UNIVERSIDADE DO EXTREMO SUL CATARINENSE – UNESC

CURSO DE CIÊNCIA DA COMPUTAÇÃO

RENAN ROSSO DA SILVA

GERENCIAMENTO DE SERVIDORES LINUX, A PARTIR DE DISPOSITIVOS CELULARES COM SUPORTE A J2ME, UTILIZANDO PROTOCOLO DE COMUNICAÇÃO SSH

CRICIÚMA, JUNHO DE 2010

RENAN ROSSO DA SILVA

GERENCIAMENTO DE SERVIDORES LINUX, A PARTIR DE DISPOSITIVOS CELULARES COM SUPORTE A J2ME, UTILIZANDO PROTOCOLO DE COMUNICAÇÃO SSH

Trabalho de Conclusão de Curso apresentado para obtenção do Grau de Bacharel em Ciência da Computação da Universidade do Extremo Sul Catarinense

Orientador: MSc Rogério Antônio Casagrande

CRICIÚMA, JUNHO DE 2010

RENAN ROSSO DA SILVA

Gerenciamento de Servidores Linux, a partir de dispositivos celulares com suporte a J2ME, utilizando protocolo de comunicação SSH

Submetido ao corpo docente do Curso de Ciência da Computação da Universidade do Extremo Sul Catarinense como um dos requisitos para obtenção do grau de Bacharel em Ciência da Computação.

Profa. MSc. Ana Claudia Garcia Barbosa
Coordenadora do Curso de Ciência da Computação

Banca Examinadora:

Prof. MSc. Rogério Antônio Casagrande (UNESC)
Orientador

Prof. MSc. Paulo João Martins (UNESC)

Prof. Esp. Sérgio Coral (UNESC)

RESUMO

Com o avanço dos dispositivos na telefonia celular, novos recursos podem ser desenvolvidos buscando aumentar a mobilidade e acessibilidade do usuário. Diariamente novos aplicativos J2ME surgem utilizando a transferência de dados via rede celular ou wireless, popularizando o acesso a *Internet* a partir dos aparelhos móveis. O aplicativo a ser implementado neste trabalho tem como principal objetivo o acesso a servidores Linux por meio de uma conexão SSH a partir de um dispositivo celular com suporte a J2ME. Este aplicativo disponibiliza ao usuário a possibilidade de conectar-se a qualquer servidor Linux ligado a uma rede, por meio de uma conexão segura, com acesso a funções de gerenciamento sem a necessidade de um segundo aplicativo instalado no servidor. A *inteface* com o usuário exibe as funções organizadas em um menu, facilitando o acesso. Atualmente com a tecnologia *touch screen* sendo popularizado entre os aparelhos celulares, o midlet desenvolvido proporciona o controle de seus servidores com extrema facilidade e agilidade e de forma remota.

Palavras-Chave: J2ME, SSH, Linux, 3G, HSPA, EDGE, Wireless, Telefonia Celular.

ABSTRACT

With the advances in mobile comunications, new applications can be developed to increase the user's mobility and acessibility. Everyday new J2ME applications are created to ease the access of mobile devices to internet, transferring data through wi-fi or cellular networks. The main objective of the application developed in this research is to provide access to a Linux server using the SSH protocol through a mobile device supporting the J2ME technology. This application make it possible to the user to connect to any Linux server available in a network, through a secure connection, providing management services without the need of an additional application running on the server. The user interface let the user pick some options arranged in a simple menu, making it easy to use. Nowadays, with the increasing popularity of touch screen technology on mobile devices, the midlet developed in this research provide a quick way to manage one's servers remotely, with great ease.

Keywords: J2ME, SSH, Linux, 3G, HSPA, EDGE, Wireless, Cellular network.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3G	Terceira Geração
AMPS	*Advanced Mobile Phone Service* (AMPS)
AT&T	*American Telephone and Telegraph*
CDC	*Connected Device Configuration*
CDMA	*Code Division Multiple Access*
CLDC	*Configuration Limited Device Configuration*
DAS	*Digital Signature Algorithm*
ECDSA	*Elliptic Curve Digital Signature Algorithm*
EDGE	*Enhanced Data rates for GSM Evolution*
EGPRS	*Enhanced General Packet Radio Service*
GPRS	*General Packet Radio Services*
GSM	*Global Service for Mobile*
HSDPA	*High-Speed Downlink Packet Access*
HSPA	*High-Speed Packet Access*
HSPA+	*High-Speed Packet Access +*
HSUPA	*High-Speed Uplink Packet Access*
IMT-2000	*International Mobile Telecommunications-2000*
IP	*Internet Protocol*
IRC	*Internet Relay Chat*
ITU	*International Telecommunication Union*
J2EE	*Java 2 Enterprise Edition*
J2ME	*Java Plataform Micro Edition*
J2SE	*Java 2 Standard Edition*
JAR	*Java ARchive*

JDK	*Java Development Kit*
JVM	*Java Virtual Machine*
KB	*Kilobyte*
Kbits/s	Quilobits por segundo
Km/h	Quilometro por hora
MB	*Megabyte*
Mb/s	Megabit por segundo
MHz	*Mega-Hertz*
MIDlet	MID *Profile application*
MIDP	*Mobile Information Device Profile*
MIMO	*Multiple Input Multiple Output*
MSC	*Mobile Switching Center*
PDA	*Personal Digital Assistant*
PID	*Process IDentifier*
RAM	*Random-Access Memory*
RISC	*Reduced Instruction Set Computer*
ROM	*Read Only Memory*
RSA	*Rivest, Shamir and Adleman*
SSH	*Secure Shell*
SSH-AUTH	*Secure Shell - Authentication Protocol*
SSH-CONN	*Secure Shell - Connection Protocol*
SSH-TRANS	*Secure Shell - Transport Layer Protocol*
TCP	*Transmission Control Protocol*
TDMA	*Time Division Multiple Access*
UMTS	*Universal Mobile Telecommunication System*
USB	*Universal Serial Bus*

VNC	*Virtual Network Computing*
WCDMA	*Wideband Code Division Multiple Access*

SUMÁRIO

INTRODUÇÃO

Administradores de sistemas são responsáveis pela manutenção dos sistemas e de serviços, possui uma grande sensibilidade dos sistemas e necessita manter rigoroso acompanhamento de como e quando os sistemas são modificados (WADLOW, 2004).

Administradores de Sistemas e de Redes devem acompanhar seu ambiente virtual corporativo para intervir perante qualquer mudança que possa prejudicar o bom funcionamento da rede e dos sistemas. A dependência deste profissional acaba trazendo impedimentos e atrasos diante de problemas ocasionados quando os mesmos não estão disponíveis na empresa.

Segundo Morimoto (2006) uma das grandes vantagens do uso de Linux em servidores, é a facilidade do acesso remoto, tanto via linha de comando, Secure Shell (SSH), quanto com acesso à interface gráfica (VNC, FreeNX, SSH).

Estes profissionais utilizam largamente o recurso de acesso remoto, podendo administrar seus servidores e estações, a partir de qualquer computador, desktop ou notebook, com acesso a *Internet*. Em muitos casos, os administradores não terão acesso a estes dispositivos, caso estejam em viagem ou em zona rural, por exemplo.

Segundo Bortolini (2004) os administradores podem receber notificações por meio de um telefone celular, por serviços prestados pelo servidor, avisando sobre problemas existentes na rede.

Uma solução encontrada para facilitar a administração remota seria o uso dos dispositivos móveis. A partir de uma conexão a *Internet*, o mesmo poderia conectar-se a um servidor Linux e com uma interface de fácil uso agilizaria e facilitaria o controle do servidor.

Baseado nesses dados, este trabalho tem por função o desenvolvimento e documentação de um sistema com certo grau de segurança para o gerenciamento de servidores Linux por meio de dispositivos celulares.

1.1 OBJETIVO GERAL

Desenvolver um aplicativo J2ME para gerenciamento de servidores Linux, utilizando o protocolo de comunicação SSH.

1.2 OBJETIVOS ESPECÍFICOS

De acordo com o objetivo geral deste projeto, os objetivos específicos são:

a) desenvolver um aplicativo, com interface que facilite o acesso e o acompanhamento dos Administradores de sistemas e de rede, em seus servidores e estações;
b) compreender o funcionamento das redes EDGE e HSDPA;
c) aplicar projeto em uma rede.

1.3 JUSTIFICATIVA

Com o grande crescimento da rede móvel, a cobertura mundial proporciona as pessoas uma incrível possibilidade de conexão, oferecendo além da telefonia a capacidade de envio e recebimento de dados a partir de um dispositivo celular.

Neste princípio, o desenvolvimento de um aplicativo para o gerenciamento de servidores Linux disponibilizaria ao usuário uma grande facilidade de acesso ao seu sistema, sendo que o acesso seria feito a partir do dispositivo celular, utilizando uma conexão EDGE

ou 3G (HSDPA) e uma traria uma capacidade maior de mobilidade para o administrador da rede, pois ele poderia obter dados de seu sistema independente da sua localização.

Este aplicativo disponibilizaria as principais funções para administração do servidor, como iniciar e finalizar processos e serviços, possibilidade de desconectar usuários conectados, verificar os principais status dos servidores, como uso do CPU, memória RAM disponível e espaço utilizado nos discos rígidos.

> Qualquer atividade de administração, correção de erros do sistema ou assistência para resolução de problemas que não seja realizada de forma direta, ou seja, em que o profissional não lide pessoalmente com a instalação física do servidor ou estação de trabalho com problemas, é considerada um tipo de administração remota. (CARMONA, 2007, p.324)

Segundo Barrett (2001) o protocolo SSH oferece autenticação e criptografia, bem como a integridade dos dados transmitidos. É solicitada uma identificação digital para realizar a conexão cliente / servidor, todos os dados transmitidos são criptografados e assinados digitalmente, se um terceiro captar e modificar os dados transmitidos, o SSH detectará esse fato.

Ainda segundo Barret (2001) o servidor Linux ao qual o cliente se conectará necessita somente do serviço sshd configurado e que o mesmo esteja conectado a rede, assim, um aplicativo utilizando deste protocolo de comunicação para se conectar a um servidor obterá sucesso na comunicação sem a necessidade de desenvolver e instalar um segundo aplicativo no mesmo.

Com o desenvolvimento de um aplicativo para dispositivos celulares, na plataforma *Java 2 Micro Edition* (J2ME), e que apresente uma interface que ofereça fácil acesso aos principais recursos de um servidor Linux, o usuário ganharia uma grande agilidade no acesso e na administração dos servidores e estações.

1.4 ESTRUTURA DO TRABALHO

No Capítulo 1 é apresentada a introdução deste trabalho, trazendo o objetivo geral, objetivos específicos e a justificativa para o desenvolvimento deste projeto.

Capitulo 2 é iniciada a fundamentação teórica, com abordagem nas redes celulares, sua evolução, a estrutura das redes GSM e UMTS e as vantagens de utilização das mesmas.

O terceiro capítulo aborda o protocolo Secure Shell, com suas vantagens de utilização, funcionamento da conexão cliente-servidor, protocolos envolvidos na segunda versão do SSH e a sua relação com o sistema operacional Linux.

Já o sistema operacional Linux é estudado no Capítulo 4, onde temos algumas definições, finalidades e vantagens no uso do sistema operacional.

No Capitulo 5 temos uma breve abordagem sobre a tecnologia Java a sua versão destinada a dispositivos limitados, o J2ME.

No sexto capítulo temos a descrição de alguns trabalhos correlatos, que têm por objetivo final os mesmos deste trabalho desenvolvido, o auxílio na administração de servidores a partir de dispositivos móveis.

No capítulo 7 temos o capitulo dedicado ao desenvolvimento do aplicativo, nele temos a metodologia do trabalho, a arquitetura do aplicativo, o processo de desenvolvimento e por fim o aplicativo implementado.

2 REDES CELULARES

Buscando uma maior mobilidade para o uso do aplicativo, o uso das redes celulares nos disponibiliza uma imensa liberdade quanto ao acesso a *Internet*. Uma conexão pode ser iniciada a partir de qualquer ponto dentro da cobertura de uma rede celular.

De acordo com Dahlman (2007) os primeiros aparelhos de comunicação móvel começaram a ser comercializados no fim da década de 40 pela AT&T, devido ao tamanho e peso desses equipamentos, estes eram instalados em veículos, que recebiam o rótulo de *Car-borne telephony service*, ou seja, carros com serviço telefônico. Mesmo com serviço limitado, foi comercializado em muitos países durante os anos 1950 e 1960, porém os usuários podiam ser contatos, no máximo, em milhares.

Segundo Tanenbaum (2003) em todas as redes de telefonia móvel, a região onde será instalada é dividida por várias células, por este motivo, os dispositivos móveis utilizados para conexão nas redes, são chamados de celulares.

Estas células são geradas por uma estação-base, que serve de ponto de acesso a rede para os dispositivos móveis que estão posicionados na área que a estação-base abrange. Estas células não possuem um formato definido, o que pode gerar alguns pontos sem cobertura celular. Abaixo, na Figura 1 é apresentada a estrutura de uma rede de comunicação celular.

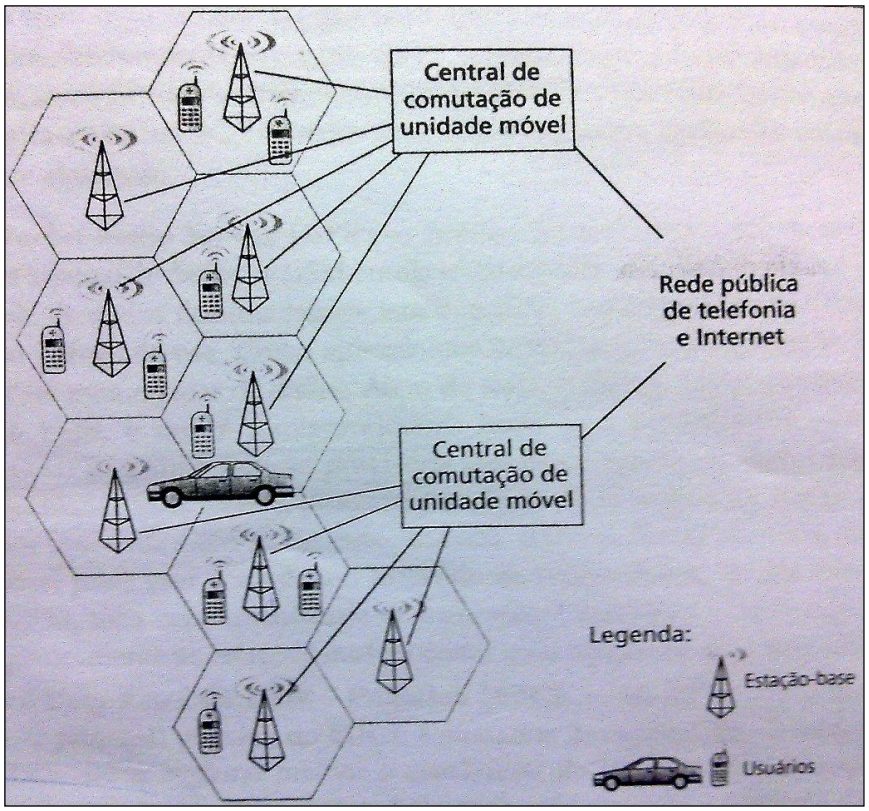

Figura 1. Componentes da arquitetura de rede celular
Fonte: KUROSE, J.; ROSS, K. (2006, p.445)

Conforme Kurose (2006) cada estação-base está conectada a uma central de comutação de unidade móvel (*Mobile Switching Center* – MSC), que gerencia as chamadas de e para usuários móveis. Um MSC possui o mesmo funcionamento de uma central de comutação telefônica, adicionada a capacidade de gerenciar a mobilidade de seus usuários. Esta arquitetura de rede é utilizada para as três gerações da telefonia móvel.

A primeira geração de telefones celulares era analógica; a segunda geração era digital. Da mesma maneira que não havia nenhuma padronização mundial durante a primeira geração, também não havia nenhuma padronização durante a segunda (TANENBAUM, 2003).

De acordo com Bortolini (2004) os sistemas de telefonia celular utilizam vários métodos de acesso ao meio físico, entre eles, o *Advanced Mobile Phone Service* (AMPS), *Time Division Multiple Access* (TDMA), *Code Division Multiple Access* (CDMA) e o *Global Service for Mobile* (GSM).

O método de acesso ao meio físico consiste no método de modulação utilizado para transmissão dos pacotes de comunicação de voz e dados entre o dispositivo móvel e a estação-base.

As tecnologias citadas por Bortolini pertenciam a primeira (AMPS) e segunda geração de celulares (TDMA, CDMA e GSM). Segundo Sverzut (2005) com a necessidade de aumento da capacidade de transmissão de dados, novas tecnologias surgiram, tendo base a rede GSM. Criaram-se então as tecnologias *General Packet Radio Services* (GPRS) e *Enhanced Data rates for GSM Evolution* (EDGE), denominadas redes de 2,5 e 2,75 gerações respectivamente.

Em 2003 quase todos os lugares do mundo já utilizavam um sistema chamado GSM - Sistema Global para Comunicações Móveis - e ele estava começando a ser usado até mesmo nos EUA, em escala limitada (TANENBAUM, 2003). Hoje as redes GSM já estão presentes em 219 países do mundo (GSM Association, 2009, tradução nossa).

Segundo Sverzut (2005) as redes 3G tiveram inicio com a implantação da tecnologia *Universal Mobile Telecommunication System* (UMTS), um sistema que possui total compatibilidade com o GPRS e o EDGE, e segundo a GSM Association (2009), o UMTS pode oferecer serviços multimídias, como suporte a música, TV e vídeo, conteúdo de entretenimento e acesso a *Internet*, com downloads efetuados a até 384 Kb/s.

No Brasil, a tecnologia *High-Speed Packet Access* (HSPA) está presente nos principais centros urbanos, e GSM em quase todo o território nacional povoado. Na Figura 2 temos o mapa da cobertura das redes GSM e 3G do Sul e Suldeste do Brasil.

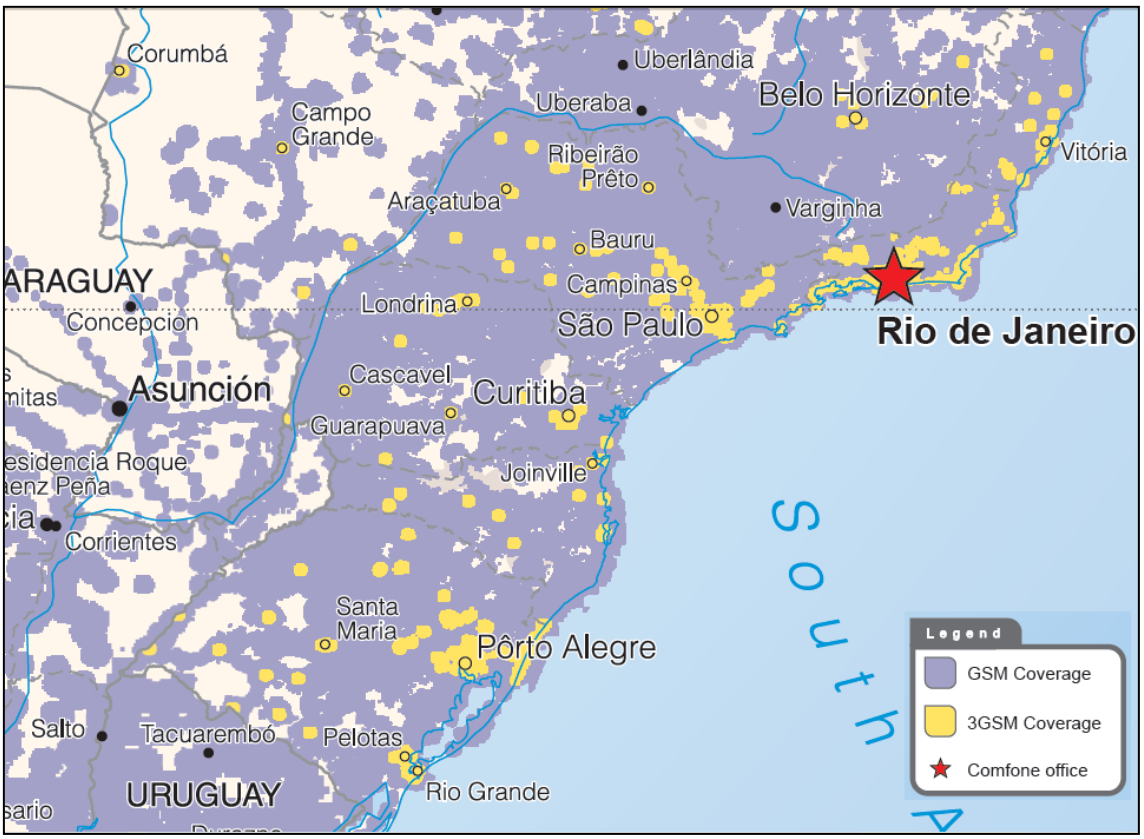

Figura 2. Cobertura GSM Sul e Sudeste do Brasil
Fonte: Adaptado de GSM Association (2009).

Segundo mapas disponibilizados pela GSM Association (2009) a cobertura GSM no centro-sul e no nordeste do Brasil cobre quase toda a região, se mostrando falha na região do pantanal e da Amazônia no centro-oeste e norte do país.

Com base nestes dados, neste projeto darei ênfase nas redes EDGE e HSPA, pois a EDGE surge do aperfeiçoamento das redes GSM, resultando em uma rede que utiliza toda a cobertura GSM e oferece maior capacidade de transmissão de dados. E nas redes HSPA, que surgiram da necessidade de maior capacidade de transmissão de dados.

2.1 GLOBAL SERVICE FOR MOBILE (GSM)

Conforme Kurose (2006) o GSM surgiu da necessidade da sociedade européia de criar um sistema digital que substituísse os sistemas de primeira geração, incompatíveis entre

si, provendo a capacidade a um usuário móvel de ultrapassar fronteiras sem a interrupção do sinal celular.

Com a implantação desta tecnologia na Europa os usuários poderiam utilizar o mesmo aparelho móvel em diversos países. O que não acontecia na geração celular anterior, onde países utilizavam diferentes meios de acesso a rede celular.

Segundo Dahlman (2007) a tecnologia GSM foi lançada em 1992 e tornou-se rapidamente o padrão celular mais utilizado no mundo com mais de 2 bilhões de usuários em 2007.

De acordo com a GSM Association (2009) a cobertura GSM abrange mais de 80% da população do mundo e oferece conexões de dados com velocidade de até 9,6 kbits/s.

2.1.1 Enhanced Data rates for GSM Evolution (EDGE)

A tecnologia *Enhanced Data rates for GSM Evolution* (EDGE) é uma evolução da tecnologia GPRS, também conhecida como GPRS Melhorado (*Enhanced General Packet Radio Service* – EGPRS) (TANENBAUM, 2003).

O aprimoramento do sistema GSM é chamado de EDGE, o desenvolvimento do EDGE, inicialmente foi focado na maneira que os dados chegavam ao usuário através da introdução de taxas de modulação de maiores ordens em GSM. Durante a continuação do trabalho, o foco mudou para o reforço da tecnologia GPRS (DAHLMAN, 2007, tradução nossa).

Segundo a GSM Association (2009) a tecnologia EDGE proporciona um aumento de três vezes a capacidade de transmissão de dados em relação a tecnologia GPRS. EDGE usa a mesma estrutura da rede GSM, e tem fácil implantação, resumindo-se a uma atualização de software na rede de transmissão.

Dahlman (2007) relata que o desenvolvimento da tecnologia EDGE tinha como objetivo o aumento de 50% na capacidade de transmissão de dados e voz, o aumento em 100% nas taxas de pico nas transmissões *downlink* e *uplink*, aumentar a cobertura de dados e voz e diminuir a latência de conexão.

2.2 UNIVERSAL MOBILE TELECOMMUNICATION SYSTEM (UMTS)

Em meados da década de 1980, a *International Telecommunication Union* (ITU) criou o padrão único de uma família de tecnologias intitulada "IMT-2000", para servir como base do sistema de terceira geração (3G) de celular (AUDREY, 2001, tradução nossa).

A partir da criação da IMT-2000, empresas e órgãos reguladores de todo o mundo passaram a estudar e propor soluções para a criação de um sistema que atingisse os requisitos necessários (SVERZUT, 2005).

De acordo com Sverzut (2005) entre os requisitos do padrão IMT-2000 destacava-se algumas características interessantes, como atributos do usuário e tipos de serviço. Os usuários são definidos como estacionário, pedestre, veicular, veicular de alta velocidade, aeronáutico e satélite, que variam de 0 km/h a 27.000 km/h. E a categoria serviços formada por serviços de voz, áudio, texto, imagem, vídeo, sinalização e dados.

Diversas empresas e órgãos reguladores propuseram sistemas para serem implantados, no total, 13 foram submetidos. Porém, apenas quatro foram selecionados pela ITU. Na prática temos a evolução da tecnologia UMTS (*Wideband* CDMA) que segue os padrões GSM.

Desta forma temos a uma tecnologia compatível com as redes GSM, assim, um dispositivo que conecte a uma rede 3G, possui compatibilidade com as redes GSM sem a necessidade da implantação de um hardware adicional no dispositivo móvel.

Segundo a GSM Association (2009) são 1618 dispositivos compatíveis com o 3G. Entre eles 509 aparelhos celulares, 393 notebooks e 169 modems USB.

2.2.1 High-Speed Packet Access (HSPA)

Com a evolução do WCDMA temos o aperfeiçoamento da transmissão de dados, na quinta versão, temos o surgimento da tecnologia HSDPA, que traz maior desempenho na transmissão de dados *downlink*, e logo depois, na sexta versão, complementada com a *Enhanced Uplink* aumentando a capacidade de transmissão de dados do dispositivo móvel (DAHLMAN, 2007, tradução nossa). Na Figura 3 temos um diagrama com a evolução da tecnologia WCDMA.

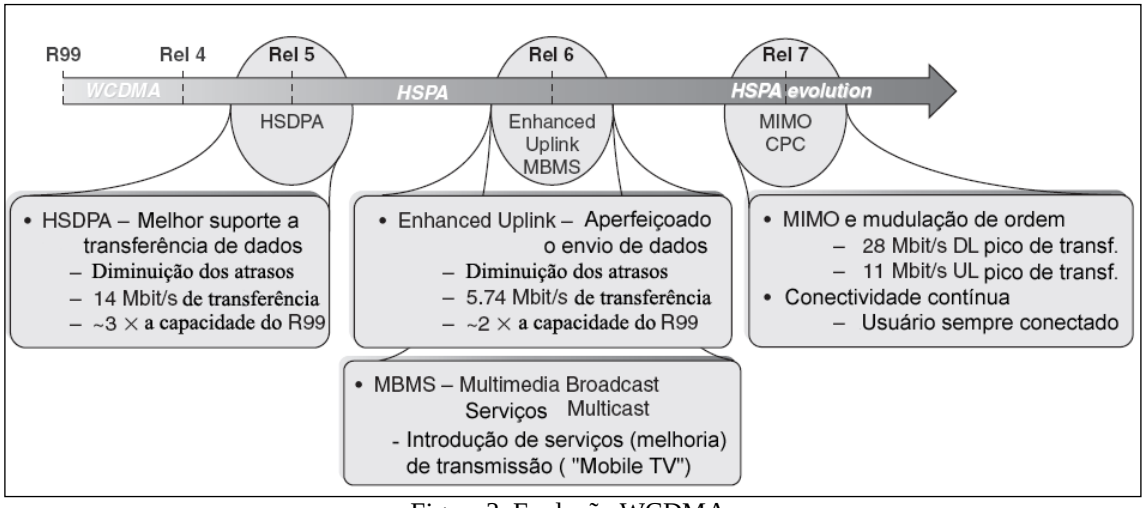

Figura 3. Evolução WCDMA
Fonte: Adaptado de DAHLMAN, E. (2007, tradução nossa, p.130).

Conforme Dahlman (2007) o WCDMA teve inicio com a Liberação 99 (*Release* 99), que leva este nome devido ao ano em que foi definido, 1999. Aprimoramentos na tecnologia foram lançados com a Liberação 4 (*Release* 4) em março de 2001. Já na Liberação

5, de março de 2002, foi lançada a tecnologia HSDPA. Em março de 2005, com a Liberação 6 surgiu o HSUPA e novos recursos como MIMO são esperadas para as próximas Liberações.

Segundo a GSM Association (2009) HSPA é um termo utilizado para englobar as siglas HSDPA e HSUPA, e também HSPA +, esta tecnologia já está presente em todo o mundo com um enorme número de dispositivos conectados e também conta com uma imensa variedade destes dispositivos.

2.2.1.1 High-Speed Downlink Packet Access (HSDPA)

É uma atualização de software da Liberação 99 da norma UMTS. Esta atualização aumenta a eficiência de recebimento de dados e diminui a latência das ligações (GSM Association, 2009, tradução nossa).

A Liberação 5 reforça a capacidade de download para a WCDMA, aumentando a capacidade de transmissão de dados, taxas de picos mais altas e diminuição da latência na comunicação. Este resultado é obtido com a modulação de ordem superior, controle de freqüência, programação dos canais independentes entre outras melhorias (DAHLMAN, 2007, tradução nossa).

2.2.1.2 High-Speed Uplink Packet Access (HSUPA)

Utiliza as mesmas técnicas do HSDPA na adaptação do link para o envio de dados, podendo criar transmissões de dados síncronas de até 5.7 Mb/s (GSM Association, 2009, tradução nossa).

Segundo Dahlman (2007) o HSUPA, também conhecido como *Enhanced Uplink*, foi adicionado ao WCDMA na sexta liberação. Oferece melhorias no envio de dados a partir

do dispositivo móvel em relação a taxas mais elevadas, latência reduzida e melhorou a capacidade do sistema. Juntos os dois (HSDPA e HSUPA), são referidos como High-Speed Packet Access (HSPA).

2.2.1.3 High-Speed Packet Access + (HSPA+)

Este sistema irá melhorar a recepção de dados para 42 Mb/s e o envio de dados para 11.5 Mb/s. Diminuindo a latência das ligações, mantendo um estado diferente para dispositivos ativos e total compatibilidade com as versões anteriores da Liberação 99 da norma UMTS, versões 5 e 6 HSPA (GSM Association, 2009, tradução nossa).

Segundo a GSM Association (2009) outro recurso para ajudar na realização do aumento de taxas de dados é a adição de antenas de múltiplo acesso de entrada e saída (*Multiple In Multiple Out* – MIMO).

2.2.1.4 Multiple Input Multiple Output (MIMO)

Segundo Dahlman (2007) MIMO foi o principal recurso adicionado na versão 7 do HSPA, tem como objetivo o uso de múltiplas antenas de recepção simultaneamente. Esta técnica disponibiliza uma maior taxa de transferência, utilizando em células pequenas, permitindo também o uso de células próximas.

2.3 SEGURANÇA NA REDE MÓVEL

A implementação de uma rede móvel pode trazer vários problemas relacionados a segurança caso não seja implementada de forma correta. Seria possível acessar informações

confidenciais, explorar fraquezas internas e implementar diversos tipos de ataques (BALDO, 2007).

Para garantir a integridade dos dados a serem transmitidos via rede celular, é indispensável que seja adotada uma técnica de criptografia. Desta forma os dados transmitidos não poderão ser capturados por ataques na rede celular.

Buscado um protocolo que atendesse a necessidade de comunicação, criptografia e compatibilidade entre a rede e os dispositivos envolvidos, o Secure Shell foi o que melhor se enquadrou nestes requisitos.

3 SECURE SHELL (SSH)

O protocolo SSH foi escolhido para a conexão cliente-servidor por que oferece segurança na troca de pacotes. Toda a conexão é criptografada e cria um tunelamento cliente-servidor, onde não é possível espionar as informações transmitidas.

Conforme Barrett (2001) a primeira versão do protocolo SSH, o SSH-1, foi desenvolvido em 1995 por Tatu Ylönen, um pesquisador da Helsinki University of Technology na Finlândia. Depois que a rede da universidade foi atacada por um espião de senhas (*Password-Sniffing*), Ylönen iniciou o desenvolvimento e os testes de um *software* que desse suporte ao protocolo SSH-1 para uso próprio.

Em julho de 1995, o SSH1 (programa que dava suporte ao protocolo SSH-1) foi liberado para o público como *software* livre, estimasse que no fim do mesmo ano, 20.000 usuários em 50 países adotaram o programa e Ylönen recebia 150 e-mails por dia solicitando suporte (BARRETT, 2001, tradução nossa).

Desta forma, pode-se constar que o protocolo SSH teve grande aceitação com os usuários das redes digitais, pela segurança que o mesmo oferecia e tinha como grande diferencial dos protocolos utilizados até então, como exemplo de protocolo temos o TELNET, que efetua conexões cliente-servidor sem nenhuma criptografia.

O Secure Shell oferece ao usuário a capacidade de conectar-se remotamente a outro computador com segurança, tendo em vista que todos os dados utilizados na rede serão criptografados, impedindo a compreensão destes por outros usuários, ao contrário do Telnet que transmite em texto puro (PETERSON, 2004).

Segundo Thomas (2007) em 1997 a *Internet Engineering Task Force* (IETF) liberou o SSH-2, segunda versão do protocolo SSH, que melhorou a segurança e a

funcionalidade do seu antecessor. A partir deste momento, o SSH-1 estava sendo lentamente defasado em favor do SSH-2.

O protocolo SSH tem como principal finalidade o *login* remoto, porém pode ser utilizado como túnel criptográfico para outros propósitos, como copiar arquivos, criptografar conexões de *e-mail* e execução de programas remotos (THOMAS, 2007).

Thomas (2007) cita outras vantagens que o SSH oferece, como o envio do *IP Spoofing* nos pacotes; criptografar os pacotes para impedir que os mesmos sejam lidos por hosts intermediários; e impede a manipulação de dados por pessoas que controlam os dispositivos ao longo da rota dos seus pacotes.

3.1 ARQUITETURA CLIENTE / SERVIDOR SECURE SHELL (SSH)

Segundo Dwivedi (2004) um servidor SSH permite que vários clientes SSH conectem-se a ele. O servidor SSH executa um serviço de escuta, normalmente na porta 22 de uma conexão TCP, estas e outras definições podem ser alteradas em um arquivo de configuração. E o Cliente SSH precisa saber o endereço IP (ou *hostname*) do servidor SSH e a porta que ele está escutando, assim o cliente só precisa se autenticar para ter acesso a sessão. Na figura 4 temos ilustrado o inicio de uma conexão SSH cliente / servidor.

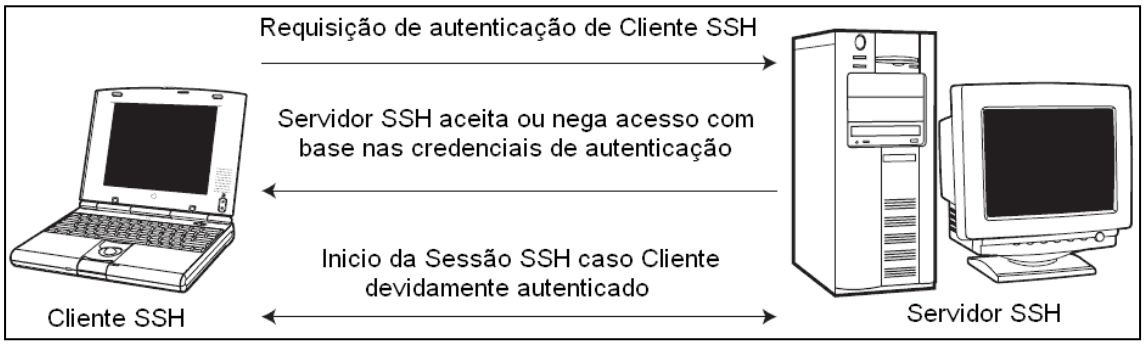

Figura 4. Comunicação do Cliente SSH para o Servidor SSH
Fonte: Adaptado de DWIVEDI, H (2004, tradução nossa, p.12).

Como mostrado na Figura 4, o processo de conexão do SSH é explicado por Dwivedi (2004) da seguinte forma:

a) o cliente SSH envia o pedido de autenticação para o servidor SSH. Nesta primeira ligação, o cliente recebe uma chave de *host* que será utilizada nas próximas conexões para verificação de servidor;
b) no segundo passo, o servidor SSH determina se o cliente será autorizado a conectar-se com o servidor, verificando o usuário e senha que o cliente forneceu na requisição;
c) se o servidor SSH autenticar o cliente e o mesmo estiver autorizado, a sessão SSH começa entre os dois *hosts*. Toda a comunicação é completamente criptografada.

Esta autenticação somente permite ao cliente ter acesso ao serviço SSH, enquanto outros serviços como *e-mail*, *intranet*, *extranet* e IRC necessitam de uma autorização adicional, segundo Dwinvedi (2004) a arquitetura SSH oferece aos clientes a possibilidade de uma única fonte de autenticação e/ou autorização, sem a necessidade de mais nomes de usuários e senhas.

3.2 DIFERENÇAS ENTRE OS PROTOCOLOS SSH-1 E SSH-2

Segundo Barret (2001) o protocolo SSH-1 é monolítico, ou seja, todas as funções estão em um único protocolo. Já o SSH-2 é dividido em módulos e é formado por três protocolos que trabalham em conjunto, o SSH Protocolo de transporte (SSH-TRANS), SSH Protocolo de Autenticação (SSH-AUTH), SSH Protocolo de conexão (SSH-CONN).

3.2.1 Protocolo de Transporte SSH (SSH-TRANS)

Segundo Ylönen (2006) o protocolo de transporte SSH foi desenvolvido para operar de modo simples e flexível para permitir parâmetros de negociação, e diminuir o número de viagens.

Este protocolo oferece um canal criptografado para uma rede insegura. Ele executa a autenticação do servidor, troca de senhas, criptografia e integridade para os dados trafegados (YLÖNEN, 2006, tradução nossa).

3.2.2 Protocolo de Autenticação SSH (SSH-AUTH)

De acordo com Ylönen (2006) este protocolo realiza a autenticação entre o cliente e o servidor SSH, o protocolo de autenticação SSH é executado sobre o protocolo de transporte SSH. A partir dele o servidor SSH envia ao cliente os métodos de autenticação que poderão ser utilizados para a conexão, assim, o cliente tem a liberdade de optar por um método que tenha suporte.

3.2.3 Protocolo de Conexão SSH (SSH-CONN)

O protocolo de conexão SSH oferece ao cliente a possibilidade de criar uma sessão de *login*, executar comandos remotamente e transmissões por uma conexão TCP/IP ou X11. Este protocolo foi desenvolvido para ser executado no topo da camada de transporte SSH e os protocolos de autenticação (YLÖNEN, 2006, tradução nossa).

3.3 SECURE SHELL EM SERVIDORES LINUX

Conforme Maximum (2001) os recursos de acesso remoto são extremamente necessários no sistema operacional Linux, principalmente se o sistema é utilizado para servidores de *Internet* ou *Intranet*. As principais maneiras de acesso remoto ao Linux são com a utilização do protocolo Telnet ou SSH, sendo o segundo altamente recomendado devido ao canal criptográfico.

As configurações Linux podem ser feitas através do terminal modo texto, permitindo que um servidor Linux possa ser atualizado e gerenciado remotamente através de uma conexão SSH (MORIMOTO, 2006).

O recurso de acesso remoto utilizando um terminal e conexão com o protocolo SSH é padrão para os sistemas Linux, ou seja, após instalado o sistema operacional em um microcomputador a possibilidade de acessá-lo remotamente só depende de um acesso a rede.

Segundo Mota Filho (2006) caso o servidor e cliente SSH não estejam instalados no Linux, os mesmos poderão ser instalados com o comando: # apt-get install ssh, e o aplicativo será instalado com as configurações padrões.

4 LINUX

O sistema operacional Linux será utilizado neste projeto por se tratar de um sistema utilizado em grande número de servidores, pois apresenta um bom desempenho em tarefas que exijam processamento, e além de ser um software livre, podendo ser utilizado, modificado e multiplicado livremente.

Segundo Kay (2002) Linus Torvalds iniciou o projeto Linux quando ainda era estudante da Universidade de Helsinki. Torvalds criou o Kernel do Linux que podia trabalhar com aplicações UNIX. E em 1991, ele lançou seu Kernel para plataforma Intel x86, que foi amplamente distribuído através da *Internet*.

Linux é livre, tal como Unix, tem código aberto, é otimizado para *Internet*, opera em sistemas de rede de 32 ou 64 bits, incluindo processadores Intel (x86) e processadores RISC (MAXIMUM, 2001, tradução nossa).

De acordo com Tabler (2000) o Linux é um sistema operacional multiusuário, permitindo que vários usuários se conectem simultaneamente na mesma máquina e utilize seus serviços. Linux é o sistema operacional mais utilizado em servidores, pelos seguintes fatores: Desempenho, segurança e custo benefício.

Do ponto de vista de um administrador de sistemas, o Linux oferece confiabilidade e segurança com a disponibilidade de vários servidores como Apache, Samba, Perl, PHP, FTP, entre outros. E do ponto de vista dos programadores, o sistema é atrativo, pois oferece recursos de *prompt* muito ricos que pode ser utilizado em conjunto com programas no modo gráfico (MORIMOTO, 2002).

5 JAVA

A plataforma Java foi criada como a idéia de desenvolver uma linguagem na qual seria escrito o código fonte uma só vez, e então o executaria em qualquer plataforma que suportasse a máquina virtual Java (MUCHOW, 2004).

A Linguagem de programação Java é uma linguagem avançada, orientada a objeto, com uma sintaxe semelhante à do C. Os criadores tentaram tornar a linguagem Java poderosa, e ao mesmo tempo tentaram evitar os recursos extremamente complexos que afundaram outras linguagens orientadas a objeto. (FLANAGAN, 2000).

Os programas Java são portáteis, desde que tenha instalado no sistema operacional que possua a máquina virtual Java (JVM) instalada. A JVM é disponibilizada pela Sun para os principais sistemas operacionais, como Solaris, Windows, sistemas Apple, Unix e também para dispositivos móveis (FLANAGAN, 2000).

A principal plataforma atualmente disponível é a *Java 2 Standard Edition* (J2SE) destinada a máquinas simples como computadores pessoais e estações de trabalho. Também é desenvolvida plataformas destinadas a Servidores e dispositivos móveis, *Java 2 Enterprise Edition* (J2EE) e *Java 2 Micro Edition* (J2ME) respectivamente (FLANAGAN, 2000).

5.1 JAVA 2 MICRO EDITION

J2ME é destinado diretamente aos dispositivos com poder limitado. Com a introdução do Java para os dispositivos móveis, temos acesso aos recursos da linguagem e da plataforma Java. Ou seja, uma linguagem de programação fácil, um ambiente que fornece uma plataforma segura e portável e acesso a conteúdo dinâmico (MUCHOW, 2004).

O J2ME consiste em uma plataforma baseada no J2SE com uma coleção de APIs reduzida, dividida em duas configurações mínimas para os dispositivos compatíveis. Estas configurações foram denominadas *Connected Device Configuration* (CDC) e *Connected Limited Device Configuration* (CLDC), o primeiro para dispositivos com uma capacidade de processamento maior e o segundo para dispositivos com capacidade de processamento e conexão limitada. Outro conceito criado é referente a arquitetura do dispositivo chamada *Mobile Information Device Profile* (MIDP) (ORACLE CORPORATION, 2010, tradução nossa).

5.1.1 Configurações

A configuração fornece um conjunto de bibliotecas básicas para o desenvolvimento de aplicativos J2ME. Segundo Muchow (2004) uma configuração define uma plataforma Java para uma ampla variedade de dispositivos. Ela está diretamente ligada a uma JVM. Uma configuração define os recursos de linguagem e as bibliotecas Java básicas da JVM para esta configuração.

5.1.1.1 Connected Device Configuration

O CDC tem o foco voltado a dispositivos móveis que possuem um maior poder de processamento e conexão a rede como *smart communicators*, PDAs e set-top boxes.

De acordo com Muchow (2004) a configuração mínima de um dispositivo para uso o CDC é de 512 kb de memória para execução do Java; 256 kb de memória para alocação durante a execução; e conectividade de rede de banda possivelmente alta.

Os dispositivos CDC geralmente possuem um microprocessador de 32 bits, cerca de 2 MB de RAM e 2,5 MB de ROM disponíveis para o ambiente de aplicações JAVA. (ORACLE CORPORATION, 2010, tradução nossa).

5.1.1.2 Connected Limited Device Configuration

O CLDC é projetado para trazer vantagens no uso em dispositivos conectados a rede com baixo poder de processamento, pouca memória e capacidade gráfica reduzida, como celulares, *pagers*, agendas eletrônicas. Os dispositivos geralmente possuem as seguintes características:

a) um processador de 16 ou 32 bits com clock de 16MHz ou superior;

b) ao menos 160 KB de memória ROM para bibliotecas CLDC da máquina virtual;

c) pelo menos 192 KB de memória RAM disponível para a plataforma Java;

d) baixo consumo de energia, geralmente alimentado por bateria;

e) conectividade com algum tipo de rede, geralmente wireless (ORACLE CORPORATION, 2010, tradução nossa).

A configuração dispositivo deve possuir no mínimo 128 kb de memória para executar o Java; 32 kb de memória para alocação em tempo de execução; baixo poder e geralmente alimentado por bateria e conectividade com a rede (MUCHOW, 2004).

5.1.2 Perfis

Um perfil é uma extensão de uma configuração. O perfil fornece as bibliotecas para o desenvolvimento de aplicativos para um determinado tipo de dispositivo. Por exemplo

o MIDP define as APIs para os componentes, entrada e tratamento de eventos de interface, armazenamento persistente, interligação com rede tudo isso levando em consideração as limitações de memória e tela do dispositivo.

5.1.2.1 Mobile Information Device Profile

Com a união de um perfil de dispositivo (MIDP) oferece então ao desenvolvedor uma plataforma completa para a criação de aplicativos para usuários de dispositivos móveis.

A primeira versão do MIDP já se encontra em desuso, oferece as funcionalidades básicas do celular, interface simples e segurança de rede. O MIDP 2.0 é a atual versão da especificação e incluem novos recursos como uma nova interface melhorada, multimídia e funções de jogo, mais recursos de conexão e segurança de comunicação (ORACLE CORPORATION, 2010, tradução nossa).

Um MIDlet é um aplicativo Java projetado para um dispositivo móvel. Uma MIDlet consistem em uma ou mais MIDlets empacotadas, usando um arquivo *Java Archive* (JAR) (MUCHOW, 2004).

6 TRABALHOS CORRELATOS

Este capítulo é dedicado aos trabalhos correlatos, abaixo temos aplicativos que usam dos mesmos recursos e oferecem ao usuário final, funções similares ao trabalho desenvolvido.

6.1 MIDPSSH

O MidpSSH é um cliente SSH e Telnet para aparelhos celulares e outros dispositivos móveis com suporte a MIDP 1.0 e 2.0, com suporte a aplicativos Java. MidpSSH é começou a ser desenvolvido por Karl Von Rondon liberado como software livre (KARL, 2009).

O aplicativo MidpSSH da suporte a conexões SSH1, SSH2 e Telnet. Utiliza de algoritmos de criptografia adaptados por Karl Von Rondow para MIDP. A interface com o usuário é feita através de uma janela terminal (console).

6.2 PUTTY

Putty é um cliente SSH desenvolvido por Simon Tatham e outros colaboradores, este, por sua vez, pode ser utilizados em *smartphones* com sistema operacional Symbian S60 e S80.

Ele dá suporte somente ao protocolo SSH e, assim como o MidpSSH, a interface com o usuário é via terminal, porém oferece a alguns aparelhos, a possibilidade de utilizar o toque na tela como entrada de dados. (SIMON, 2009, tradução nossa.)

6.3 REMUS

REMUS é um software J2ME desenvolvido por Clésio Rubens de Matos, referente ao seu trabalho de monografia de pós-graduação em Administração em Redes Linux na Universidade Federal de Lavras. Este aplicativo oferece ao administrador de rede a possibilidade de conectar-se aos seus servidores Linux e efetuar uma série de comandos pré-definidos por pesquisa de uso.

O cliente seleciona comandos *Shell Scripts,* enviados por uma conexão GPRS/EDGE com o protocolo de HTTP, para comunicação com um servidor Web TomCat executado com poderes de administrador no servidor (MATOS, 2009).

6.4 INICIALIZAÇÃO DE SERVIÇOS EM UM SERVIDOR J2EE

O Trabalho desenvolvido pelo acadêmico Darlon Ferreira Bortolini, na Universidade do Extremo Sul Catarinense, oferece ao usuário a possibilidade de iniciar serviços em um servidor a partir de um dispositivo móvel. Esta conexão utiliza a arquitetura Cliente-Servidor entre um aplicativo J2EE e um aplicativo J2ME, executados no servidor e dispositivo móvel respectivamente (BORTOLINI, 2004).

O *servlet* pode ser executado em qualquer sistema operacional, desde que tenha suporte a Java, assim o usuário tem a liberdade de iniciar processos em servidores sem se limitar a um sistema operacional.

7 GERENCIADOR DE SERVIDORES LINUX PARA CELULAR

O sistema desenvolvido consiste em um aplicativo para a plataforma Java 2 Micro Edition, que tem por objetivo o auxílio no gerenciamento de servidores Linux. Fornecendo ao usuário a capacidade de conectar a qualquer servidor Linux conectado a *Internet* e obter na tela do seu celular, por meio de uma conexão segura, os principais indicadores e funções do servidor.

7.1 METODOLOGIA

A primeira etapa deste trabalho foi desenvolvida a definição do problema e um escopo do que seria o aplicativo a ser desenvolvido para resolução deste problema. Logo após dei início ao levantamento bibliográfico, com o estudo das redes celulares, do protocolo Secure Shell (SSH), do sistema operacional Linux e da plataforma JAVA.

Nas redes celulares é dada ênfase no estudo das redes EDGE e HSPA, buscando informações sobre os recursos que essas redes disponibilizam aos usuários, para que sejam testadas e utilizadas neste software.

Já com o protocolo Secure Shell (SSH) será utilizado para oferecer um canal direto e seguro entre a aplicação e o servidor Linux, onde serão obtidos os dados utilizados pelo *software*.

Durante a terceira etapa do trabalho, foi feita a modelagem e o desenvolvimento do *software* em *Java 2 Micro Edition* (J2ME), buscando atender os objetivos definidos durante o desenvolvimento do trabalho, e por fim serão efetuados testes com a ferramenta e documentado todo o desenvolvimento.

7.1.1 Modelagem

A modelagem de dados auxilia o desenvolvedor a definir e desenvolver o seu aplicativo, de forma a esclarecer as características e recursos a serem implementados no sistema. Um bom trabalho de modelagem impede futuros problemas ao decorrer das etapas seguintes.

7.1.1.1 Requisitos

Na Tabela 1 são apresentados os requisitos funcionais para a utilização do aplicativo desenvolvido. Os requisitos funcionais são os recursos e funções que devem ser disponibilizados pelo aplicativo.

Tabela 1. Requisitos funcionais

Requisitos	Descrição	Descrição do Fluxo
RF1	Solicitar host, usuário e senha	Ao acessar o sistema é solicitado o endereço do servidor, o usuário e senha que serão utilizados para efetuar o acesso.
RF2	Iniciar conexão	Ao confirmar o acesso será iniciada uma conexão com a *Internet*.
RF3	Estabelecer conexão SSH	Após iniciada a conexão com a *Internet* o sistema estabelecerá uma conexão segura com o servidor.
RF4	Exibe processos que poderão ser finalizados	Buscar lista de processos que estão sendo executados no servidor e disponibilizar a possibilidade de finalizá-los.
RF5	Lista informações sobre os discos e memórias	Consulta dados referentes aos discos e memórias disponíveis no servidor.
RF6	Gerencia usuários do servidor	Possibilidade de listar, criar e desativar usuários
RF7	Desligar / Reiniciar servidor	Possibilidade de reiniciar ou desligar o servidor a partir do aplicativo
RF8	Execução de comando digitado	Disponibiliza um campo para digitar um comando a ser executado no servidor.

Nos requisitos não-funcionais são demonstradas as qualidades que o sistema oferecerá ao usuário. Vantagens na usabilidade, segurança, disponibilidade, desempenho entre outros. Na Tabela 2 são apresentados os requisitos não-funcionais.

Tabela 2. Requisitos não-funcionais

Requisitos	Descrição	Descrição do Fluxo
RNF1	A conexão deve ser criptografada	Segurança
RNF2	O acesso as funções deve ser fácil e rápido	Usabilidade
RNF3	O retorno deve ser claro	Usabilidade
RNF4	Pouco tráfego de rede	Operacional
RNF5	Compatível com o maior número de aparelhos	Disponibilidade
RNF6	A conexão deverá ser estabelecida com qualquer microcomputador Linux com acesso a rede	Disponibilidade

7.1.1.2 Caso de uso

O diagrama de caso de uso é uma técnica de modelagem que apresenta graficamente as funcionalidades do sistema e os usuários que estarão envolvidos no funcionamento do mesmo. Na Figura 5 pode ser analisado o diagrama de Caso de Uso desenvolvido para este aplicativo.

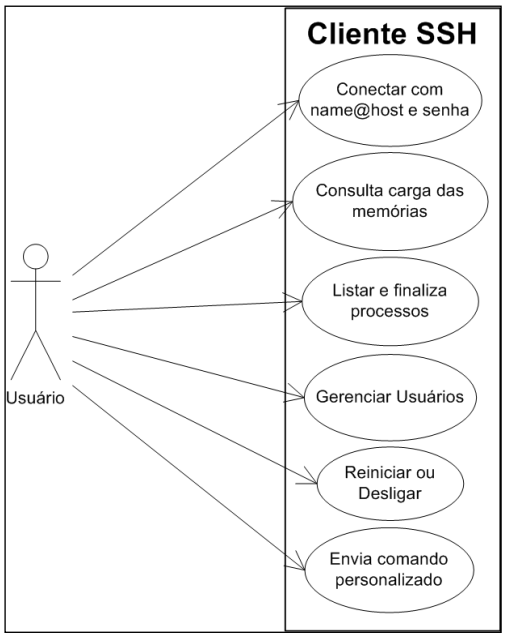

Figura 5. Caso de uso

7.1.1.3 Diagrama de Telas

O diagrama desenvolvido na Figura 6, demonstra a ligação existente entre as telas do aplicativo, como pode ser visto, após o *login* todas as funções são encontradas por meio do menu principal.

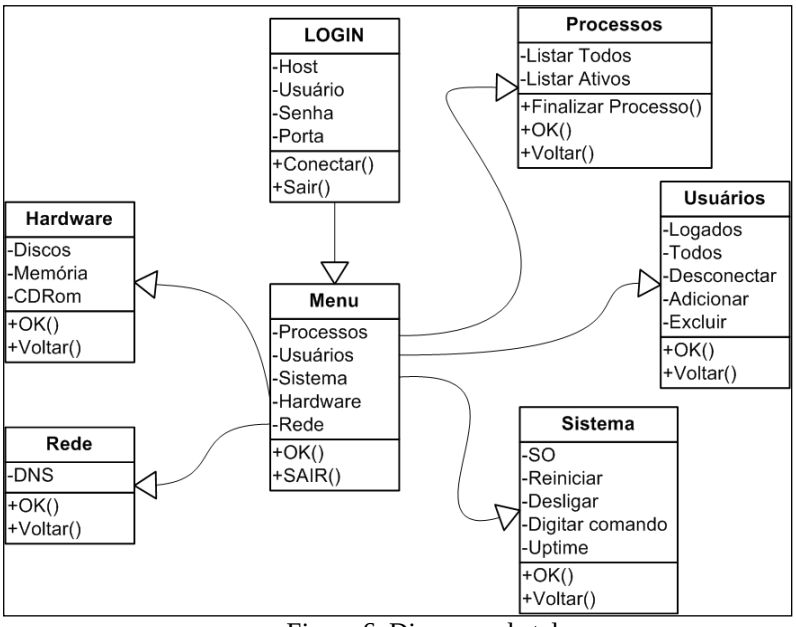

Figura 6. Diagrama de telas

7.1.2 Arquitetura do Aplicativo

O aplicativo desenvolvido neste trabalho utiliza a arquitetura Cliente / Servidor, sendo o aplicativo no aparelho celular o Cliente buscando as informações solicitadas em um microcomputador com sistema operacional Linux definido Servidor.

A conexão é iniciada pelo aplicativo Cliente, onde é definido o endereço ao qual ele deverá se conectar e os dados necessários para a identificação do usuário no Servidor (Usuário e Senha). Com a confirmação dos dados será solicitado a conexão a *Internet* ao qual o aparelho utilizará para se conectar ao servidor.

Ao utilizar uma conexão de dados de uma rede celular, uma requisição de conexão será enviada a estação base que verificará com a central de comutação de unidade móvel a disponibilidade de conexão com a *Internet* para o aparelho Cliente. Caso o cliente possua este recurso junto a operadora, será concretizada uma conexão de dados com a rede mundial.

Depois de estabelecida esta conexão, o aplicativo efetuará a conexão com o seu Servidor Linux. Com a autorização cedida ao usuário informado, o servidor envia ao cliente uma chave de identificação que será utilizada para próximas requisições, neste momento também é definida qual a técnica de criptografia que será utilizada.

Como toda a conexão é criptografada, e a técnica é definida no momento da conexão, caso seja capturado algum dado transmitido é praticamente impossível o decifrar. Assim, garantido a segurança das informações transmitidas mesmo que as redes utilizadas hospedem espiões e vulnerabilidades.

A Figura 7 apresenta o diagrama da estrutura de rede que será utilizada na conexão de dados via uma rede celular.

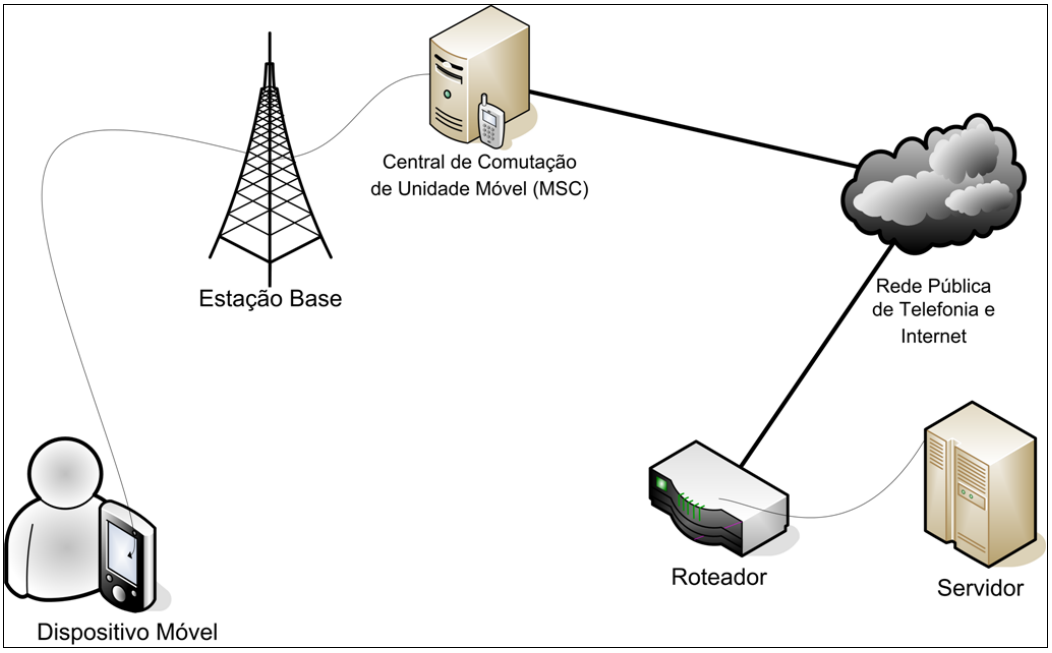

Figura 7. Estrutura da conexão Cliente / Servidor em uma rede celular

No diagrama são exibidos os componentes envolvidos nesta conexão Cliente / Servidor entre o aparelho celular e o servidor Linux. A estrutura envolve o aparelho celular onde será iniciada a conexão, a estação base responsável pela conexão do aparelho celular com a rede móvel, a Central de Comutação de Unidade Móvel (MSC) que tem o papel de identificar o cliente disponibilizá-lo o acesso a rede *Internet*.

Com a permissão de acesso, o dispositivo celular tem a possibilidade de localizar o seu servidor na *Internet*, estabelecer a conexão e dar inicio a troca de informações entre o Dispositivo Móvel e o Servidor Linux.

7.1.3 Implementação do aplicativo

O aplicativo foi desenvolvido para ser utilizado em plataforma *Java 2 Micro Edition* (J2ME), desta forma foi utilizado o ambiente de desenvolvimento *NetBeans IDE* 6.8 com suporte a programação J2ME. O ambiente *NetBens IDE* exige a instalação prévia do *Java SE Development Kit* (JDK), para o desenvolvimento foi utilizada a versão 6 atualização 19.

O aplicativo é compatível com a configuração *Configuration Limited Device Configuration* (CLDC) 1.1 e o perfil *Mobile Information Device Profile* (MIDP) 2.0, tendo em vista os recursos de implementação e conexão oferecidos e a grande gama de aparelhos compatíveis.

Para implementar a conexão com o protocolo Secure Shell foi utilizada a biblioteca de código aberto *JSch for J2ME*, um projeto destinado a dispositivos com configuração limitada.

A biblioteca *Bouncy Castle* é utilizada pela *JSch for J2ME* para a criptografia dos dados enviados na conexão SSH estabelecida pelo dispositivo móvel com o servidor. A utilização da *Bouncy Castle* pela biblioteca *JSch* trouxe complicações durante o desenvolvimento do aplicativo que foram solucionadas com o uso do *Proguard*.

O processo de desenvolvimento da aplicação foi feito em plataforma Windows Seven, utilizando Ubuntu no servidor e na máquina virtual utilizada para testes. Testes com o aplicativo foram feitos usando um emulador para aplicativos J2ME e em um aparelho celular Nokia N97 com sistema operacional Symbian S60.

7.1.3.1 JSch

O JSch é biblioteca implementada em puro Java para dar suporte ao protocolo SSH. *JSch* lhe permite conectar a um servidor usando o protocolo SSH redirecionado por uma porta do roteador, efetuar transferência de arquivos, entre outros recursos. JSch possui a licença pública *GNU*. (JCRAFT INC., 2009, tradução nossa)

A versão destinada a dispositivos móveis da biblioteca chama-se *JSch for J2ME*, existem duas bibliotecas disponíveis uma para dispositivos CDC e outra para dispositivos CLDC / MIDP 2.0. Para o desenvolvimento do aplicativo foi utilizado a segunda biblioteca. Aparentemente o projeto foi interrompido, tendo a sua última atualização no início de 2005.

7.1.3.2 Bouncy Castle

A API Bouncy Castle é desenvolvida para as linguagens C e JAVA. Bouncy Castle disponibiliza alguns algoritmos de criptografia, são eles: Rivest, Shamir and Adleman (RSA); Digital Signature Algorithm (DAS); Elliptic Curve Digital Signature Algorithm (ECDSA) e GOST (ATLASSIAN SOFTWARE SYSTEMS, 2010).

O uso desta API pela biblioteca *JSch for J2ME* trouxe atraso durante o desenvolvimento do aplicativo, pois na versão utilizada da API Bouncy Castle há classes que utilizam nomes reservados ao MIDP 2. Como pode ser observado na Figura 8 o uso das classes java.io, java.lang, java.math e java.security.

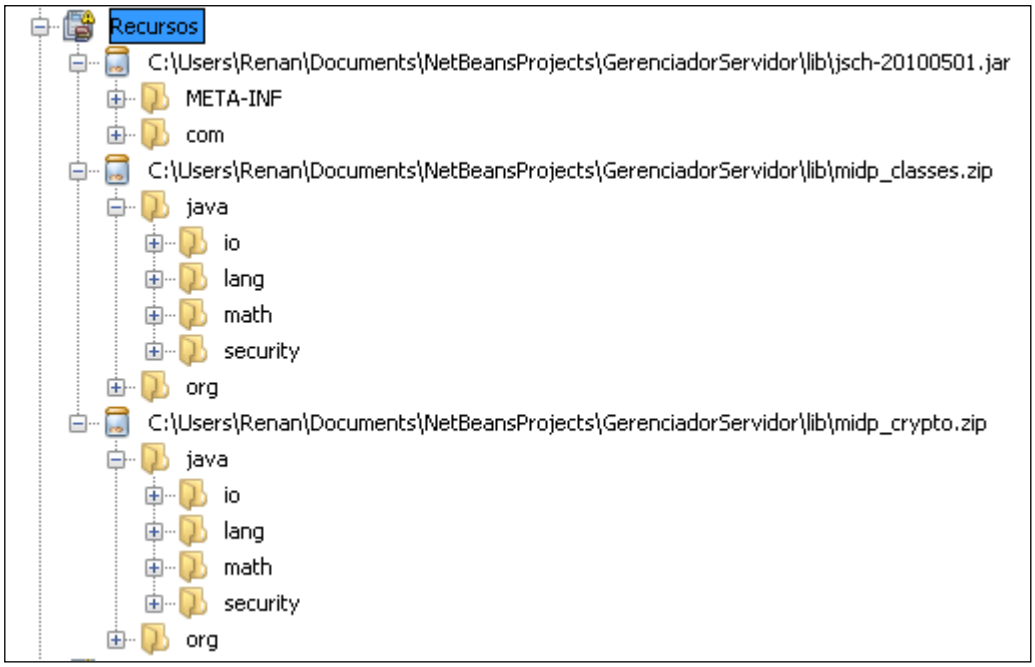

Figura 8. Bibliotecas utilizadas

Com este conflito ao tentar compilar o aplicativo, o compilador do java causou uma exceção. Para resolução deste conflito de classes foi utilizado um ofuscador de código que acompanha o NetBeans chamado *Proguard*.

7.1.3.3 Proguard

O recurso de ofuscação é executado durante a construção do aplicativo. Ao definir o nível de ofuscação, o Proguard pode renomear desde campos e métodos até bibliotecas. Na figura 9 temos a configuração definida para a construção do aplicativo desenvolvido.

Este método é bastante utilizado para alterar todos os nomes de classes, métodos e variáveis, assim ao tentar efetuar a engenharia reversa em um aplicativo já compilado, o resultado seja um código que não pode ser interpretado.

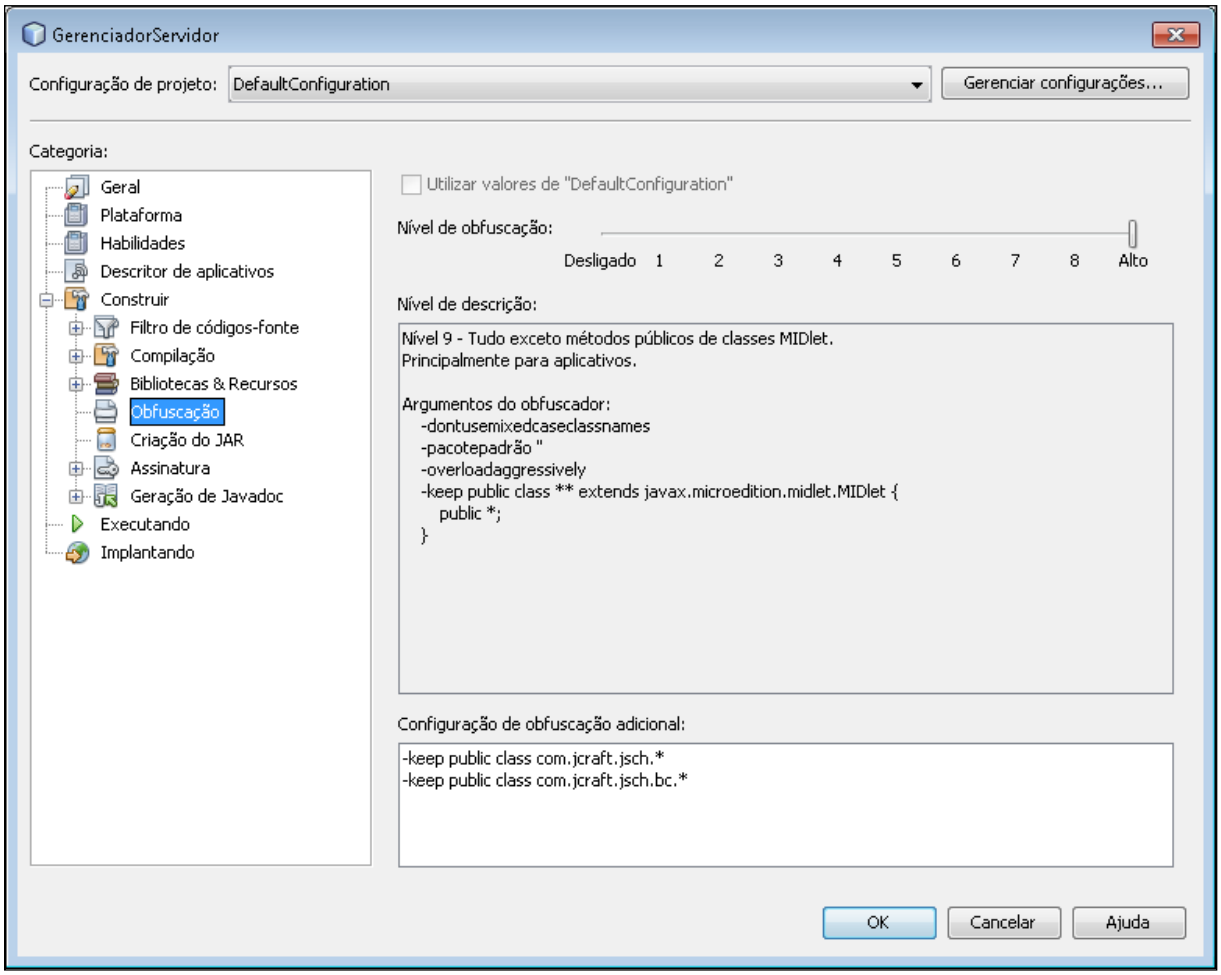

Figura 9. Configuração obfuscador Proguard

Com este nível de ofuscação foi solucionado o problema com as classes com conflito. Neste nível tudo dentro do aplicativo é ofuscado, com exceção dos métodos públicos e das classes que definimos no campo das configurações adicionais, são as classes *com.jcraft.jsch.** e *com.jcraft.jsch.bc.** , como pode ser visto na figura.

7.2 APLICATIVO DESENVOLVIDO

Com o uso da plataforma de desenvolvimento J2ME e a utilização de um código simples e genérico, sem o uso de APIs específicas a certos modelos de celulares, foi criado um aplicativo compatível com a maior parte dos dispositivos móveis disponibilizados atualmente pelo mercado.

No início do desenvolvimento do projeto, foi estipulado o uso de uma conexão de dados em uma rede telefônica, mas com a implementação do aplicativo foi possível utilizar qualquer conexão com a rede, sendo por meio de uma rede celular ou uma rede wireless.

Ao buscar um acesso a rede a Máquina Virtual Java utiliza uma conexão ativa com a *Internet* ou a conexão definida como padrão. Com isso além da conexão com a rede EDGE ou HSPA também foram testadas conexões com redes wireless, foi obtido sucesso em testes efetuados com redes wireless que não possuíam Proxy.

Ao acessar o Midlet é apresentada uma tela de *login*, onde são solicitados o host do servidor, o usuário e senha que serão utilizados no acesso e a porta que será utilizada, por padrão vem definida a porta 22. A Figura 10 mostra a tela de acesso.

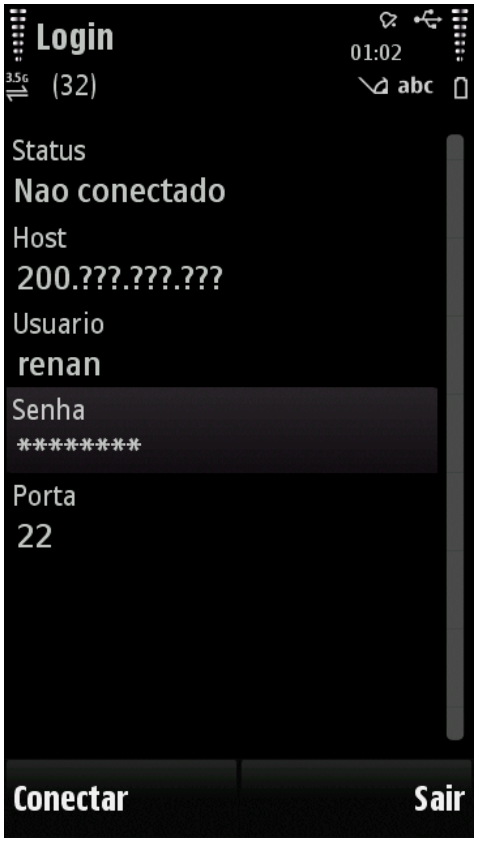

Figura 10. Tela inicial do aplicativo

Após preencher os campos com os respectivos dados e clicar em “Conectar”, será solicitado uma confirmação para que o aplicativo utilize uma conexão disponível para o envio

e recebimento de dados. Esta confirmação é padrão para os aplicativos desenvolvidos em Java.

Com a conexão ao servidor estabelecida será exibido um menu com opções referentes aos Processos, Usuários, Sistema, Hardware e rede, como pode ser visto na Figura 11.

Figura 11. Menu de acesso as opções

O menu foi organizado para o usuário ter acesso as funções com, no máximo, quatro toques na tela. As principais funções disponibilizadas são:

a) listar e finalizar processos;

b) listar, desconectar, criar ou excluir usuários;

c) informações sobre o sistema operacional;

d) reiniciar ou desligar o computador;

e) exibir informações sobre os discos;

f) exibir informações sobre a memória RAM e SWAP;

g) exibir informações sobre o DNS;

h) digitar um comando a ser executado;

A execução destas resume-se no envio de um comando pré-definido, genérico para as plataformas Linux, e o tratamento das linhas de retorno para serem exibidas na tela da melhor forma possível, gerando uma lista com informações ou com objetos.

Os comandos enviados são parametrizados para retornar os dados de maneira filtrada, buscando somente os dados mais importantes diminuindo a quantidade de dados trafegados e facilitando a exibição na tela de um dispositivo limitado. Como pode ser visualizado na Figura 12 na lista de processos são exibidas somente as colunas referentes ao usuário, PID e o processo.

Figura 12. Lista dos processos ativos

Um uma função muito interessante é a possibilidade de enviar um comando que será digitado pelo usuário. Esta opção pode ser encontrada no menu “Sistema” > “Comando Customizado”. Será exibido em tela um campo onde pode ser digitado um comando que será enviado ao servidor. Na Figura 13 é exibido um exemplo de uso, onde é enviado o comando *cat /proc/cpuinfo* comando que exibe informações sobre o processador, como fabricante, modelo e clock.

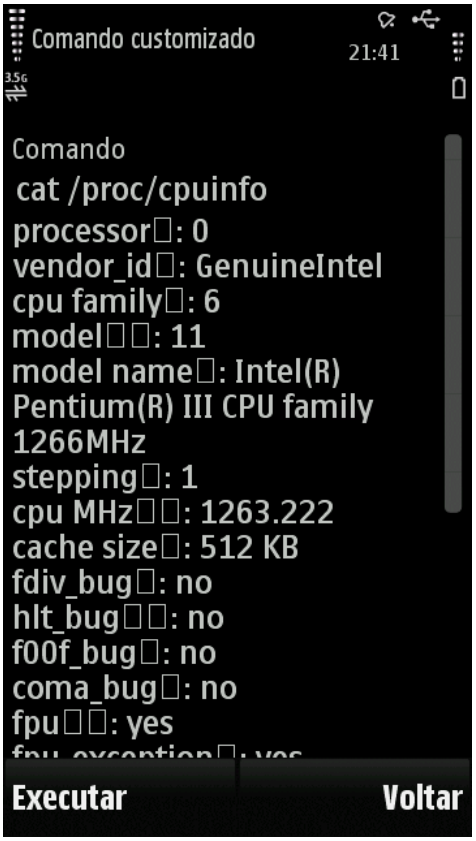

Figura 13. Retorno do comando digitado pelo usuário

O retorno deste comando não sofre nenhum filtro, ele é impresso na tela da mesma forma que sai do servidor, assim caracteres especiais podem não serem identificados pelo Java.

Com estas telas é possível demonstrar a *inteface* do aplicativo com o usuário, como pode ser visto, todas as telas são simples e de fácil interação.

Em testes realizados com a ferramenta, mesmo após a execução de vários comandos, a quantidade de dados transferidos entre o dispositivo e o servidor de conexão é mínima. Na Figura 14 pode ser verificado o *status* de uma conexão com duração de 4 minutos e o uso de 19,29 KB no total de dados transferidos.

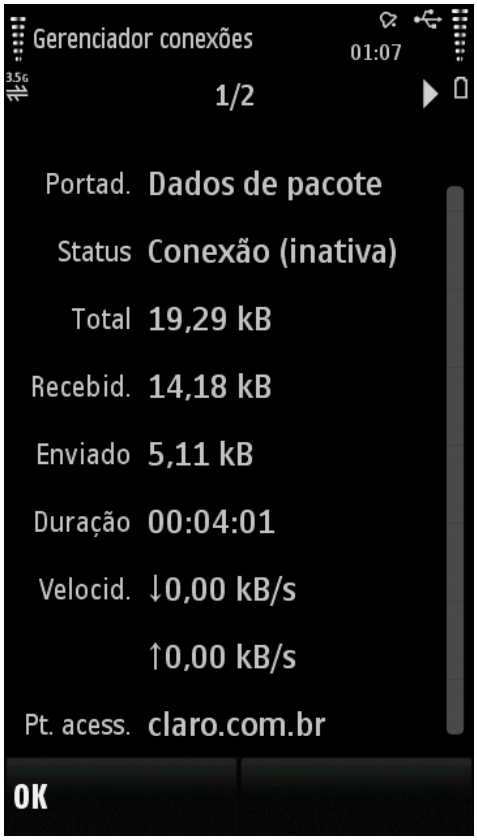

Figura 14. Dados da conexão

Devido ao baixo tráfego de rede, o uso do aplicativo não acarreta um alto custo na utilização do acesso a *Internet* a partir de uma rede celular mesmo que o usuário não possua um plano telefônico com pacote de dados. Também apresentou um bom tempo de respostas para as requisições feitas.

7.3 CONCLUSÃO

Com o desenvolvimento deste aplicativo disponibiliza-se um novo recurso aos administradores de rede, onde em poucos toques é possível efetuar alguns comandos em um servidor remoto sem nenhuma complexidade.

No desenvolvimento deste aplicativo foram utilizadas tecnologias que estão em constante atualização e popularização no mercado, que são o uso das redes celulares para a transmissão de dados e os aplicativos em J2ME desenvolvidos para dispositivos móveis, a robusteza dos servidores Linux e a segurança do protocolo SSH, que é um protocolo seguro utilizado para conexões remotas a servidores Linux.

Foi encontrada grande dificuldade no inicio do projeto na busca de bibliografia sobre o protocolo SSH e as redes celulares, principalmente as redes HSPA. Já bibliografia destinada ao desenvolvimento em J2ME e ao sistema operacional Linux foram encontradas com facilidade.

Com o estudo das redes celulares foi possível explorar uma rede que tem pouco estudo acadêmico direcionado, uma rede que sofre constantes atualizações e oferece muitos recursos a serem utilizados.

O uso das redes celulares para estabelecer uma conexão a *Internet* proporciona uma imensa mobilidade para o usuário, que poderá se conectar ao seu servidor em qualquer ponto de cobertura da rede.

O uso do protocolo SSH apresentou um ótimo desempenho, muito melhor do que o esperado. Por se tratar de um protocolo criptografado, houve o receio de demandar uma grande quantidade de dados durante as conexões, porém, como documentado, em conexões duradouras e com várias requisições ao servidor, o tráfego de rede se mostrou muito baixo e com rápida resposta. Outra grande vantagem do uso deste protocolo é a possibilidade de se

conectar a qualquer servidor Linux, sem configuração prévia, pois se trata de um protocolo padrão deste sistema operacional.

A utilização deste protocolo trás ao meio acadêmico um grande recurso a ser explorado, pois o protocolo SSH pode ser utilizado para inúmeros fins como transferência de arquivos, conexões de e-mails e execução de comandos remotos. Desta forma, muitos outros aplicativos podem ser desenvolvidos para a utilização deste recurso de comunicação e segurança.

No inicio do projeto acreditava-se que o aplicativo definia a conexão a ser utilizada para a comunicação, mas durante o desenvolvimento do aplicativo notou-se que seria possível que o aplicativo utilizasse qualquer conexão com a rede, pois a máquina virtual Java solicita ao dispositivo que seja criada uma conexão com uma rede podendo ser uma rede celular, sem fio ou outro tipo de acesso disponível no aparelho.

O uso da plataforma Java no desenvolvimento do aplicativo ofereceu um grande número de dispositivos compatíveis com a aplicação, podendo ser utilizado em centenas de modelos de aparelhos celulares.

Desta forma, propõe-se para trabalhos futuros o desenvolvimento de novos recursos para o aplicativo, como o gerenciamento de arquivos do servidor, o envio de arquivos entre cliente e servidor e um possível dinamismo dos recursos oferecidos na tela do celular, como a possibilidade de adicionar novos comandos e criar perfis com os dados referentes a conexão. Com a dependência cada vez maior dos dispositivos móveis e o crescimento do uso da rede celular para conexão de dados, tem-se neste trabalho e no seu código fonte um grande auxílio para futuros trabalhos acadêmicos que venham a utilizar os mesmos recursos.

REFERÊNCIAS

AUDREY SELIAN. **3G Mobile Licensing Policy**: From GSM to IMT-2000 - A Comparative Analysis. San Francisco, 2001. 50 p.

ATLASSIAN SOFTWARE SYSTEMS. The Legion of the Bouncy Castle. Disponível em: <http://www.bouncycastle.org>. Acesso em: 12 maio 2010.

BALDO, Jardel Pavan. MOBILE IP x HIP: UM ESTUDO SOBRE SEGURANÇA EM REDES MÓVEIS. 2007. 34 f. Monografia (Especialista) - Curso de Segurança de Redes de Computadores, Faculdade Salesiana de Vitória, Vitória, 2007.

BARRETT, Daniel J., Silverman, Richard. **SSH, The Secure Shell:** The Definitive Guide. United States of America: O'Reilly, 2001. 558 p. ISBN 0596000111.

BORTOLINI, Darlon Ferreira. **Inicialização de serviços em um servidor utilizando J2EE, por meio de um telefone celular executando J2ME.** 2004. 87 f. - Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2004.

CARMONA, Tadeu; HEXSEL, Roberto A.. **Universidade Redes**. São Paulo: Digerati Books, 2007. 226 p. ISBN 978-85-60480-31-9.

DAHLMAN, Erik et al. **3G Evolution**: HSPA and LTE for Mobile Broadband. San Diego: Elsevier, 2007. 485 p. ISBN 9780123725332.

DWIVEDI, Himanshu. **Implementing SSH:** Strategies for optimizing the Secure Shell. Indianapolis: Wiley Publishing, 2004. 376 p. ISBN 0471458805.

FLANAGAN, David. Java o guia essencial. 3. ed. Rio de Janeiro: Campus, 2000. 718 p. ISBN 8535205082.

GSM Association. EDGE. Disponível em: <http://www.gsmworld.com/technology/edge.htm>. Acesso em: 20 set. 2009.

GSM Association. GSM Coverage Maps. Disponível em: <http://www.gsmworld.com/technology/roaming/gsminfo/index.htm>. Acesso em: 20 set. 2009.

GSM Association. HSPA. Disponível em: <http://www.gsmworld.com/technology/hspa.htm>. Acesso em: 20 set. 2009.

JCRAFT INC.. JSch - Java Secure Channel. Disponível em: <http://www.jcraft.com/jsch/>. Acesso em: 18 jun. 2009.

KARL VON RONDON. MidpSSH: SSH and Telnet client for Mobile devices (MIDP/J2ME). Disponível em: <http://www.xk72.com/midpssh/>. Acesso em: 24 out. 2009.

KAY, Trevor. **Linux Certification Bible**. New York: Hungry Minds, 2002. 670 p. ISBN 0-7645-4881-6.

KUROSE, James F.; ROSS, Keith W.. **Redes de computadores e a internet:** uma abordagem top-down. 3. ed São Paulo: Pearson Addison Wesley, 2006. 634 p. ISBN 8588639181.

MATOS, Clésio Rubens de. ReMoS: Um Software para a Gerência de Servidores Linux em Dispositivos Móveis usando a Linguagem de Programação J2ME e o Protocolo GPRS/EDGE. 2009. 74 f. Monografia (Pós-graduação) - Curso de Administração em Redes Linux, Departamento de Ciência da Computação, Universidade Federal de Lavras, Lavras, 2009.

Maximum Linux Security. 2. ed. United States of America: Sams, 2001. 870 p. ISBN 0-672-32134-3.

MORIMOTO, Carlos E. **Entendendo e Dominando o Linux:** Guia Prático. São Paulo: GDH Press e Sul Editores, 2002. 766 p. Disponível em <http://www.gdhpress.com.br/etdl/#indice>. Acesso em: 7 de nov. 2009.

MORIMOTO, Carlos E. **Redes e Servidores Linux:** Guia Prático. São Paulo: GDH Press e Sul Editores, 2006. 448 p. ISBN 85-9959-306-4.

MOTA FILHO, João E.. **Descobrindo o Linux**: Entenda o sistema operacional GNU/Linux. São Paulo: Novatec, 2006. 424 p. ISBN 85-7522-090-X.

MUCHOW, John W. Core J2ME: tecnologia & MIDP. São Paulo: Pearson Educação, 2004. 588 p. ISBN 8534615225.

ORACLE CORPORATION. O que é J2ME? Disponível em: <http://www.java.com/pt_BR/download/faq/whatis_j2me.xml>. Acesso em: 06 maio 2010.

_____. Java ME Technology - CDC. Disponível em: <http://java.sun.com/javame/technology/cdc/overview.jsp>. Acesso em: 06 maio 2010.

_____. Java ME Technology - CLDC. Disponível em: <http://java.sun.com/products/cldc/overview.html>. Acesso em: 06 maio 2010.

_____. Mobile Information Device Profile (MIDP); JSR 37, JSR 118 Overview. Disponível em: <http://java.sun.com/products/midp/overview.html>. Acesso em: 06 maio 2010.

PETERSON, Larry L; Davie, Bruce S. **Redes de computadores:** uma abordagem de sistemas. 3. ed Rio de Janeiro: Elsevier, 2004. 587 p. ISBN 8535213805.

SIMON TATHAM. Putty for Symbian OS. Disponível em: <http://s2putty.sourceforge.net/>. Acesso em: 24 out. 2009.

SVERZUT, José Umberto. **Redes GSM, GPRS, EDGE e UMTS:** evolução a caminho da terceira geração (3G). 1. ed São Paulo: Érica, 2005. 454 p. ISBN 8536500875.

TABLER, Michael J.. **Inside Linux**, 2000. United States of America: New Riders, 2000. 790 p. ISBN 0-7357-0940-8.

TANENBAUM, Andrew S. **Redes de computadores**. Rio de Janeiro: Campus, 2003. 945 p. ISBN 8535211853.

THOMAS, Tom M. **Segurança em redes:** Primeiros Passos. Rio de Janeiro: Ciência Moderna Ltda, 2007. 395 p. ISBN 9788573936186.

WADLOW, Thomas A.. **Segurança de redes:** projeto e gerenciamento de redes seguras. Rio de Janeiro: Campus, 2000. 269 p. ISBN 8535206949.

YLÖNEN, Tatu. The Secure Shell (SSH) Authentication Protocol, RFC 4252, Jan 2006.

_____, Tatu. The Secure Shell (SSH) Connection Protocol, RFC 4254, Jan 2006.

_____, Tatu. The Secure Shell (SSH) Protocol Architecture, RFC 4251, Jan 2006.

_____, Tatu. The Secure Shell (SSH) Transport Layer Protocol, RFC 4253, Jan 2006.

APÊNDICE A

Gerenciamento de servidores Linux, a partir de dispositivos celulares com suporte a J2ME, utilizando protocolo de comunicação SSH

[1]Renan Rosso da Silva, [2]Prof. MSc. Rogério Antônio Casagrande

[1]Acadêmico do Curso de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC - Brasil
[2]Professor(a) do Curso de Ciência da Computação – Universidade do Extremo Sul Catarinense (UNESC) – Criciúma, SC – Brasil

renanrosso@gmail.com, roc@unesc.net

Abstract. Everyday new J2ME applications are created to ease the access of mobile devices to internet, transferring data through wi-fi or cellular networks. The main objective of the application developed in this research is to provide access to a Linux server using the SSH protocol through a mobile device supporting the J2ME technology. This application make it possible to the user to connect to any Linux server available in a network, through a secure connection, providing management services without the need of an additional application running on the server. Nowadays, with the increasing popularity of touch screen technology on mobile devices, the midlet developed in this research provide a quick way to manage one's servers remotely, with great ease.

Keywords: *J2ME, SSH, Linux, 3G,Wireless, mobile devices*

Resumo. Diariamente novos aplicativos J2ME surgem utilizando a transferência de dados via rede celular ou wireless, popularizando o acesso a *Internet* a partir dos aparelhos móveis. O aplicativo desenvolvido neste trabalho tem como principal objetivo o acesso a servidores Linux por meio de uma conexão SSH a partir de um dispositivo celular com suporte a J2ME. Este aplicativo disponibiliza ao usuário a possibilidade de conectar-se a qualquer servidor Linux ligado a uma rede, por meio de uma conexão segura, com acesso a funções de gerenciamento sem a necessidade de um segundo aplicativo instalado no servidor. Atualmente com a tecnologia *touch screen* sendo popularizado, o midlet desenvolvido proporciona o controle de seus servidores com extrema facilidade e agilidade e de forma remota.

Palavras-chave: *Dispositivos Celulares, J2ME, SSH, 3G, Wireless, Acesso Remoto, Linux*

1. Introdução

Administradores de Sistemas e de Redes devem acompanhar seu ambiente virtual corporativo para intervir perante qualquer mudança que possa prejudicar o bom funcionamento da rede e dos sistemas. A dependência deste profissional acaba trazendo

impedimentos e atrasos diante de problemas ocasionados quando os mesmos não estão disponíveis na empresa.

Uma das grandes vantagens do uso de Linux em servidores, é a facilidade do acesso remoto, tanto via linha de comando, Secure Shell (SSH), quanto com acesso à interface gráfica (VNC, FreeNX, SSH) [Morimoto 2006].

Estes profissionais utilizam largamente o recurso de acesso remoto, podendo administrar seus servidores e estações, a partir de qualquer computador, desktop ou notebook, com acesso a *Internet*. Em muitos casos, os administradores não terão acesso a estes dispositivos, caso estejam em viagem ou em zona rural, por exemplo.

Uma solução encontrada para facilitar a administração remota seria o uso dos dispositivos móveis. A partir de uma conexão a *Internet*, o mesmo poderia conectar-se a um servidor Linux e com uma interface de fácil uso agilizaria e facilitaria o controle do servidor.

Baseado nessas informações, este artigo tem por função o desenvolvimento e documentação de um sistema com certo grau de segurança para o gerenciamento de servidores Linux por meio de dispositivos celulares.

2. Redes Celulares

Buscando uma maior mobilidade para o uso do aplicativo, o uso das redes celulares nos disponibiliza uma imensa liberdade quanto ao acesso a *Internet*. Uma conexão pode ser iniciada a partir de qualquer ponto dentro da cobertura de uma rede celular.

Todas as redes de telefonia móvel, a região onde será instalada é dividida por várias células, por este motivo, os dispositivos móveis utilizados para conexão nas redes, são chamados de celulares [Tanenbaum 2003].

Estas células são geradas por uma estação-base, que serve de ponto de acesso a rede para os dispositivos móveis que estão posicionados na área que a estação-base abrange. Estas células não possuem um formato definido, o que pode gerar alguns pontos sem cobertura celular

Cada estação-base está conectada a uma central de comutação de unidade móvel (*Mobile Switching Center* – MSC), que gerencia as chamadas de e para usuários móveis. Um MSC possui o mesmo funcionamento de uma central de comutação telefônica, adicionada a capacidade de gerenciar a mobilidade de seus usuários. Esta arquitetura de rede é utilizada para as três gerações da telefonia móvel [Kurose 2006].

O método de acesso ao meio físico consiste no método de modulação utilizado para transmissão dos pacotes de comunicação de voz e dados entre o dispositivo móvel e a estação-base.

Os sistemas de telefonia celular utilizam vários métodos de acesso ao meio físico, entre eles, o *Advanced Mobile Phone Service* (AMPS), *Time Division Multiple Access* (TDMA), *Code Division Multiple Access* (CDMA) e o *Global Service for Mobile* (GSM) [Bortolini 2004].

Com a necessidade de aumento da capacidade de transmissão de dados, novas tecnologias surgiram, tendo base a rede GSM. Criaram-se então as tecnologias *General Packet Radio Services* (GPRS) e *Enhanced Data rates for GSM Evolution* (EDGE), denominadas redes de 2,5 e 2,75 gerações respectivamente [Sverzut 2005].

As redes 3G tiveram inicio com a implantação da tecnologia *Universal Mobile Telecommunication System* (UMTS), um sistema que possui total compatibilidade com o GPRS e o EDGE [Sverzut 2005].

No Brasil, a tecnologia *High-Speed Packet Access* (HSPA) está presente nos principais centros urbanos, e GSM em quase todo o território nacional povoado.

2.1. Enhanced Data rates for GSM Evolution

O GSM surgiu da necessidade da sociedade européia de criar um sistema digital que substituísse os sistemas de primeira geração, incompatíveis entre si, provendo a capacidade a um usuário móvel de ultrapassar fronteiras sem a interrupção do sinal celular [Kurose 2006].

O aprimoramento do sistema GSM é chamado de EDGE, o desenvolvimento do EDGE, inicialmente foi focado na maneira que os dados chegavam ao usuário através da introdução de taxas de modulação de maiores ordens em GSM. Durante a continuação do trabalho, o foco mudou para o reforço da tecnologia GPRS [Dahlman 2007].

A tecnologia EDGE proporciona um aumento de três vezes a capacidade de transmissão de dados em relação a tecnologia GPRS. EDGE usa a mesma estrutura da rede GSM, e tem fácil implantação, resumindo-se a uma atualização de software na rede de transmissão [GSM Association 2009].

2.2. High-Speed Packet Access

Em meados da década de 1980, a *International Telecommunication Union* (ITU) criou o padrão único de uma família de tecnologias intitulada "IMT-2000", para servir como base do sistema de terceira geração (3G) de celular [Audrey 2001].

Entre os requisitos do padrão IMT-2000 destacava-se algumas características interessantes, como atributos do usuário e tipos de serviço. Os usuários são definidos como estacionário, pedestre, veicular, veicular de alta velocidade, aeronáutico e satélite, que variam de 0 km/h a 27.000 km/h. E a categoria serviços formada por serviços de voz, áudio, texto, imagem, vídeo, sinalização e dados [Sverzut 2005].

Diversas empresas e órgãos reguladores propuseram sistemas para serem implantados, no total, 13 foram submetidos. Porém, apenas quatro foram selecionados pela ITU. Na prática temos a evolução da tecnologia UMTS (*Wideband* CDMA) que segue os padrões GSM.

Com a evolução do WCDMA temos o aperfeiçoamento da transmissão de dados, na quinta versão, temos o surgimento da tecnologia HSDPA, que traz maior desempenho na transmissão de dados *downlink*, e logo depois, na sexta versão, complementada com a *Enhanced Uplink* aumentando a capacidade de transmissão de dados do dispositivo móvel [Dahlman 2007].

2.3. Segurança na Rede Móvel

A implementação de uma rede móvel pode trazer vários problemas relacionados a segurança caso não seja implementada de forma correta. Seria possível acessar informações confidenciais, explorar fraquezas internas e implementar diversos tipos de ataques [Baldo 2007].

Para garantir a integridade dos dados a serem transmitidos via rede celular, é indispensável que seja adotada uma técnica de criptografia. Desta forma os dados transmitidos não poderão ser capturados por ataques na rede celular.

3. Secure Shell

O protocolo SSH foi escolhido para a conexão cliente-servidor por que oferece segurança na troca de pacotes. Toda a conexão é criptografada e cria um tunelamento cliente-servidor, onde não é possível espionar as informações transmitidas.

A primeira versão do protocolo SSH, o SSH-1, foi desenvolvido em 1995 por Tatu Ylönen, um pesquisador da Helsinki University of Technology na Finlândia. Depois que a rede da universidade foi atacada por um espião de senhas (*Password-Sniffing*), Ylönen iniciou o desenvolvimento e os testes de um *software* que desse suporte ao protocolo SSH-1 para uso próprio [Barrett 2001].

O Secure Shell oferece ao usuário a capacidade de conectar-se remotamente a outro computador com segurança, tendo em vista que todos os dados utilizados na rede serão criptografados, impedindo a compreensão destes por outros usuários, ao contrário do Telnet que transmite em texto puro [Peterson 2004].

A *Internet Engineering Task Force* (IETF) liberou o SSH-2, segunda versão do protocolo SSH, que melhorou a segurança e a funcionalidade do seu antecessor. A partir deste momento, o SSH-1 estava sendo lentamente defasado em favor do SSH-2.

O protocolo SSH tem como principal finalidade o *login* remoto, porém pode ser utilizado como túnel criptográfico para outros propósitos, como copiar arquivos, criptografar conexões de *e-mail* e execução de programas remotos [Thomas 2007].

3.1. Arquitetura Cliente / Servidor Secure Shell

Um servidor SSH permite que vários clientes SSH conectem-se a ele. O servidor SSH executa um serviço de escuta, normalmente na porta 22 de uma conexão TCP, estas e outras definições podem ser alteradas em um arquivo de configuração. O Cliente SSH precisa saber o endereço IP (ou *hostname*) do servidor SSH e a porta que ele está escutando, assim o cliente só precisa se autenticar para ter acesso a sessão. Na figura 4 temos ilustrado o inicio de uma conexão SSH cliente / servidor.

O processo de conexão do SSH é efetuado em três passos: primeiro o cliente SSH envia o pedido de autenticação para o servidor SSH. Nesta primeira ligação, o cliente recebe uma chave de *host* que será utilizada nas próximas conexões para verificação de servidor; no segundo passo o servidor SSH determina se o cliente será autorizado a conectar-se com o servidor, verificando o usuário e senha que o cliente forneceu na requisição; se o servidor SSH autenticar o cliente e o mesmo estiver autorizado, a sessão SSH começa entre os dois *hosts*.

3.2. Secure Shell em Servidores Linux

Os recursos de acesso remoto são extremamente necessários no sistema operacional Linux, principalmente se o sistema é utilizado para servidores de *Internet* ou *Intranet*. As principais maneiras de acesso remoto ao Linux são com a utilização do protocolo Telnet ou SSH, sendo o segundo altamente recomendado devido ao canal criptográfico [Maximum 2001].

As configurações Linux podem ser feitas através do terminal modo texto, permitindo que um servidor Linux possa ser atualizado e gerenciado remotamente através de uma conexão SSH [Morimoto 2006].

O recurso de acesso remoto utilizando um terminal e conexão com o protocolo SSH é padrão para os sistemas Linux, ou seja, após instalado o sistema operacional em um microcomputador a possibilidade de acessá-lo remotamente só depende de um acesso a rede.

4. Linux

O sistema operacional Linux será utilizado neste projeto por se tratar de um sistema utilizado em grande número de servidores, pois apresenta um bom desempenho em tarefas que exijam processamento, e além de ser um software livre, podendo ser utilizado, modificado e multiplicado livremente.

Linus Torvalds iniciou o projeto Linux quando ainda era estudante da Universidade de Helsinki. Torvalds criou o Kernel do Linux que podia trabalhar com aplicações UNIX. E em 1991, ele lançou seu Kernel para plataforma Intel x86, que foi amplamente distribuído através da *Internet* [Kay 2002].

Linux é livre, tal como Unix, tem código aberto, é otimizado para *Internet*, opera em sistemas de rede de 32 ou 64 bits, incluindo processadores Intel (x86) e processadores RISC [Maximum 2001].

O Linux é um sistema operacional multiusuário, permitindo que vários usuários se conectem simultaneamente na mesma máquina e utilize seus serviços. Linux é o sistema operacional mais utilizado em servidores, pelos seguintes fatores: Desempenho, segurança e custo benefício [Tabler 2000].

Do ponto de vista de um administrador de sistemas, o Linux oferece confiabilidade e segurança com a disponibilidade de vários servidores como Apache, Samba, Perl, PHP, FTP, entre outros. E do ponto de vista dos programadores, o sistema é atrativo, pois oferece recursos de *prompt* muito ricos que pode ser utilizado em conjunto com programas no modo gráfico [Morimoto 2002].

5. JAVA

A plataforma Java foi criada como a idéia de desenvolver uma linguagem na qual seria escrito o código fonte uma só vez, e então o executaria em qualquer plataforma que suportasse a máquina virtual Java [Muchow 2004].

A Linguagem de programação Java é uma linguagem avançada, orientada a objeto, com uma sintaxe semelhante à do C. Os criadores tentaram tornar a linguagem Java poderosa, e ao mesmo tempo tentaram evitar os recursos extremamente complexos que afundaram outras linguagens orientadas a objeto. [Flanagan 2000].

Os programas Java são portáteis, desde que tenha instalado no sistema operacional que possua a máquina virtual Java (JVM) instalada. A JVM é disponibilizada pela Sun para os principais sistemas operacionais, como Solaris, Windows, sistemas Apple, Unix e também para dispositivos móveis [Flanagan 2000].

5.1. JAVA 2 Micro Edition

J2ME é destinado diretamente aos dispositivos com poder limitado. Com a introdução do Java para os dispositivos móveis, temos acesso aos recursos da linguagem e da plataforma Java. Ou seja, uma linguagem de programação fácil, um

ambiente que fornece uma plataforma segura e portável e acesso a conteúdo dinâmico [Muchow 2004].

O J2ME consiste em uma plataforma baseada no J2SE com uma coleção de APIs reduzida, dividida em duas configurações mínimas para os dispositivos compatíveis. Estas configurações foram denominadas *Connected Device Configuration* (CDC) e *Connected Limited Device Configuration* (CLDC). Outro conceito criado é referente a arquitetura do dispositivo chamada *Mobile Information Device Profile* (MIDP) [Oracle Corporation 2010].

6. Gerenciador de Servidores Linux para Celular

O sistema desenvolvido consiste em um aplicativo para a plataforma Java 2 Micro Edition, que tem por objetivo o auxílio no gerenciamento de servidores Linux. Fornecendo ao usuário a capacidade de conectar a qualquer servidor Linux conectado a *Internet* e obter na tela do seu celular, por meio de uma conexão segura, os principais indicadores e funções do servidor.

6.1. Implementação do Aplicativo

O aplicativo foi desenvolvido para ser utilizado em plataforma *Java 2 Micro Edition* (J2ME), desta forma foi utilizado o ambiente de desenvolvimento *NetBeans IDE* 6.8 com suporte a programação J2ME. O ambiente *NetBens IDE* exige a instalação prévia do *Java SE Development Kit* (JDK), para o desenvolvimento foi utilizada a versão 6 atualização 19.

O aplicativo é compatível com a configuração *Configuration Limited Device Configuration* (CLDC) 1.1 e o perfil *Mobile Information Device Profile* (MIDP) 2.0, tendo em vista os recursos de implementação e conexão oferecidos e a grande gama de aparelhos compatíveis.

Para implementar a conexão com o protocolo Secure Shell foi utilizada a biblioteca de código aberto *JSch for J2ME*, um projeto destinado a dispositivos com configuração limitada.

A biblioteca *Bouncy Castle* é utilizada pela *JSch for J2ME* para a criptografia dos dados enviados na conexão SSH estabelecida pelo dispositivo móvel com o servidor. A utilização da *Bouncy Castle* pela biblioteca *JSch* trouxe complicações durante o desenvolvimento do aplicativo que foram solucionadas com o uso do *Proguard*.

O processo de desenvolvimento da aplicação foi feito em plataforma Windows Seven, utilizando Ubuntu no servidor e na máquina virtual utilizada para testes. Testes com o aplicativo foram feitos usando um emulador para aplicativos J2ME e em um aparelho celular Nokia N97 com sistema operacional Symbian S60.

6.2. Aplicativo Desenvolvido

Com o uso da plataforma de desenvolvimento J2ME e a utilização de um código simples e genérico, sem o uso de APIs específicas a certos modelos de celulares, foi criado um aplicativo compatível com a maior parte dos dispositivos móveis disponibilizados atualmente pelo mercado.

No início do desenvolvimento do projeto, foi estipulado o uso de uma conexão de dados em uma rede telefônica, mas com a implementação do aplicativo foi possível utilizar qualquer conexão com a rede, sendo por meio de uma rede celular ou uma rede wireless.

Ao buscar um acesso a rede a Máquina Virtual Java utiliza uma conexão ativa com a *Internet* ou a conexão definida como padrão. Com isso além da conexão com a rede EDGE ou HSPA também foram testadas conexões com redes wireless, foi obtido sucesso em testes efetuados com redes wireless que não possuíam Proxy.

Ao acessar o Midlet é apresentada uma tela de *login*, onde são solicitados o host do servidor, o usuário e senha que serão utilizados no acesso e a porta que será utilizada, por padrão vem definida a porta 22.

Após preencher os campos com os respectivos dados e clicar em "Conectar", será solicitado uma confirmação para que o aplicativo utilize uma conexão disponível para o envio e recebimento de dados. Esta confirmação é padrão para os aplicativos desenvolvidos em Java. Com a conexão ao servidor estabelecida será exibido um menu com opções referentes aos Processos, Usuários, Sistema, Hardware e rede.

O menu foi organizado para o usuário ter acesso as funções com, no máximo, quatro toques na tela. As principais funções disponibilizadas são: listar e finalizar processos; listar, desconectar, adicionar e excluir usuários; informações sobre o sistema operacional; reiniciar e desligar o servidor; exibir informações sobre os discos; exibir informações sobre a memória RAM e SWAP; exibir informações sobre o DNS e digitar um comando a ser executado.

A execução destas resume-se no envio de um comando pré-definido, genérico para as plataformas Linux, e o tratamento das linhas de retorno para serem exibidas na tela da melhor forma possível, gerando uma lista com informações ou com objetos.

Os comandos enviados são parametrizados para retornar os dados de maneira filtrada, buscando somente os dados mais importantes diminuindo a quantidade de dados trafegados e facilitando a exibição na tela de um dispositivo limitado

Em testes realizados com a ferramenta, mesmo após a execução de vários comandos, a quantidade de dados transferidos entre o dispositivo e o servidor de conexão é mínima. Ao verificar o *status* de uma conexão com duração de 4 minutos e o uso de 19,29 KB no total de dados transferidos.

Devido ao baixo tráfego de rede, o uso do aplicativo não acarreta um alto custo na utilização do acesso a *Internet* a partir de uma rede celular mesmo que o usuário não possua um plano telefônico com pacote de dados. Também apresentou um bom tempo de respostas para as requisições feitas.

7. Conclusão

Com o desenvolvimento deste aplicativo disponibiliza-se um novo recurso aos administradores de rede, onde em poucos toques é possível efetuar alguns comandos em um servidor remoto sem nenhuma complexidade.

No desenvolvimento deste aplicativo foram utilizadas tecnologias que estão em constante atualização e popularização no mercado, que são o uso das redes celulares para a transmissão de dados e os aplicativos em J2ME desenvolvidos para dispositivos móveis, a robusteza dos servidores Linux e a segurança do protocolo SSH, que é um protocolo seguro utilizado para conexões remotas a servidores Linux.

Com o estudo das redes celulares foi possível explorar uma rede que tem pouco estudo acadêmico direcionado, uma rede que sofre constantes atualizações e oferece muitos recursos a serem utilizados.

O uso das redes celulares para estabelecer uma conexão a *Internet* proporciona uma imensa mobilidade para o usuário, que poderá se conectar ao seu servidor em qualquer ponto de cobertura da rede.

O uso do protocolo SSH apresentou um ótimo desempenho, muito melhor do que o esperado. Por se tratar de um protocolo criptografado, houve o receio de demandar uma grande quantidade de dados durante as conexões, porém, como documentado, em conexões duradouras e com várias requisições ao servidor, o tráfego de rede se mostrou muito baixo e com rápida resposta. Outra grande vantagem do uso deste protocolo é a possibilidade de se conectar a qualquer servidor Linux, sem configuração prévia, pois se trata de um protocolo padrão deste sistema operacional.

No inicio do projeto acreditava-se que o aplicativo definia a conexão a ser utilizada para a comunicação, mas durante o desenvolvimento do aplicativo notou-se que seria possível que o aplicativo utilizasse qualquer conexão com a rede, pois a máquina virtual Java solicita ao dispositivo que seja criada uma conexão com uma rede podendo ser uma rede celular, sem fio ou outro tipo de acesso disponível no aparelho.

O uso da plataforma Java no desenvolvimento do aplicativo ofereceu um grande número de dispositivos compatíveis com a aplicação, podendo ser utilizado em centenas de modelos de aparelhos celulares.

Referências

Audrey Selian. **3G Mobile Licensing Policy**: From GSM to IMT-2000 - A Comparative Analysis. San Francisco, 2001. 50 p.

Baldo, Jardel Pavan. MOBILE IP x HIP: UM ESTUDO SOBRE SEGURANÇA EM REDES MÓVEIS. 2007. 34 f. Monografia (Especialista) - Curso de Segurança de Redes de Computadores, Faculdade Salesiana de Vitória, Vitória, 2007.

Bortolini, Darlon Ferreira. **Inicialização de serviços em um servidor utilizando J2EE, por meio de um telefone celular executando J2ME.** 2004. 87 f. - Curso de Ciência da Computação, Universidade do Extremo Sul Catarinense, Criciúma, 2004.

Dahlman, Erik et al. **3G Evolution**: HSPA and LTE for Mobile Broadband. San Diego: Elsevier, 2007. 485 p. ISBN 9780123725332.

Flanagan, David. Java o guia essencial. 3. ed. Rio de Janeiro: Campus, 2000. 718 p. ISBN 8535205082.

GSM Association. EDGE. Disponível em: <http://www.gsmworld.com/technology/edge.htm>. Acesso em: 20 set. 2009.

Kurose, James F.; ROSS, Keith W.. **Redes de computadores e a internet:** uma abordagem top-down. 3. ed São Paulo: Pearson Addison Wesley, 2006. 634 p. ISBN 8588639181.

Maximum Linux Security. 2. ed. United States of America: Sams, 2001. 870 p. ISBN 0-672-32134-3.

Morimoto, Carlos E. **Redes e Servidores Linux:** Guia Prático. São Paulo: GDH Press e Sul Editores, 2006. 448 p. ISBN 85-9959-306-4.

Muchow, John W. Core J2ME: tecnologia & MIDP. São Paulo: Pearson Educação, 2004. 588 p. ISBN 8534615225.

Oracle Corporation. O que é J2ME? Disponível em: <http://www.java.com/pt_BR/download/faq/whatis_j2me.xml>. Acesso em: 06 maio 2010.

Sverzut, José Umberto. **Redes GSM, GPRS, EDGE e UMTS:** evolução a caminho da terceira geração (3G). 1. ed São Paulo: Érica, 2005. 454 p. ISBN 8536500875.

Tabler, Michael J.. **Inside Linux**, 2000. United States of America: New Riders, 2000. 790 p. ISBN 0-7357-0940-8.

Tanenbaum, Andrew S. **Redes de computadores**. Rio de Janeiro: Campus, 2003. 945 p. ISBN 8535211853.