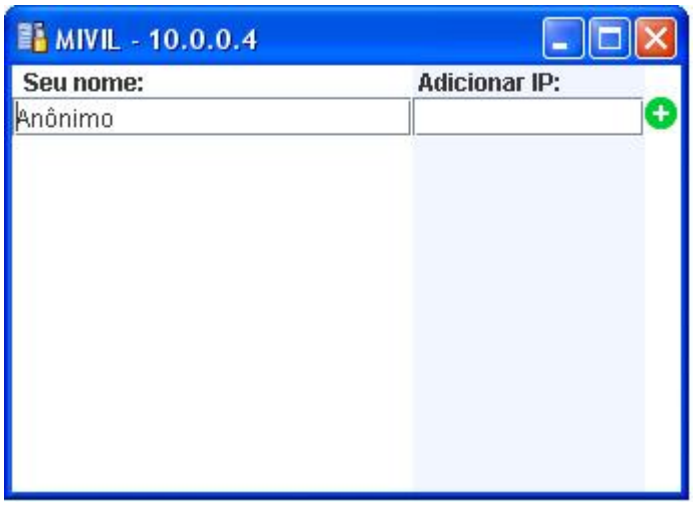

Figura 3.3.2 – MIVIL – Tela inicial do MIVIL.

Na parte de cima da tela inicial do sistema (Figura 3.3.3) é possível verificar o nome do comunicador instantâneo MIVIL e o endereço de rede do usuário que está utilizando o software.

Esta informação é de grande importância e facilita a comunicação entre as partes, pois, somente através do endereço de rede o usuário poderá cadastrar uma ou várias pessoas na qual ele deseja se comunicar, populando assim, a sua lista de contatos.

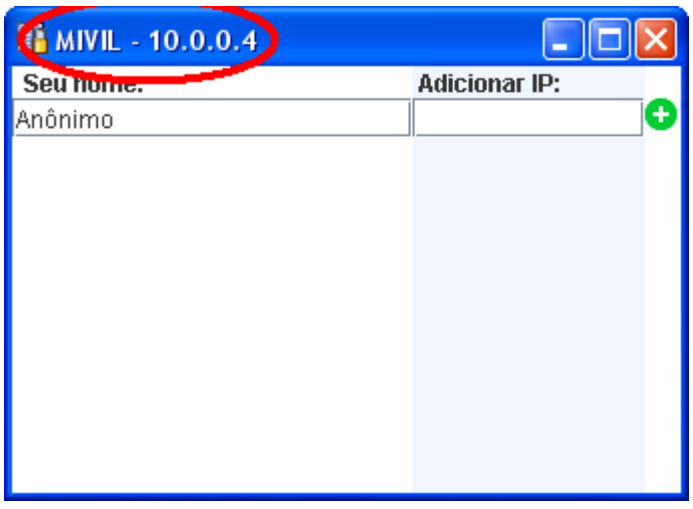

Figura 3.3.3 – MIVIL – Nome do sistema com o endereço de rede.

Caso o usuário não esteja conectado em uma rede de computadores, a informação do endereço de rede ficará conforme Figura 3.3.4. Este número

127.0.0.1 é padrão para endereços locais da máquina que não esteja interligado em uma rede [11].

Se o usuário verificar que o MIVIL detectou este endereço de rede, ele deve verificar a sua comunicação para que o sistema possa funcionar, pois dessa forma ele não conseguirá se comunicar com outras pessoas.

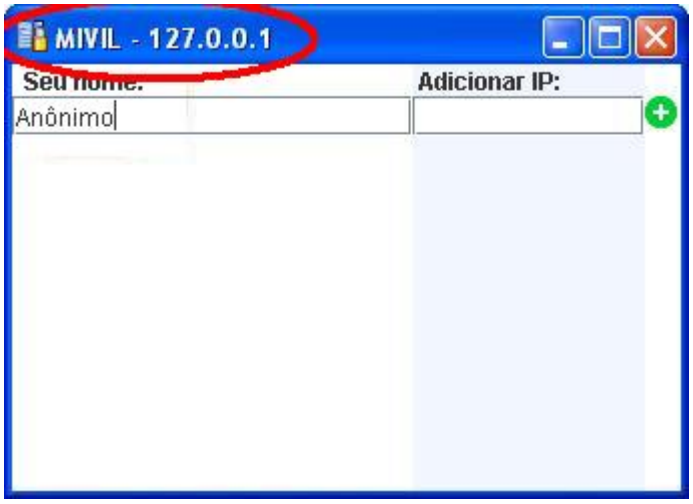

Figura 3.3.4 – MIVIL – Usuário não ligado a uma rede de computadores.

Na primeira utilização do comunicador instantâneo o usuário se depara com um campo chamado de “Seu nome” com a palavra “Anônimo” preenchendo o espaço (Figura 3.3.5).

Este campo é utilizado pelo sistema para mostrar aos usuários da lista de contatos o nome da pessoa com quem eles estão se comunicando. Sendo assim, os usuários devem colocar qualquer informação sabendo que será vista pelas pessoas com as quais ele irá se comunicar.

Após a primeira utilização, a informação colocada no campo “Seu Nome” fica armazenada em um arquivo chamado Contatos.txt, que será gravado no mesmo diretório que foi executado o MIVIL. Assim, o usuário não necessita modificar este campo toda vez que for utilizar o comunicador instantâneo.

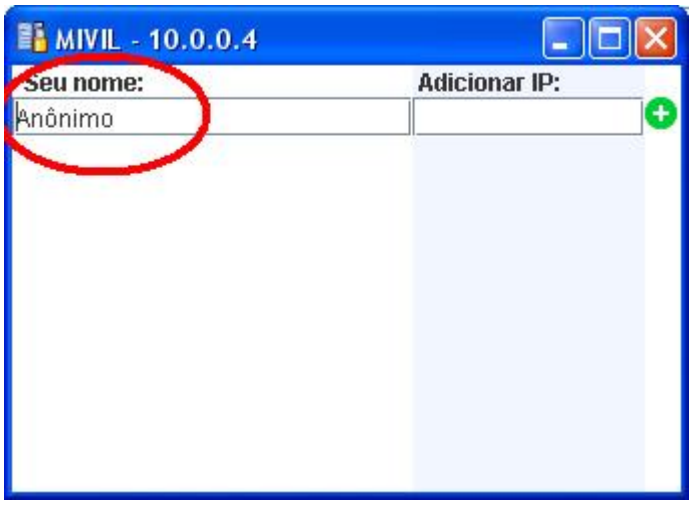

Figura 3.3.5 – MIVIL – Campo “Seu nome”.

Na parte superior direita do programa, está localizado o campo “Adicionar IP” (Figura 3.3.6). Este campo é utilizado para que o usuário adicione seus contatos através do endereço de rede dos mesmos.

Após preencher a informação com o endereço, o usuário deverá clicar no para adicionar o contato e popular a sua lista conforme Figura 3.3.7. Para remover um usuário da lista de contatos, basta clicar no e confirmar a operação, conforme Figura 3.3.8.

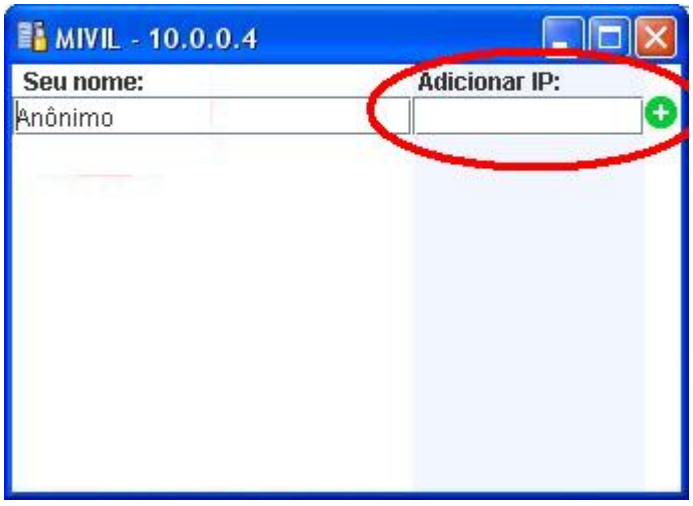

Figura 3.3.6 – MIVIL – Campo “Adicionar IP”.

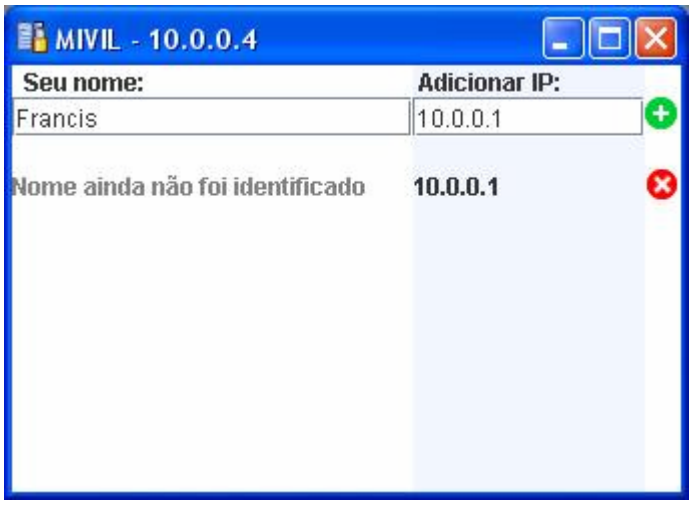

Figura 3.3.7 – MIVIL – Adicionando contatos no MIVIL.

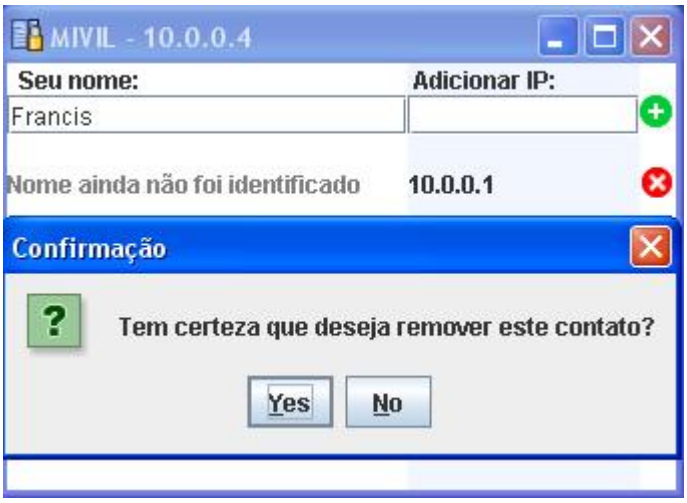

Figura 3.3.8 – MIVIL – Excluindo contatos no MIVIL.

No caso da Figura 3.3.7 o usuário Francis adicionou alguém com quem deseja se comunicar e possui o endereço de rede 10.0.0.1. Inicialmente o MIVIL adiciona os novos contatos na lista com o nome padrão “Nome ainda não foi identificado”. A cada dois segundos, o sistema busca informações deste endereço cadastrado e retorna o nome do usuário que esta utilizando o MIVIL na outra ponta (Figura 3.3.9). Se o usuário destino não estiver com o comunicador instantâneo aberto no momento da primeira busca, ficará registrado como na Figura 3.3.7.

Todos os contatos cadastrados na lista do usuário ficam armazenados no arquivo “Contatos.txt” com o nome identificado na ultima comunicação.

Quando o MIVIL é inicializado aparecem todos os contatos associados aos seus respectivos endereços de redes.

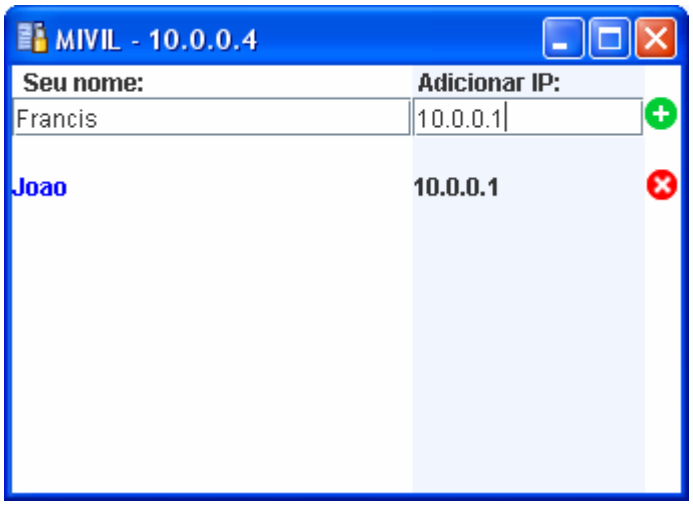

Figura 3.3.9 – MIVIL – Usuário identificado pelo endereço de rede.

Para iniciar uma comunicação instantânea com algum usuário que esteja na lista de contatos, basta clicar em cima do nome que automaticamente abrirá uma janela para ambos, onde se estabelecerá uma comunicação segura somente entre as duas partes interessadas, conforme a Figura 3.3.10.

Na Figura 3.3.10 pode-se observar que na caixa de texto inferior é onde deverá ser digitada a mensagem para o outro usuário, e para efetivar o procedimento, basta clicar em “Enviar”. O histórico do diálogo entre os usuários ficará visível na caixa de texto superior, onde o texto na cor preta é o enviado pelo usuário atual do MIVIL, e o texto em azul enviado pelo usuário que está mantendo contato, juntamente com o horário do envio da mensagem, como pode ser visto na Figura 3.3.11.

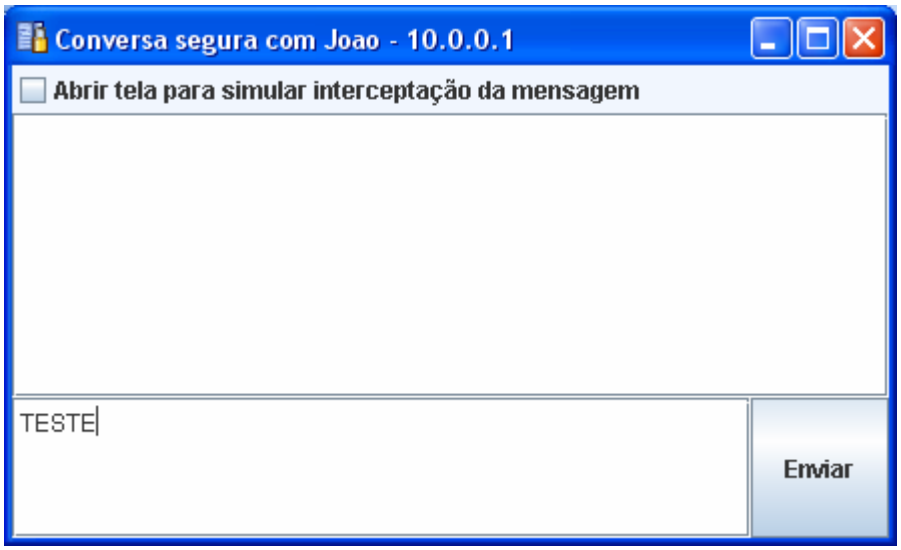

Figura 3.3.10 – MIVIL – Enviando mensagem.

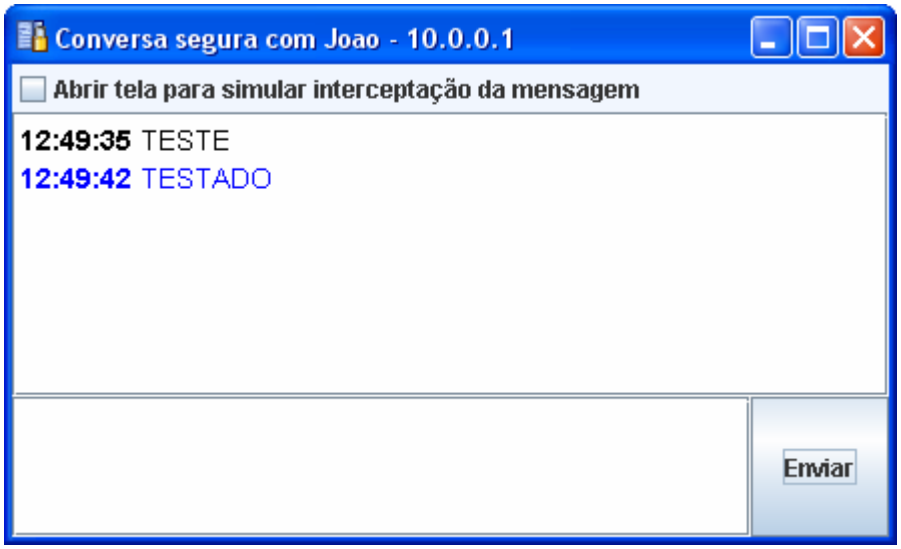

Figura 3.3.11 – MIVIL – *Chat*.

A opção “Abrir tela para simular interceptação da mensagem”, que pode ser vista na Figura 3.3.11, será explicada detalhadamente na seção 4.1 desta monografia.

Para finalizar o MIVIL o usuário deverá clicar no  que fica no canto superior direito do programa. Logo após, aparecerá uma janela confirmando a saída do sistema (Figura 3.3.12).

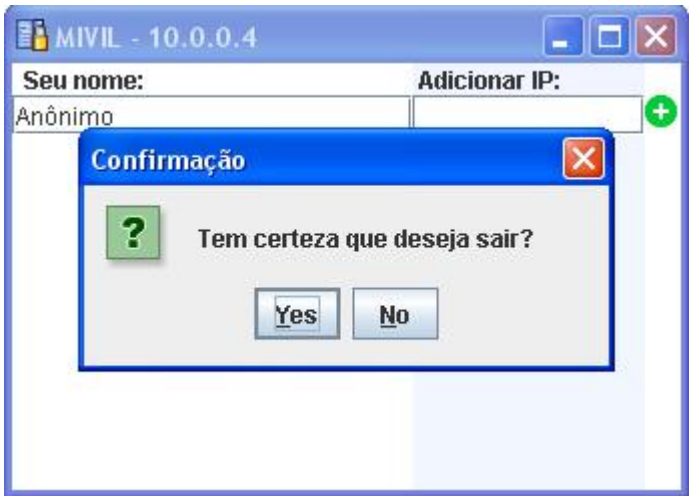

Figura 3.3.12 – MIVIL – Confirmação de saída.

A interface do MIVIL foi desenvolvida seguindo os padrões dos comunicadores instantâneos existentes no mercado. Desta forma, a sua utilização é bastante trivial e de fácil entendimento para o usuário.

3.4 PROCESSO DE COMUNICAÇÃO

A escolha do RMI para o desenvolvimento do projeto aconteceu pelo fato de ser muito mais simples do que outros métodos de comunicação e troca de informações remotas, além de atender perfeitamente e ser específico para sistemas desenvolvidos na linguagem Java.

A linguagem definida para que as diversas máquinas utilizem o MIVIL e possam conversar entre si, conhecida como protocolo de comunicação, foi o TCP/IP que possui dois protocolos importantes: TCP - Transport Control Protocol (Protocolo de Controle do Transporte) e IP - Internet Protocol (Protocolo Internet) [11].

Para que as máquinas possam se comunicar remotamente é necessário além de definir uma topologia e seu cabeamento, escolher um endereço de rede ou endereço IP diferentes para cada uma delas.

A topologia escolhida para demonstração foi o primeiro modelo de rede ponto a ponto, na qual possuem apenas duas máquinas ligadas diretamente. Os endereços de rede escolhidos fazem parte da seqüência 10.0.0.1 a

10.0.0.255 e utilizam a máscara de rede 255.255.255.0, definindo-se como uma rede classe C.

Para o meio de transporte será utilizado o *fast* ethernet (ethernet rápida), capaz de transmitir dados a 100 Mbps, configurado para estabelecer uma conexão *Full-duplex*, onde cada estação transmite e/ou recebe dados em transmissões simultâneas (figura 3.4.1).

Figura 3.4.1 – Transmissão *Full-duplex*.

A porta utilizada para comunicação nas máquinas foi a 1099, que é padrão do RMI. Ao iniciar o MIVIL esta porta é aberta para que os dados sejam transferidos entre os sistemas. Este procedimento acontece de acordo com as linhas do código do software abaixo que estão destacadas em negrito.

```
public static void iniciarServicoRMI(){
        TelaInicial ti = Sessao.getInst().getTelaInicial();
        try {lerContatos();} catch (IOException e) {}
        try {
          java.rmi.registry.LocateRegistry.createRegistry(1099);
          Receptor serv = new ReceptorImpl();
          Naming.rebind("rmi://localhost:1099/NomServico", serv);
        } catch (Exception e) {
          Alertas.showMensErro(ti,"Ocorreu um erro ao iniciar serviço
RMI:\n"+e.getMessage());
        }
    }
```

Com as máquinas interligadas e configuradas corretamente, como foi descrito anteriormente, o processo de comunicação para troca de mensagens acontecerá como mostra a Figura 3.4.2.

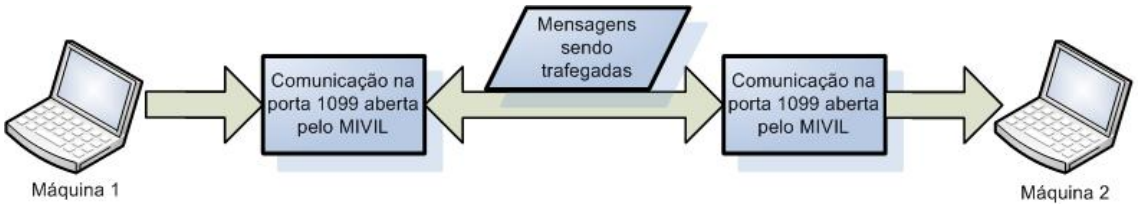

Figura 3.4.2 – Processo de comunicação do MIVIL.

3.5 TRANSFERÊNCIA DE MENSAGEM

Neste item será detalhado todo o processo de como o MIVIL aplica as técnicas de segurança do projeto, no momento em que o usuário envia uma mensagem através do sistema. Os códigos demonstrados neste item serão retirados do código fonte na linguagem Java, do MIVIL.

Inicialmente, ao clicar em um usuário que esteja na lista de contatos, o sistema gera as chaves de criptografia assimétrica do algoritmo RSA e também as chaves da assinatura digital do algoritmo DSA dos dois usuários que irão se comunicar. Logo em seguida, as chaves públicas dos algoritmos de cada usuário são trocadas entre os sistemas que irão se comunicar.

Código que gera as chaves:

```
//chaves para assinatura
        KeyPairGenerator chsAss = KeyPairGenerator.getInstance("DSA");
                KeyPair parChsAss = chsAss.generateKeyPair();
                chavePrivGeradaParaGerarAss = parChsAss.getPrivate();
                chavePubGeradaParaVerifAss = parChsAss.getPublic();

//chaves para criptografia
        KeyPairGenerator chsCrip = KeyPairGenerator.getInstance("RSA");
                KeyPair parChsCrip = chsCrip.generateKeyPair();
                chavePrivGeradaParaDescriptog = parChsCrip.getPrivate();
                chavePubGeradaParaCriptog = parChsCrip.getPublic();
```

Código que recebe as chaves do usuário que irá estabelecer conversa:

```
public void receberChaves(PublicKey chavePubRecebidaParaCriptog, PublicKey
chavePubRecebidaParaVerifAss){
```

```
this.chavePubRecebidaParaCriptog = chavePubRecebidaParaCriptog;
this.chavePubRecebidaParaVerifAss = chavePubRecebidaParaVerifAss;
```

Agora, o usuário “A” já possui as chaves públicas do usuário “B” e de posse dessas informações pode enviar uma mensagem de forma segura. Quando o botão “enviar” da tela onde o usuário escreve a mensagem for acionado, esta informação passará pelo algoritmo de criptografia assimétrica RSA, que utilizará a chave pública enviada pelo usuário destino da mensagem. Assim, a mensagem já está escrita de forma ilegível, e pode ser enviada pela rede, garantindo o seu sigilo, e a confidencialidade das informações.

Criptografando a mensagem original:

```
byte[] mensCriptografada = Crypto.criptografar( mensEmBytes,chavePubRecebidaParaCriptog
);
```

A mesma mensagem original, passará agora pelo algoritmo de assinatura digital DSA do usuário remetente, que utilizará sua chave privada para assinar digitalmente as informações que serão enviadas, e assim, garantir a autenticidade dos dados, provando que a mensagem foi realmente originada pela fonte anunciada como emissora da informação.

Assinando a mensagem original:

```
byte[] ass = Crypto.gerarAss(mensEmBytes,chavePrivGeradaParaGerarAss);
```

Depois dos procedimentos descritos acima, somente faltará a última etapa da segurança proposta, a integridade, antes da mensagem ser enviada de fato. E este processo se dá, de forma que na informação original, o algoritmo MD5 da função *Hash*, gere um resumo da mensagem, que é simplesmente um conjunto fixo de caracteres únicos que deverá ser enviado para o usuário destino, a fim de garantir que os dados não foram danificados ou modificados ao longo do trajeto.

Gerando um resumo da mensagem:

```
byte[] hash = Crypto.gerarHash(mensEmBytes);
```

Assim, todas as técnicas de segurança propostas foram aplicadas em cima da mensagem. O código abaixo monta a estrutura e mostra as informações que serão enviadas para o destinatário:

```
maqExterna.receberMensagem(mensCriptografada,ass,hash,Sessao.getInst().getIpLoc
al() );
```

A mensagem criptografada, a sua assinatura e o seu *hash* são enviados para que o sistema do destinatário receba essas informações e possa tratá-las, de modo que garanta ou não a segurança dos dados.

3.6 VALIDAÇÃO DA MENSAGEM

Após receber o pacote seguro, o MIVIL do destinatário segue alguns procedimentos para validar a mensagem.

Primeiramente a mensagem criptografada passa pelo processo de descriptografia. Neste caso, é utilizada a chave privada do usuário destino gerada pelo algoritmo RSA. Se não houver erros neste processo, pode-se afirmar que a mensagem foi trafegada de modo confidencial e que seu conteúdo continuou em sigilo.

Descriptografando a mensagem:

```
mensDescr=Crypto.descriptografar(mensCript,telaLocal.getChavePrivGeradaParaDescriptog());
```

O processo seguinte é verificar se a assinatura digital é válida. Para isso é necessário utilizar a chave pública do emissor, enviada anteriormente quando foi estabelecida a comunicação. Se não houver erro nesta verificação, significa que a mensagem é provinda do usuário que diz ser o emissor da mesma. Desta forma, o comunicador instantâneo MIVIL, consegue garantir a autenticidade das informações trocadas entre usuários.

Validando a assinatura:

```
if ( !Crypto.verifAss(mensDescr,ass,telaLocal.getChavePubRecebidaParaVerifAss()) ){
    telaLocal.addMensErroAss();
```

Por último, é necessário gerar o resumo da mensagem recebida através da função *Hash*, e comparar com o que foi gerado no emissor. Se ambos estiverem totalmente iguais, pode-se afirmar que as informações chegaram íntegras ao destino, ou seja, sem sofrer nenhuma modificação ao longo da sua transmissão. Desta forma, o MIVIL garante a integridade das mensagens trafegadas.

Comparando *Hash*:

```
byte[] hashGeradoLocal = Crypto.gerarHash(mensDescr);
        if ( !Crypto.compararHashs(hashGeradoLocal,hash) )
```

Se a validação da mensagem no destino ocorrer sem erros, pode-se afirmar que o MIVIL aumenta a segurança no tráfego das informações trocadas entre usuários. Além desta afirmação, a proposta inicial do projeto torna-se viável e de grande usabilidade.

Capítulo 4 – Testes, Simulações, Condições/Requisitos operacionais

Neste capitulo serão mostrados os resultados que foram obtidos e analisados nos testes e simulações dos módulos do projeto.

Também serão citados os problemas que foram ocorrendo durante o desenvolvimento deste projeto.

4.1 SIMULAÇÕES

A partir de agora, será apresentada a funcionalidade incluída no MIVIL que tem como objetivo simular a interceptação de uma mensagem enviada na comunicação instantânea, mostrando e disponibilizando os códigos gerados pelas técnicas de segurança aplicadas no projeto.

Os códigos exibidos são os mesmos que estão sendo enviados através da rede de computadores e que poderão ser vistos no caso de uma interceptação no trajeto da mensagem. Porém, da forma que os dados são enviados pelo MIVIL, não se pode identificar o conteúdo das informações trafegadas.

Para simular a alteração na estrutura da mensagem foi desenvolvida uma funcionalidade na janela de envio que é a “Abrir tela para simular interceptação da mensagem” (Figura 4.1.1). Ao selecionar esta opção, e clicar em “Enviar” para efetivar o envio da mensagem para o outro usuário, abrirá uma janela com uma interface que pode ser visualizada na Figura 4.1.2, e também pode ser visto que em cada caixa de texto estão sendo exibidos os códigos gerados pelas técnicas de segurança propostas no projeto.

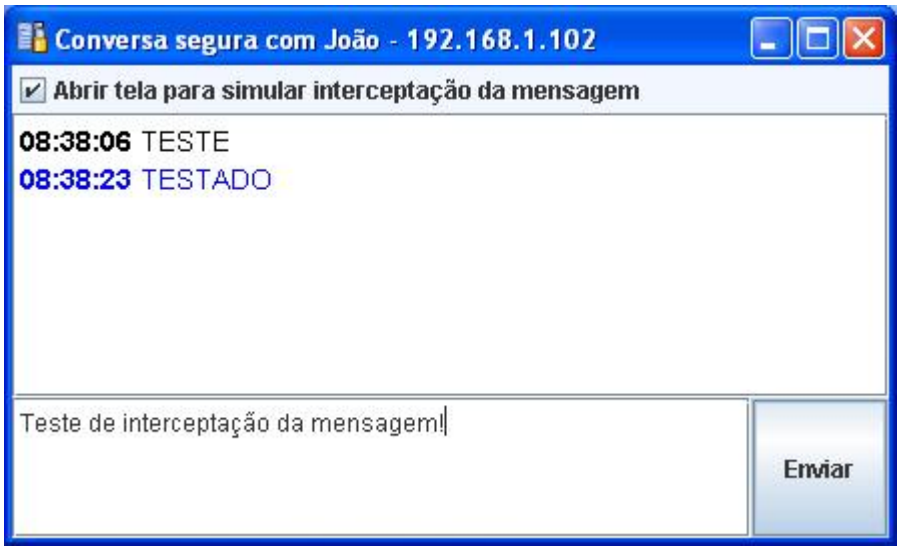

Figura 4.1.1 – MIVIL – Interceptação da mensagem.

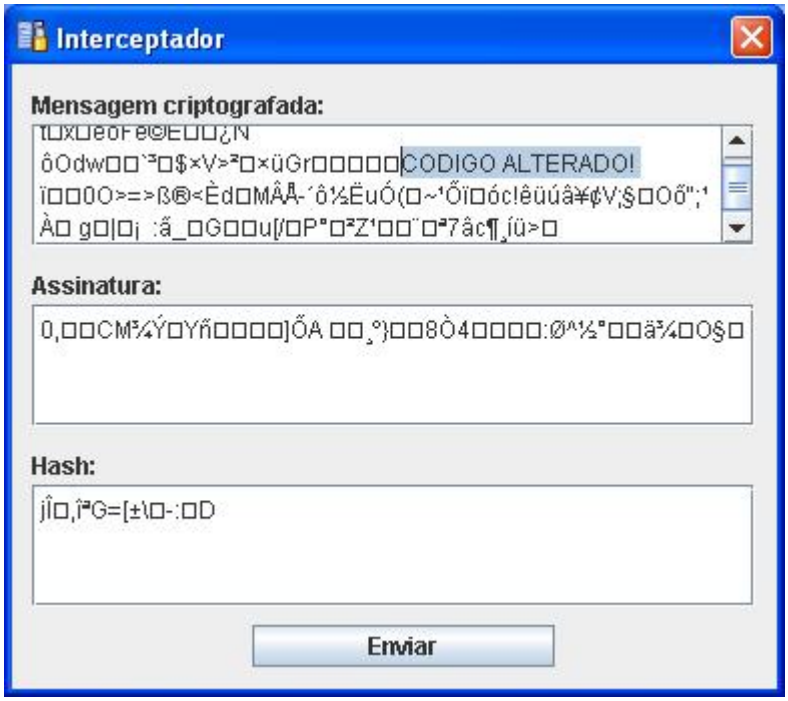

Figura 4.1.2 – MIVIL – Interceptação da mensagem criptografada.

Na Figura 4.1.2 foi simulada a alteração da mensagem criptografada, ao clicar em “Enviar”, o resultado pode ser visto na Figura 4.1.3. O usuário recebe uma notificação abordando sobre a interceptação, com a seguinte mensagem: “Não é possível descriptografar esta mensagem. Está em um formato de bytes inválido ou foi mudada durante a transmissão.”. Fazendo assim, com que o usuário saiba que a mensagem não é confiável e que ela pode ter sofrido

alguma modificação ao longo do trajeto, seja por um erro na transmissão ou por alguém mal intencionado.

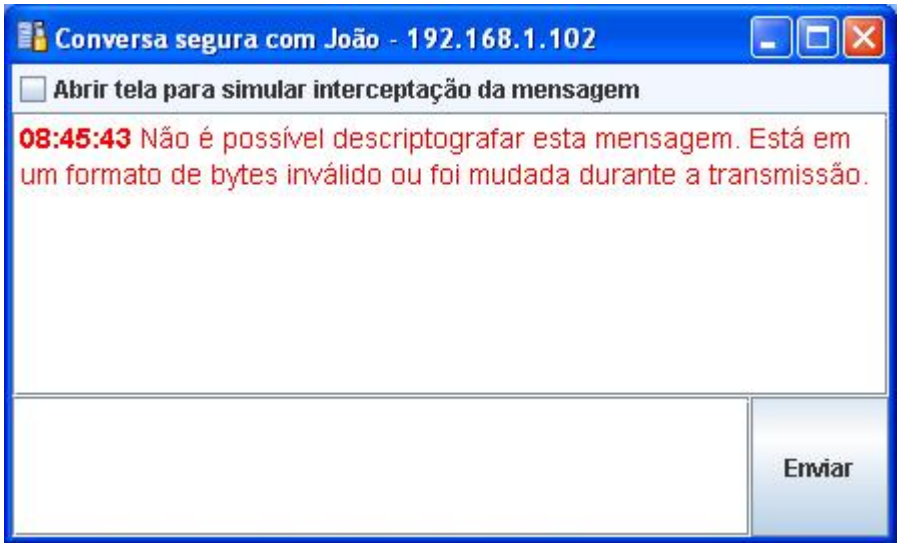

Figura 4.1.3 – MIVIL – Mensagem avisando interceptação da mensagem criptografada.

Foi simulada também a alteração da assinatura da mensagem, como pode ser visto na Figura 4.1.4. E de forma semelhante, o usuário recebe a notificação que houve uma interceptação na mensagem, porém o texto que vem comunicar é: “A mensagem recebida não confere com a assinatura digital!”, como na Figura 4.1.5.

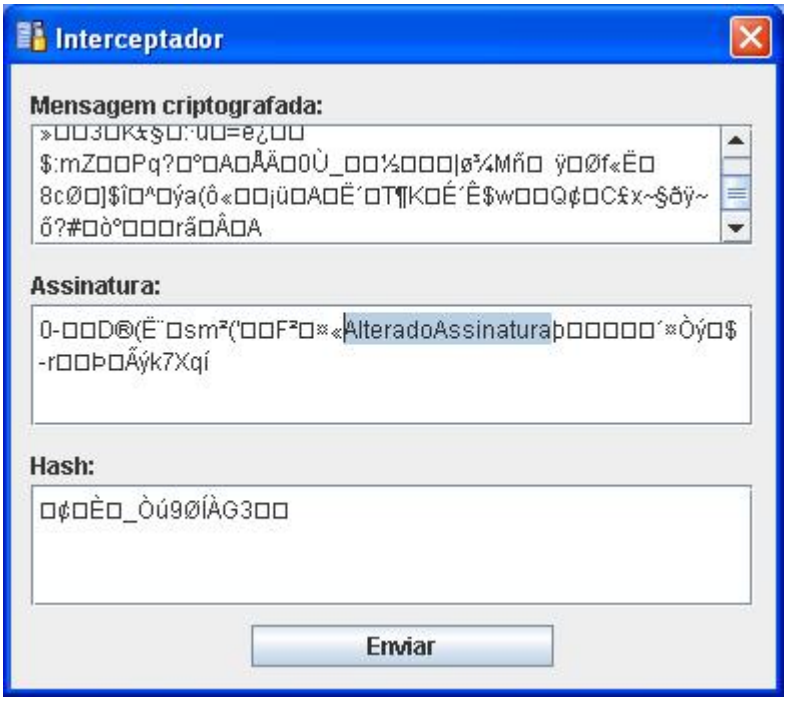

Figura 4.1.4 – MIVIL – Interceptação da assinatura.

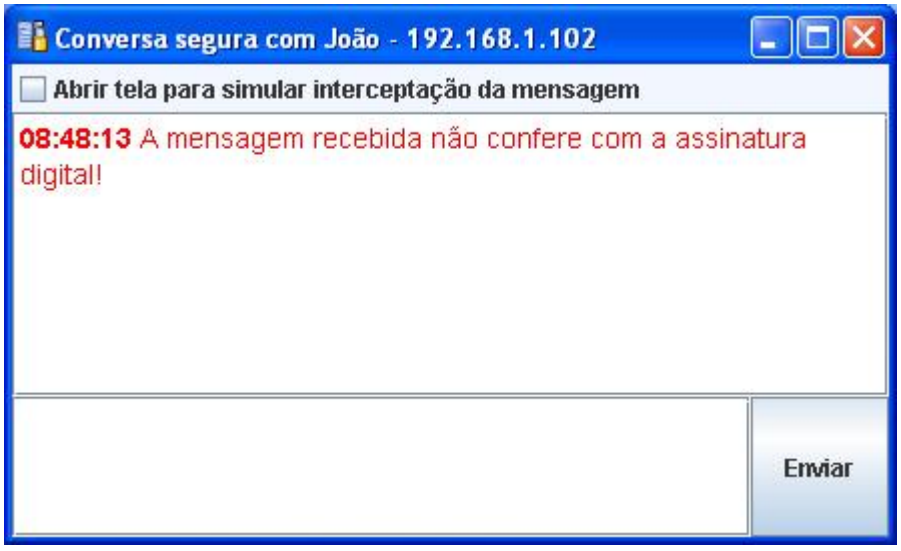

Figura 4.1.5 – MIVIL – Mensagem avisando interceptação na assinatura.

E por fim nas simulações no MIVIL, foi utilizado o interceptador para demonstrar a alteração do resumo da mensagem gerado pela função *Hash*, como na Figura 4.1.6, e a notificação que o usuário visualizará é: “A mensagem recebida não confere com o Hash!”, como pode ser visto na Figura 4.1.7.

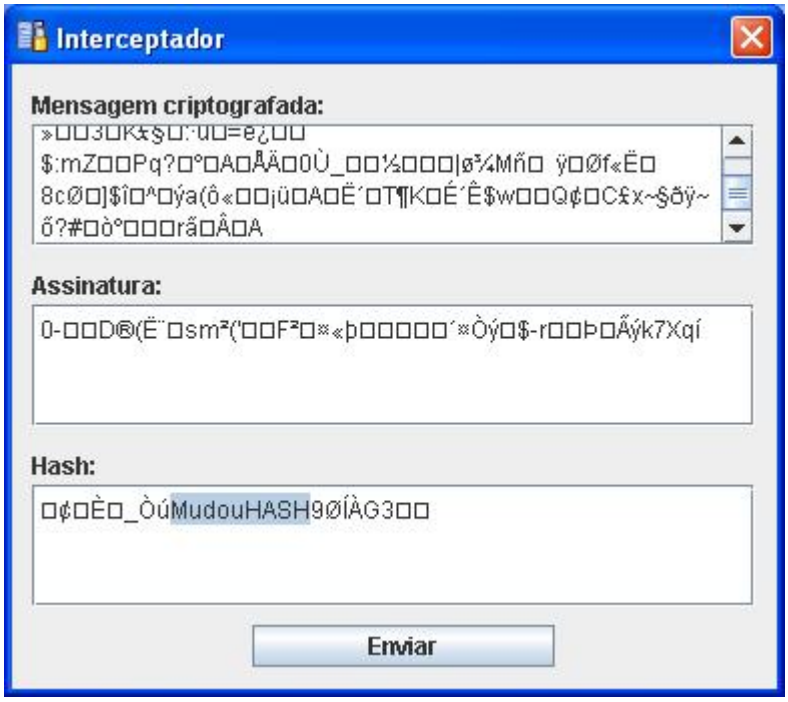

Figura 4.1.6 – MIVIL – Interceptação da assinatura.

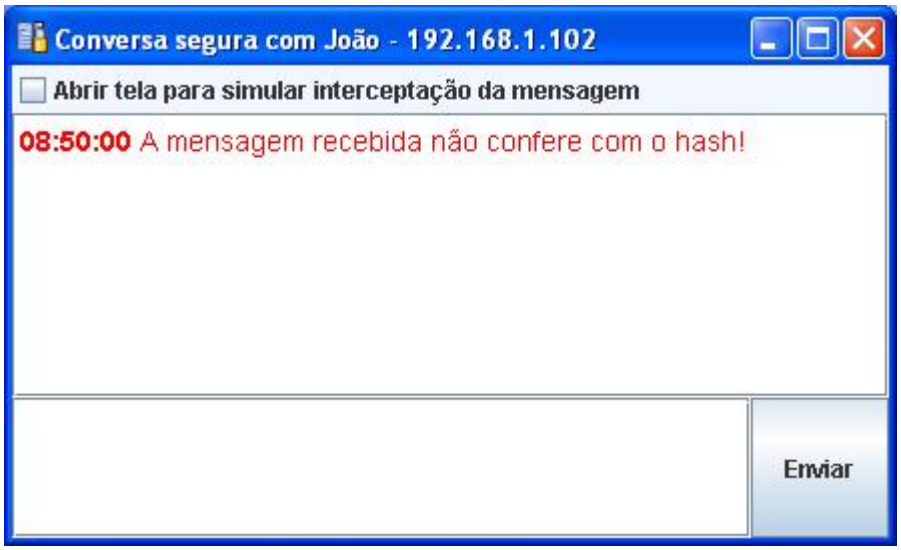

Figura 4.1.7 – MIVIL – Mensagem avisando interceptação no *Hash*.

Desta forma, foram realizadas as simulações de interceptação da mensagem utilizando o MIVIL, para demonstrar as soluções propostas do projeto. O software de comunicação instantânea desenvolvido está seguro e garantindo a autenticidade dos dados trocados no tráfego das mensagens.

4.2 CRÍTICAS DO SISTEMA

Neste item serão abordadas as críticas do sistema, ou seja, os tratamentos internos do MIVIL, englobando os possíveis erros premeditados e esperados após uma determinada ação.

4.2.1 Porta 1099

Quando o MIVIL for inicializado pelo usuário e a porta 1099 já estiver sendo utilizada por outro programa ou até mesmo por outro MIVIL em execução na mesma máquina, aparecerá uma mensagem de erro informando que a porta 1099 já está em uso (Figura 4.2.1.1).

Figura 4.2.1.1 – Erro quando a porta 1099 já estiver em uso.

O processo de comunicação do MIVIL para abertura da porta 1099 foi definido no item 3.4 desta monografia. A figura 4.2.1.2 abaixo mostra o fluxograma deste processo.

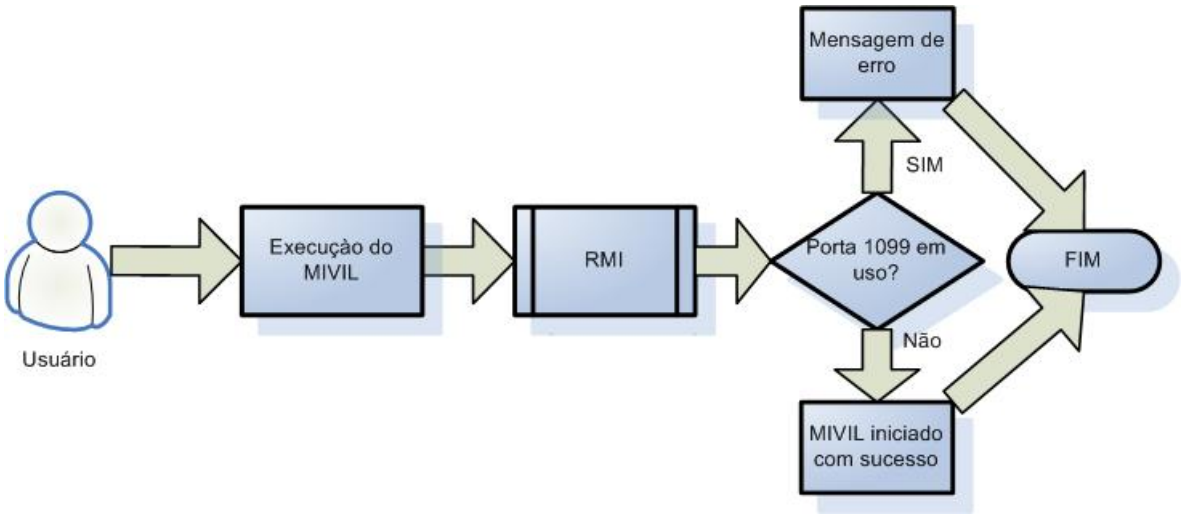

Figura 4.2.1.2 – Abertura da porta 1099 para utilização do MIVIL.

4.2.2 Endereço Local

O usuário do programa não poderá adicionar o seu próprio endereço de rede como parte da sua lista de contatos, pois o sistema foi desenvolvido para se comunicar remotamente, e caso o usuário tente cadastrar um endereço local, aparecerá a crítica como pode ser visto na Figura 4.2.2.1.

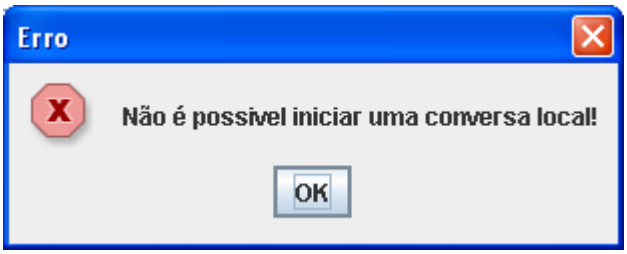

Figura 4.2.2.1 – Alerta ao tentar adicionar um endereço local como contato.

4.2.3 Endereço Padrão

Como foi citado no item anterior, o MIVIL não foi desenvolvido para aceitar comunicação local, por isso, ao tentar cadastrar o endereço padrão de máquina 127.0.0.1 na lista de contatos, o nome do usuário correspondente a este endereço aparecerá em cinza com o texto “Este contato é a máquina local”, conforme a figura 4.2.3.1.

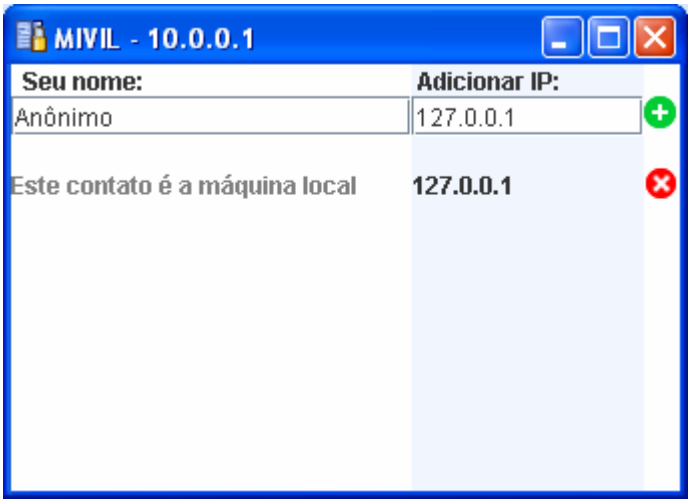

Figura 4.2.3.1 – O MIVIL não identifica usuário no endereço padrão 127.0.0.1

4.2.4 Sessão Finalizada

Caso um usuário “A” estiver trocando mensagens com um usuário “B” e finalizar o MIVIL, o usuário “B” deve fazer o mesmo para que eles possam se comunicar novamente quando ficarem ativos. Caso isso não aconteça e o

usuário “A” abrir o MIVIL novamente, o usuário “B” irá ver o status de “A” como ativo, porém, ao enviar uma mensagem aparecerá um alerta de erro como o da figura 4.2.4.1.

Este procedimento ocorre porque o usuário “B” guarda as chaves utilizadas para a comunicação com “A” até que o MIVIL seja finalizado. Como as chaves criptográficas são geradas a cada comunicação, quando “A” ficou ativo novamente as chaves já não eram as mesmas da primeira sessão.

Isto evita que alguém mal intencionado possa se passar pelo usuário “A” em uma conversa com o usuário “B” utilizando uma chave antiga, não válida.

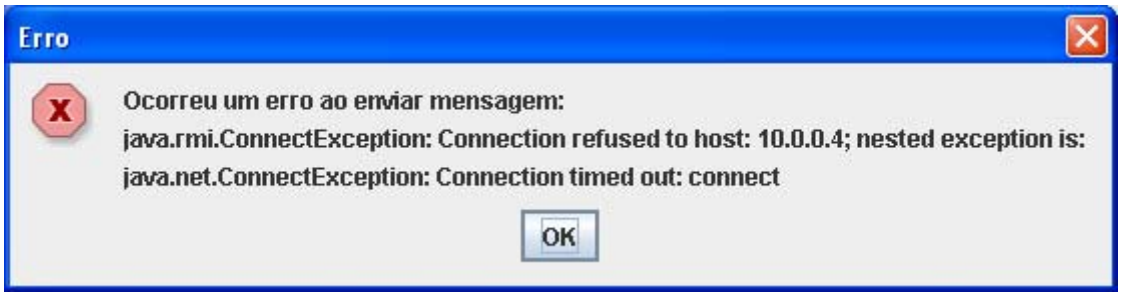

Figura 4.2.4.1 – Sessão finalizada.

4.2.5 Inclusão de usuário já cadastrado

O sistema foi projetado para não aceitar o cadastro de contatos com o mesmo endereço de rede que por ventura já estejam na lista de contatos. Caso ocorra a tentativa de adicionar um IP já existente o (botão que efetua a operação de adicionar um contato), ficará desabilitado.

4.3 TESTES

Foram realizados todos os testes das funcionalidades do comunicador instantâneo descritas no capítulo 3, tendo êxito em seus resultados finais.

O teste mais importante realizado foi a captura e análise das mensagens enviadas utilizando o MIVIL. Este teste foi necessário para se ter a certeza de

que o desenvolvimento do comunicador instantâneo ocorreu como o previsto e esperado, integrando toda segurança estudada para o projeto.

Para realizar o teste foi instalado um software para captura de pacotes, que trafegam em uma rede, chamado de Ethereal. Este software é de código aberto e a versão 0.10.14 utilizada no projeto pode ser adquirida no link http://www.ethereal.com/distribution/win32/all-versions/ethereal-setup-0.10.14.exe .

Uma mensagem foi enviada pelo emissor e a sua captura aconteceu no destinatário, que continha o software Ethereal instalado. O objetivo do teste era a localização da mensagem nos pacotes recebidos e provar que a informação estava codificada de acordo com as técnicas de segurança explicadas anteriormente e aplicadas no MIVIL.

A mensagem escolhida para ser enviada foi “Teste de captura” (Figura 4.3.1) e os códigos gerados a partir desta mensagem podem ser vistos na figura 4.3.2.

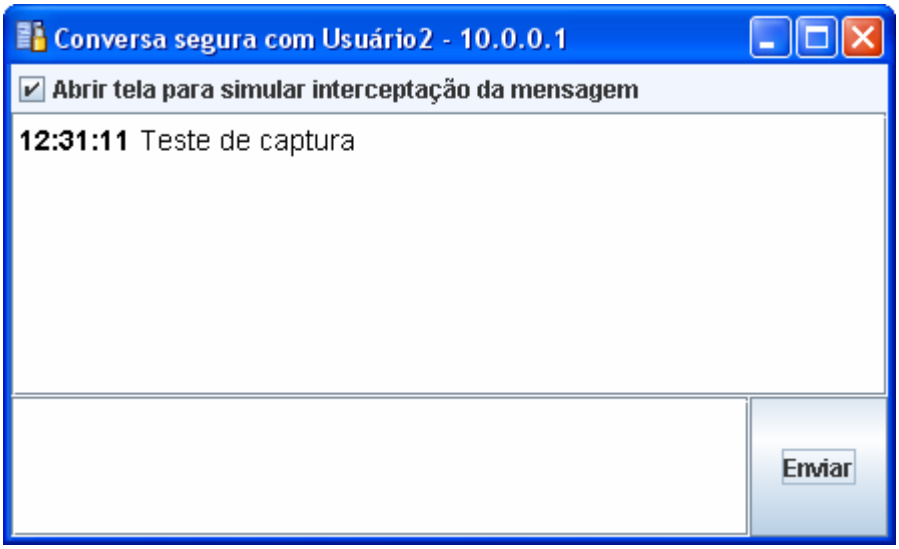

Figura 4.3.1 – Mensagem enviada.

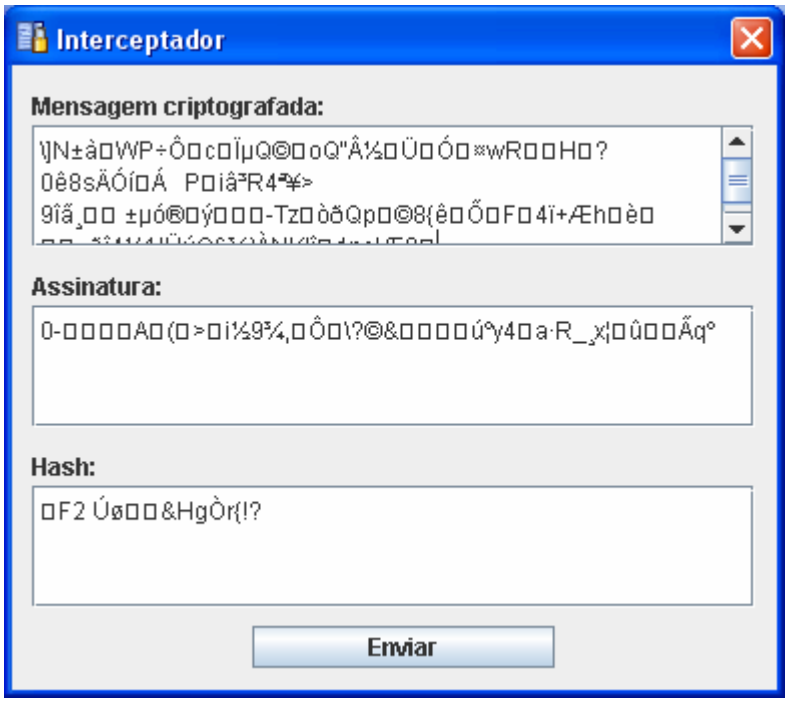

Figura 4.3.2 – Códigos gerados.

O software Ethereal estava capturando os pacotes no momento em que a mensagem chegou ao destino e algumas das informações obtidas podem ser vistas na Figura 4.3.3.

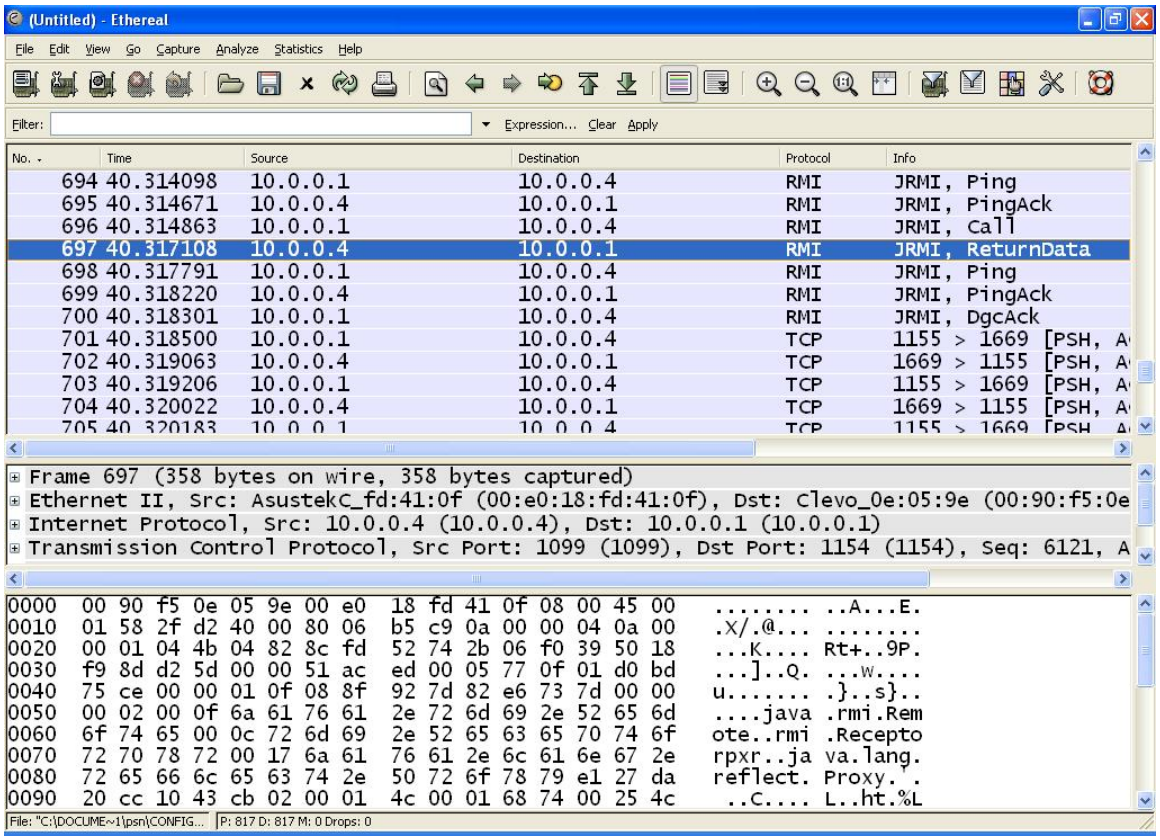

Figura 4.3.3 – Pacotes que chegaram na máquina destino.

Após a captura dos pacotes, foi realizada uma análise das informações para localizar a mensagem recebida. E para obter sucesso neste procedimento foi necessário procurar os códigos mostrados na figura 4.3.2 dentro das informações capturadas.

Para identificar o pacote foi necessário comparar alguns caracteres dos códigos gerados. A Figura 4.3.4 mostra o pacote que contêm a mensagem, a sua assinatura e o seu *Hash* escritos em códigos. Estas informações estão escritas em hexadecimal e traduzidas em códigos, conforme Figura 4.3.4.

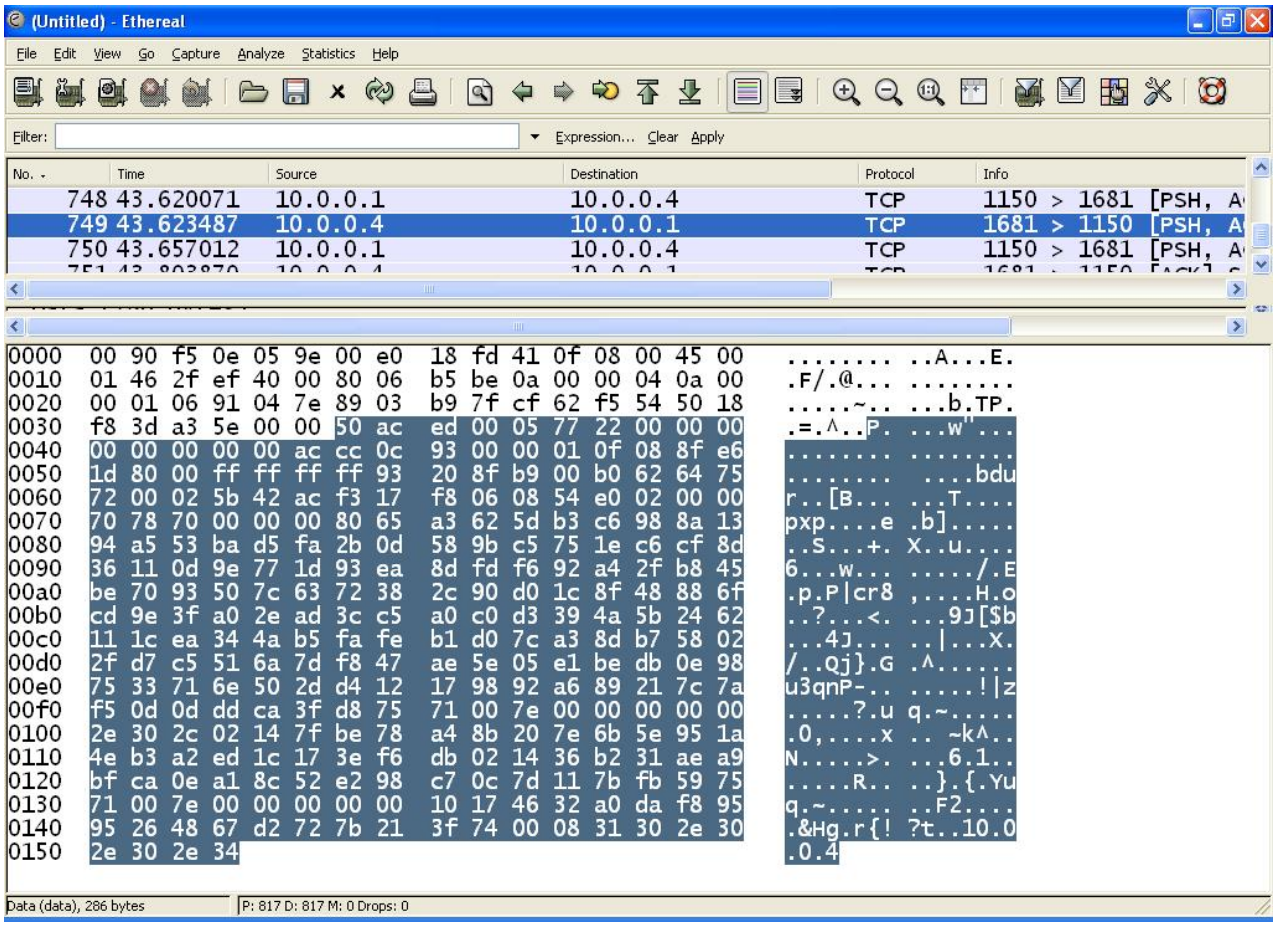

Figura 4.3.4 – Informação codificada.

Após localizar a mensagem e verificar que ela está totalmente codificada, tornou-se possível afirmar que as informações estão sendo trafegadas de forma segura. Desta forma, se alguém mal intencionado interceptar a mensagem enviada, este não conseguirá identificar o conteúdo das informações.

Com a realização deste teste fica confirmado que o desenvolvimento do MIVIL aconteceu de acordo com a implementação descrita e prevista no projeto.

4.4 DIFICULDADES

Este item explanará sobre as dificuldades encontradas durante todo o desenvolvimento do projeto.

4.4.1 Segurança do MIVIL

O conhecimento inicial das técnicas de segurança que seriam aplicadas no projeto, não bastava para que o sistema funcionasse de forma segura como descrito na proposta. Foi necessário um estudo mais completo e muito mais aprofundado para tornar o projeto viável e bastante seguro.

No momento da pesquisa foram encontradas falhas em algumas técnicas de segurança existentes, como a criptografia simétrica. Isto fez com que novas tecnologias fossem buscadas para suprir a necessidade inicial de desenvolver um comunicador seguro para troca de mensagens instantâneas. Neste caso, a criptografia escolhida para substituir a idéia inicial foi a assimétrica, que atendeu a todos os requisitos mínimos necessários para o projeto.

4.4.2 Assinatura Digital

Existem diversas aplicações e formas de implementação da assinatura digital. Muitas referências divergem no conceito e afirmam existir apenas uma forma de ser utilizada. Após um estudo minucioso foi possível entender toda a funcionalidade desta técnica de segurança que garante a autenticidade na troca de informações entre os usuários do MIVIL.

Diferentemente de como muitos autores afirmam, a assinatura digital não precisa necessariamente trabalhar em conjunto com uma função *Hash*.

Este método é muito utilizado em sistemas que não necessitam criptografar todo conteúdo das informações e por assinar somente o resumo da mensagem, fazendo com que o processo seja muito mais rápido. Porém, o bom desempenho não é afetado em um sistema como o MIVIL, que demanda pouco recurso de processamento no envio das informações.

Assim, a solução adotada foi assinar digitalmente todas as mensagens que forem trafegadas através do comunicador instantâneo, além de utilizar uma criptografia para garantir a confidencialidade dos dados, tornando-os muito mais seguros.

O DSA foi desenvolvido de tal maneira que já possui na sua estrutura a função *Hash* SHA-1 para gerar o resumo da mensagem. Entretanto, através do desenvolvimento em Java foi possível utilizar somente o DSA, desta forma, o projeto foi elaborado de acordo com a proposta inicial de implementar a função *Hash* MD5 para garantir a integridade das informações trafegadas pelo MIVIL.

4.4.3 Dificuldade na detecção da mensagem criptografada

Devido a falta de conhecimento para analisar a captura do tráfego de rede, o processo de localizar a mensagem criptografada nas informações capturadas levou um período maior que o imaginado para ser finalizado.

A tentativa de localização dos caracteres da mesma forma em que é exibida na interface do usuário, não foi obtida com sucesso. Isso fez com que outro software de captura fosse instalado, o Sniffer Portable da empresa Network General. O resultado da captura foi o mesmo, levando desta forma, à uma análise mais aprofundada do tráfego das informações.

Após várias tentativas, a mensagem foi localizada através da comparação de alguns caracteres que não são alterados na hora de gerar o pacote para ser transmitido pela rede. Assim, as mensagens começaram a ser identificadas e utilizando o Ethereal foi possível provar que as informações estão sendo trafegadas de forma segura, como previsto.

4.4.4 Falha na placa de rede

Em alguns momentos durante os testes com o MIVIL foram notadas diversas falhas na comunicação e troca de mensagens. Inicialmente, acreditava-se que o problema seria com o desenvolvimento do software, mas depois foi notado que a placa de rede do *notebook* estava apresentando intermitência mesmo com o status de conectada. A solução foi conseguir um outro equipamento para continuar realizando os procedimentos de testes previstos para o projeto, assim obtendo êxito nos demais testes.

4.5 DEMONSTRAÇÃO DA IMPLEMENTAÇÃO

A demonstração será feita de forma que sejam exibidas as funcionalidades e processos do MIVIL, que vieram suprir os problemas levantados na proposta do projeto, assim como, serão detalhados os testes realizados durante o desenvolvimento do projeto e os resultados obtidos.

Passo a passo da demonstração:

- Serão utilizados dois *notebooks* conectados diretamente através de um cabo par trançado, formando assim, uma rede ponto a ponto.
- Para a configuração de rede será atribuído um endereço IP para cada *notebook*, neste exemplo serão utilizados os endereços 10.0.0.1 e 10.0.0.2, ambos com máscara 255.255.255.0.
- Após o MIVIL ser inicializado em cada máquina, serão colocados os nomes de “Usuário1” e “Usuário2”, respectivamente, em cada programa, para que possam ser identificados na comunicação.
- O “Usuário1” adicionará o endereço IP 10.0.0.2 e logo depois o MIVIL irá identificá-lo, e aparecerá na lista de contatos do sistema como “Usuário2”. Da mesma forma, o “Usuário2” deverá adicionar o endereço IP 10.0.0.1 na sua lista de contatos.

- Após popular a lista de contatos, iniciará a comunicação através da troca de mensagens estabelecidas pelos dois usuários. Todas as funcionalidades e segurança da interface poderão ser visualizadas e testadas neste passo.
- O próximo procedimento da demonstração será a simulação de interceptação utilizando a funcionalidade “Abrir tela para simular interceptação da mensagem” do MIVIL. Nesta fase poderão ser visualizados todos os códigos gerados através das técnicas de segurança aplicadas na mensagem original. Para simular a interceptação da mensagem ao longo do seu trajeto, serão modificados, com auxilio do programa, os códigos gerados pela criptografia, pela assinatura digital e pela função *Hash.* Desta forma, o MIVIL do usuário destinatário ao tentar verificar todos os códigos, detectará que os mesmos estão divergentes e alertará ao usuário sobre uma possível modificação da mensagem.
- Para provar que as mensagens estão realmente trafegando de forma segura ao longo da rede de computadores, será utilizado um software chamado Ethereal, que tem como funcionalidade capturar pacotes na rede para análise das informações trafegadas. Com o Ethereal instalado em um dos *notebooks*, a fim de capturar a mensagem enviada pela outra ponta, as informações serão localizadas nos pacotes recebidos, e assim, poderá ser vista a mensagem totalmente codificada, provando desta forma a segurança do sistema.

Capítulo 5 – Conclusão

Conforme o estudo realizado neste projeto, foi verificado que a necessidade de segurança nas conversas via softwares de mensagem instantânea é primordial desde as grandes empresas, até um usuário doméstico. Sendo assim, este foi o foco do projeto, a busca da confidenciabilidade, integridade e autenticidade para essas aplicações.

A maior preocupação é com o sigilo e a confiança das informações, que podem ser comprometidas por um invasor da própria rede corporativa ou da internet. Podendo ser um programa malicioso ou alguém mal intencionado, tentando obter informações secretas ou confidenciais.

Com a utilização do MIVIL, veio a confiança dos quesitos citados acima, quitando assim, as falhas detectadas nos principais comunicadores instantâneos utilizados atualmente. Desta forma, obtemos um software de fácil utilização e com uma segurança melhorada quando comparada com as utilizadas hoje em dia.

E ainda, com detalhes importantes, como a utilização de algoritmos de criptografia muito utilizados no mercado e difíceis de serem quebrados, o que diminui os riscos em cima da informação trafegada.

Desta forma, é de grande interesse de empresas e usuários caseiros a utilização de um software como o desenvolvido neste projeto, devido a troca de informações importantes estar muito vulnerável a interceptação de terceiros, assim como a troca de mensagens entre usuários comuns, que se sentem lesados pela fragilidade dos comunicadores da atualidade.

5.1 PROJETOS FUTUROS

As perspectivas de evolução do projeto são:

- Desenvolver uma solução para que o MIVIL trabalhe em uma rede cliente-servidor, desta forma, tornando-se um sistema distribuído;
- Utilizar certificado digital para uma maior segurança, se possível;

-Implementar uma Autoridade Certificadora que possa aumentar a segurança distribuindo as chaves de criptografia e os certificados;

- Aprimorar o comunicador instantâneo, implantando transferência de arquivos, conversa por voz, conversa por vídeo, entre outras funcionalidades existentes nos grandes comunicadores instantâneos do mercado.

- Estruturar um ambiente onde o MIVIL possa ser utilizado sem restrições com bloqueio e redirecionamento de portas.

Referências Bibliográficas

[1].SCHWARTZMAN, Michel L. *O Brasil é líder no uso de Messenger*. Março, 2005 *(*http://www.lent.com.br/viu/archives/2005/03 em 01/09/2006).

[2].HINDOCHA Neal, 2003, *Threats to Instant Messaging* – Symantec Security Response.

[3].E-COMMERCE, *Dados Estatísticos sobre a Internet e Comércio Eletrônico (*http://www.e-commerce.org.br/STATS.htm#D em 01/09/2006).

[4].BARBER Joe, 2004*, Development of an instant messaging system to enable secure business communication*. (http://www.cs.bath.ac.uk/~amb/CM30076/projects.bho/2003 -4/JoeBarberDissertation.pdf em 03/09/2006).

[5].MOITINHO, Stoessel D. *Segurança em Sistemas Distribuídos*. Pós-Graduação em Sistemas Distribuídos. Universidade Federal da Bahia, 2001. (http://twiki.im.ufba.br/pub/GESI/WebHome/SeguranaemSistemasDistribudos.pdf em 22/08/2006).

[6].SINDIMENTAL, 2005, *Roubo de senha explode na web.* (http://www.o3.com.br/noticias.asp?id=24 em 23/08/2006).

[7].WIKIPÉDIA, 2006, *A enciclopédia livre* (http://pt.wikipedia.org/wiki/Intranet em 14/09/2006).

[8].WIKIPÉDIA, 2006, *A enciclopédia livre* (http://pt.wikipedia.org/wiki/Extranet em 14/09/2006).

[9].TREND MICRO, 2006*, Noções básicas sobre códigos maliciosos* (http://www.trendmicro.com/br/security/general/virus/overview.htm em 10/09/2006).

[10].WIKIPÉDIA, 2006, *A enciclopédia livre* (http://pt.wikipedia.org/wiki/Hackers em 11/09/2006).

[11].TORRES, Gabriel, 2001, *Redes de computadores* – curso completo – ed. Axcel books.

[12].VOSS, J.; PÉRICAS, F. *Compartilhamento de Informações Entre Computadores Através da Tecnlogia Peer-to-Peer Usando a Plataforma JXTA* – FURB, 2004.

[13].WIKIPÉDIA, 2006, *A enciclopédia livre*(http://pt.wikipedia.org/wiki/Firewall em 11/09/2006).

[14].SANS INSTITUTE, 2004, *Sans divulga relatório das 20 piores ameaças do mundo digital* (http://www.issabrasil.org/noticias_0058.asp em 28/08/2006).

[15].THE RADICATI GROUP, INC., *A Technology Market Research Firm*. (http://www.radicati.com , no dia 05/09/2006).

[16].FERREIRA, Fernando, 2003, *Segurança da Informação* – ed. Ciência Moderna.

[17].BERNADINHO, A.; CASTRO, A.; BERNADINHO, E.; RELVÃO, R. *Autenticação*, 2004. Mestrado e Curso de Especialização em Sistemas de Informação – Universidade do Minho.

[18].TANENBAUM, Andrew S., 2001, *Sistemas Operacionais Modernos* – ed. Campus.

[19].WIKIPÉDIA, 2006, *A enciclopédia livre* (http://pt.wikipedia.org/wiki/Autenticidade em 22/10/2006).

[20].KUROSE, J.; ROSS, K., 2003, *Redes de Computadores e a Internet – Uma Nova Abordagem* – ed. Addison Wesley, Pearson.

[21].SILVA, A.; FREITAS, D.; ZANCANELLA, L. Análise da Vulnerabilidade do DSA – Laboratório de Segurança em Computação – UFSC.

[22].WIKIPÉDIA, 2006, *A enciclopédia livre* (http://pt.wikipedia.org/wiki/Cabo_de_par_tran%C3%A7ado em 25/11/2006).

[23].MICROSOFT, 2003, *MSN Messenger é o comunicador instantâneo mais usado no Brasil (*http://www.microsoft.com/brasil/pr/2003/msn_maisusado.asp em 22/08/2006).

[24].INFOGUERRA, 2006*, Relatório: crimes pela Internet viram atividade lucrativa* (http://tecnologia.terra.com.br/interna/0,,OI929866-EI4805,00.html em 24/08/2006).

[25].NAR – National Association of Realtors**.** *Instant Messaging: With one hundred million users and couting, instant messaginf is proving to be an effective medium for communicating online.* NAR's Web Wizard Report, nr. 44, agosto/2003.

[26].VIRUSHELP, 2005*, Vírus para Messenger infecta em diversas línguas* (http://www.virushelp.org/index.php?name=News&catid=&topic=16&allstories=1 em 28/08/2006).

[27].MANNAN Mohammad, *On Instant Messaging Worms, Analysis, and Countermeasure* (http://portal.acm.org/citation.cfm?id=1103626.1103629 em 14/08/2006).

[28].IMEDIACONNECTION, *Nielsen//NetRatings' Week in Review (10/08/04)*

(http://www.imediaconnection.com/content/4422.asp em 28/08/2006).

[29]. FNDC, 2006 , *Reunião Multimídia* (http://www.fndc.org.br/internas.php?p=noticias&cont_key=34481 em 03/09/2006).

Apêndices

A seguir serão mostrados alguns código importantes que foram desenvolvidos para o projeto.

Como o MIVIL será vendido comercialmente, reservo o direito de não expor todos os códigos fontes do sistema.

Apêndice A

Parte da criptografia.

```
import java.io.UnsupportedEncodingException;
import java.security.InvalidKeyException;
import java.security.KeyPair;
import java.security.KeyPairGenerator;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.Signature;
import java.security.SignatureException;

import javax.crypto.BadPaddingException;
import javax.crypto.Cipher;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;

public class Crypto {

    public static String transfString(byte[] b) throws UnsupportedEncodingException{
            return new String(b,"ISO-8859-1");
    }
    public static byte[] transfBytes(String s) throws UnsupportedEncodingException {
            return s.getBytes("ISO-8859-1");
    }

    public static byte[] criptografar(byte[] b, PublicKey key) throws NoSuchAlgorithmException,
NoSuchPaddingException, InvalidKeyException, IllegalBlockSizeException,
BadPaddingException, UnsupportedEncodingException{
            Cipher cipher = Cipher.getInstance("RSA");
            cipher.init(Cipher.ENCRYPT_MODE, key );
            byte[] textoCriptog =  cipher.doFinal( b ) ;
            return textoCriptog;
    }
```

```
    public static byte[] descriptografar(byte[] b, PrivateKey key) throws
NoSuchAlgorithmException, NoSuchPaddingException, InvalidKeyException,
IllegalBlockSizeException, BadPaddingException, UnsupportedEncodingException{
            Cipher cipher2 = Cipher.getInstance("RSA");
            cipher2.init(Cipher.DECRYPT_MODE, key );
            byte[] textoDescriptog = cipher2.doFinal( b ) ;
            return textoDescriptog;
    }
    public static byte[] gerarHash(byte[] b) throws NoSuchAlgorithmException{
            MessageDigest md5 = MessageDigest.getInstance("MD5");
            return md5.digest(b);
    }
    public static byte[] gerarAss(byte[] b, PrivateKey key) throws NoSuchAlgorithmException,
InvalidKeyException, SignatureException{
            Signature sig = Signature.getInstance("DSA");
            sig.initSign(key);
            sig.update(b);
            return sig.sign();
    }
    public static boolean verifAss(byte[] mens, byte[] ass, PublicKey pubKey) throws
SignatureException, InvalidKeyException, NoSuchAlgorithmException{
            Signature clientSig = Signature.getInstance("DSA");
            clientSig.initVerify(pubKey);
            clientSig.update(mens);
            return clientSig.verify(ass);
    }
    public static boolean compararHashs(byte[] h1, byte[] h2) throws
UnsupportedEncodingException{
            return transfString(h1).equals( transfString(h2) );
    }

    /* GERACAO DE CHAVES:
     *
     *     // chaves para assinaturas
     *             KeyPairGenerator chsAss = KeyPairGenerator.getInstance("DSA");
     *             KeyPair parChsAss = chsAss.generateKeyPair();
     *             chavePrivGeradaParaGerarAss = parChsAss.getPrivate();
     *             chavePubGeradaParaVerifAss = parChsAss.getPublic();
     *     // chaves para criptografia
     *             KeyPairGenerator chsCrip = KeyPairGenerator.getInstance("RSA");
```

```
     *                KeyPair parChsCrip = chsCrip.generateKeyPair();
     *                chavePrivGeradaParaDescriptog = parChsCrip.getPrivate();
     *                chavePubGeradaParaCriptog = parChsCrip.getPublic();
     *
     */

}
```

Apêndice B

Troca de chaves na conversa, pacote da mensagem recebida e todo processo de verificação e validação da mensagem.

```
import java.rmi.Remote;
import java.rmi.RemoteException;
import java.security.PublicKey;

public interface Receptor extends Remote {

    public String getNome() throws RemoteException;

    public String getIp() throws RemoteException;

    public void iniciarConversa(PublicKey chavePubRecebidaParaCriptog, PublicKey
chavePubRecebidaParaVerifAss, String ipExterno) throws RemoteException;

    public void receberMensagem(byte[] mensCript, byte[] ass, byte[] hash, String ipExterno)
throws RemoteException;

}
```

```
import java.io.UnsupportedEncodingException;
import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;
import java.security.InvalidKeyException;
import java.security.NoSuchAlgorithmException;
import java.security.PublicKey;
import javax.crypto.BadPaddingException;
import javax.crypto.IllegalBlockSizeException;
import javax.crypto.NoSuchPaddingException;
import auxiliares.Alertas;
import visual.TelaConversa;
import controle.Sessao;
```

```
import crypto.Crypto;

public class ReceptorImpl extends UnicastRemoteObject implements Receptor {
    public ReceptorImpl() throws RemoteException {
            super();
    }
    public void iniciarConversa(PublicKey chavePubRecebidaParaCriptog, PublicKey
chavePubRecebidaParaVerifAss, String ipExterno) throws RemoteException {
            TelaConversa telaLocal = Sessao.getInst().getTela(ipExterno);
            If (telaLocal.getChavePubRecebidaParaCriptog()==null ||
telaLocal.getChavePubRecebidaParaVerifAss()==null){
    telaLocal.receberChaves(chavePubRecebidaParaCriptog,chavePubRecebidaParaVerifAss);
                    Receptor maqExterna = null;
                    try {
                            maqExterna = (Receptor) Naming.lookup(
"rmi://"+ipExterno+"/NomServico" );
                            telaLocal.setNomExterno(maqExterna.getNome());
    maqExterna.iniciarConversa(telaLocal.getChavePubGeradaParaCriptog(),telaLocal.getChav
ePubGeradaParaVerifAss(),Sessao.getInst().getIpLocal());
                            telaLocal.setVisible(true);
                    } catch (Exception e) {
                            Alertas.showMensErro(telaLocal,"Ocorreu um erro ao tentar
comunicação:\n"+e);
                    }
            }
    }

    public void receberMensagem(byte[] mensCript, byte[] ass, byte[] hash, String ipExterno)
throws RemoteException {
            TelaConversa telaLocal = Sessao.getInst().getTela(ipExterno);
            byte[] mensDescr;
            try {
mensDescr=crypto.descriptografar(mensCript,telaLocal.getChavePrivGeradaParaDescriptog());
            } catch (Exception e1) {
                    telaLocal.addMensErroCripto();
                    return;
            }
            try {
            if ( !Crypto.verifAss(mensDescr,ass,telaLocal.getChavePubRecebidaParaVerifAss())
){
```

```
                        telaLocal.addMensErroAss();
                        return;
                    }
                } catch (Exception e1) {
                    telaLocal.addMensErroAss();
                    return;
                }
                try {
                    byte[] hashGeradoLocal = Crypto.gerarHash(mensDescr);
                    if ( !Crypto.compararHashs(hashGeradoLocal,hash) ){
                        telaLocal.addMensErroHash();
                        return;
                    }
                } catch (Exception e1) {
                    telaLocal.addMensErroHash();
                    return;
                }
                try {
                    telaLocal.addConversaExterna( Crypto.transfString(mensDescr) );
                } catch (Exception e) {
                    Alertas.showMensErro(null,"Não foi possível ler a mensagem
descrptografada:"+e);
                }
                telaLocal.setVisible(true);
    }

    public String getNome() throws RemoteException {
                return Sessao.getInst().getNomLocal();
    }

    public String getIp() throws RemoteException {
                return Sessao.getInst().getIpLocal();
    }


}
```