Rafael Reis Bezerra

Agenda de Tarefas Auxiliando a Administração de Servidores Linux

Monografia apresentada no Curso de Administração em Redes Linux (ARL) da Universidade Federal de Lavras como parte das exigências da disciplina Monografia para obtenção do título de Especialista em Administração em Redes Linux.

Orientador
Prof.Msc.Cristiano Leite de Castro

Lavras
Minas Gerais – Brasil
2004

Rafael Reis Bezerra

Agenda de Tarefas Auxiliando a Administração de Servidores Linux

Monografia apresentada no Curso de Administração em Redes Linux (ARL) da Universidade Federal de Lavras como parte das exigências da disciplina Monografia para obtenção do título de Especialista em Administração em Redes Linux.

Aprovada em *18 de setembro de 2004*

_____

Prof. Joaquim Quinteiro Uchôa

_____

Prof. Willian Soares Lacerda

_____

Prof.Msc.Crisiano Leite de Castro
(Orientador)

Lavras
Minas Gerais – Brasil

À todas pessoas que eu amo.
Com carinho especial à minha avó Maria Mossler
e à minha noiva Aline Ruzante

Agradecimentos

Agradeço primeiramente a Deus por me conceder o privilégio de mais uma conquista.

Agradeço à minha tia Maria Luiza por me apresentar esse curso, ao meu tio Divino Advíncula pela ajuda, à minha noiva Aline Ruzante pelo incentivo, à minha avó Maria Mossler pelo apoio e ao orientador Prof.Msc. Cristiano Castro pela paciência. Muito obrigado.

Resumo

Um administrador de Servidores *GNU/Linux* deve realizar tarefas com determinada periodicidade: checagem de quotas, atualização do sistema, atualização de dados dos usuários, etc. Para facilitar a realização dessas tarefas foi desenvolvida uma agenda que gera lembretes diários com todas as obrigações a serem realizadas no dia corrente.

Sumário

Lista de Figuras

Capítulo 1
Introdução

A tarefa de administração de servidores de rede baseia-se no cumprimento de obrigações diárias relativas à manutenção e a administração do bom funcionamento do ambiente de rede, cujo objetivo principal deve estar dirigido no sentido de fornecer um ambiente saudável e seguro.

Para que um ambiente computacional se sustente de modo preventivo ou que esteja preparado para eventuais problemas, é necessário que se estabeleça uma boa política de administração englobando todas áreas de risco do sistema.

Algumas tarefas rotineiras podem e devem ser adotadas pelo administrador de rede com o compromisso de manter os parâmetros essenciais ao bom funcionamento do ambiente, mantendo-o atualizado e em bom funcionamento. Algumas dessas tarefas devem ser realizadas eventualmente com certa periodicidade em intervalos de tempos pré-programados.

Diante dessa reflexão e contexto, este trabalho tem por objetivo apresentar um aplicativo desenvolvido para administradores de servidores, com a função principal de exibir automaticamente para o usuário informações relativas a tarefas pré-programadas, em formato de lembretes diários.

A aplicação desenvolvida possui o objetivo central de auxiliar administradores, fornecendo de modo rápido e simples o acesso a informações sobre tarefas agendadas previamente, assemelhando-se a um "*post-it"* (bloco de recados com adesivo) digital. Isso anula o uso de quadros de tarefas, como

também cadernetas escritas à mão comumente usadas em ambientes computacionais.

A agenda automática funciona basicamente em modo texto podendo ser usada com tranqüilidade em maquinas com poucos recursos de hardware. A opção de se implementar uma aplicação em modo texto foi fortemente influenciada pelo fato da fácil adaptação aos mais diversos sistemas UNIX que trabalham muito bem em ambiente não gráfico, além da pouca exigência sobre recursos físicos da máquina a ser utilizada.

Toda a implementação foi escrita em linguagem C, com algumas inserções de bash Shell para auxiliar em processos de instalação. A linguagem C foi usada devido ao seu alto grau de aplicabilidade e à extensa gama de recursos desenvolvidos para a linguagem. Outro ponto de vista abordado por este trabalho baseia-se no uso do conhecimento adquirido e estudado em matérias oferecidas pelo curso de Administração em Redes Linux (ARL) da Universidade Federal de Lavras.

Uma versão gráfica até o momento da publicação desse trabalho encontrava-se em implementação adiantada. A aplicação aqui desenvolvida está sob licença *GPL (GNU General Public License)*, detalhadamente apresentada na seção 3.4.3.

Capítulo 2
Revisão Bibliográfica

2.1 Desenvolvimento de ferramentas *GNU/Linux*.

Diversos aplicativos podem ser encontrados atualmente para *GNU/Linux* desde aplicativos de uso residencial e pessoal até aqueles utilizados por empresas de modo geral. Dentre esses aplicativos, podem ser encontradas as seguintes classes: gerenciadores de janelas, aplicativos de escritório, clientes internet, editores gráficos, aplicativos multimídia e entretenimento, gerenciadores de banco de dados, linguagens de programação, desenvolvimento de aplicações e serviços de rede entre outros, segundo [UCHOA (2003)].

O aplicativo agenda automática aqui estudado, além de ser o produto de um trabalho de conclusão de curso (especialização em *Administração de Redes Linux*) vem também acrescentar sua pequena contribuição à gama de aplicativos existentes que constituem a família *GNU/Linux*, através do tema voltado ao desenvolvimento de ferramentas para o sistema operacional *GNU/Linux*.

Ferramentas *GNU/Linux* já possuem seu lugar ao sol. Grandes empresas internacionais como Intel e Hewlett Packard, por exemplo, firmaram parceria para a criação de um centro de desenvolvimento de aplicações *GNU/Linux* na China. O principal objetivo da novidade é a criação de aplicativos corporativos. O motivo da criação de tal estabelecimento não foi justificado pelas duas empresas, porém já é uma atitude julgada pelo mercado. Muitos apostam que é mais uma forma das duas marcas mostrarem à Microsoft que não são

dependentes dos seus softwares. Já outros afirmam que as duas companhias enfim reconheceram o potencial do *GNU/Linux* e resolveram investir. Independentemente da razão, é mais uma justificativa para o Oriente apostar no pingüim, segundo [INFO (2004)].

O desenvolvimento de ferramentas em forma de software livre (veja seção 3.4.3) para o ambiente *GNU/Linux* pode vir a trazer benefícios até mesmo no âmbito social, sendo usado como meio a política de inclusão digital pelo governo federal através de adoção de software livre, segundo [SLBR (2004)].

2.2 Análise de outras ferramentas.

A agenda automática proposta tem como principal objetivo, lembrar o usuário administrador de suas tarefas periódicas. A partir desse ponto, algumas pesquisas foram realizadas a procura de softwares que possuíssem características semelhantes à aplicação em estudo.

Foram encontradas algumas ferramentas existentes cuja semelhança em relação à aplicação aqui apresentada limita-se à sua funcionalidade. Vários "*reminders*", nome pelo qual melhor se identificam esses tipos de aplicações, possuem função de relembrar seus usuários de datas especiais e comemorativas, seguindo um padrão de lembretes por data fixa, como por exemplo: o aplicativo desenvolvido em linguagem Objective C "Birthday Reminder" desenvolvido para o sistema operacional MacOS, o "Elephant Memory" e o "RTM", ambas para Windows, que exibem janelas contendo lembretes associados em datas pré-fixadas. Em relação a aplicativos *GNU/Linux* o Cloxten e o Deja-vu se destacam [SOURCE (2004)].

O Cloxten funciona basicamente como despertador e contador. Sendo capaz de executar uma variedade de eventos e ações como configurar horário para execução de arquivos e exibir tela de lembretes. O DejaVu trabalha como uma aplicação de lembretes pessoais, tendo como propósito adicionar mensagens que expiram em um determinado intervalo de tempo.

Apesar dessas aplicações possuírem o mesmo foco da ferramenta aqui estudada, elas se encontram disponíveis apenas em modo gráfico não disponibilizando versões para rodar em modo texto, além também de não apresentarem o mesmo objetivo de exibição de lembretes em intervalos cíclicos de tempo, como a agenda aqui estudada. Um exemplo de ferramenta que possuí o mesmo modo de funcionamento, levando em consideração o modo rotativo de exibição de mensagens, é a ferramenta “Simple Time”, mas essa desenvolvida apenas para o sistema operacional MacOs [SOURCE (2004)].

Todos os aplicativos aqui apresentados estão sob licença *GPL* seção 3.4.3 e podem ser encontrados em [SOURCE (2004)].

Capítulo 3

A Estrutura da Agenda Automática.

3.1 Considerações iniciais

A ferramenta apresentada nesse trabalho de conclusão consiste na automatização da exibição de mensagens pré-estabelecidas em tempo ou não de *login* de sistema, com o intuito de lembrar ao usuário, de tarefas importantes à rotina de trabalho de um administrador de sistemas.

A aplicação é dividida em dois arquivos executáveis, que trabalham em paralelo sendo um deles o responsável pela execução da agenda exibindo as mensagens diariamente e o outro com o papel de gerenciamento e configuração da aplicação de um modo geral.

3.2 Agendaconf e Agendaexec

3.2.1 Agendaconf

Tem a função de manipular e configurar a base de dados de modo interativo, guiando o usuário final através de menus desenvolvidos com funções pertinentes às tarefas escolhidas pelo administrador. O uso desse aplicativo se torna facultativo, à medida que o usuário escolher fazer a configuração da agenda através da manipulação direta dos arquivos da base de dados, editando-os.

Vantagens no uso de Agendaconf:

- O usuário não necessita conhecer o funcionamento e as regras sobre a correta manipulação da base de dados e seus respectivos parâmetros.

- A aplicação não corre o risco de encontrar a base de dados corrompida por imperícia de edição, causando danos na correta exibição das mensagens-lembrete.

- Não apresenta gastos significativos em relação a espaço em disco e perda de desempenho do hardware.

- Possui funções adicionais que permitam a visualização da base de dados completa de forma amigável, facilitando o processo de gerenciamento da aplicação.

3.2.2 Agendaexec

Tem a função de executar e exibir as mensagens em formato de lembrete ao usuário ao fazer o *login* no sistema.

A agenda varre a estrutura da base de dados e anexa o tempo real com os parâmetros encontrados (conectivos) retornando a comparação que resultar em sucesso.

O sistema trabalha sobre uma base de dados gerada a partir de informações concedidas pelo usuário final. Essa base é constituída de quatro arquivos texto, que podem ser acessados via software (menu interativos do sistema) como também podem ser editadas à mão, tornando o processo de agendamento de tarefas livre da aplicação de configuração. Arbitrariamente o uso de *agendaconf* fica a critério da vontade do usuário, podendo desta maneira ignorá-lo caso esse ache melhor assim proceder.

3.3 A base de dados

Uma agenda de tarefas se caracteriza por ser um objeto de anotação, em que se possa ter acesso e poder de manipulação a compromissos previamente estabelecidos, sem maiores complicações.

Para que anotações sejam feitas e acessadas satisfatoriamente, é necessário que se estabeleça uma base que acomode as informações de maneira organizada. Com a visão voltada para ambientes *GNU/Linux* e respeitando sua filosofia, a aplicação aqui apresentada trabalha com arquivos textos de configuração, que englobam funcionalidade e acesso independente da aplicação de gerenciamento *agendaconf*. Além disso, arquivos textos podem ser manipulados diretamente pela aplicação sem a necessidade do uso de um banco de dados, o q permite um pacote final mais enxuto e escalável.

Quatro arquivos textos são os responsáveis pelo armazenamento das mensagens criadas pelo usuário, formando a base de dados do aplicativo. São eles os arquivos, *AG_DIA.DB, AG_SEM.DB, AG_MES.DB e AG_ANO.DB*. Todos eles são gerados automaticamente pela aplicação *agendaconf* em */var/lib.*

Cada um deles segue um mesmo padrão, ou seja, constituem-se de linhas de textos com um conectivo, um delimitador e o lembrete armazenado pelo usuário salvo a exceção da base *AG_DIA.DB*, que funciona de maneira diferenciada devido à sua característica única. Abaixo são exibidas as funcionalidades e características de cada um separadamente:

3.3.1 A base AG_DIA.DB

Responsável pelo armazenamento das informações que serão exibidas diariamente.

Constitui-se de linhas de textos simples. Cada linha do arquivo contém uma única mensagem que será executada e mostrada todos os dias. Excepcionalmente não existem delimitadores e conectivos como parâmetros para esse arquivo, devido à intenção direta da exibição diária.

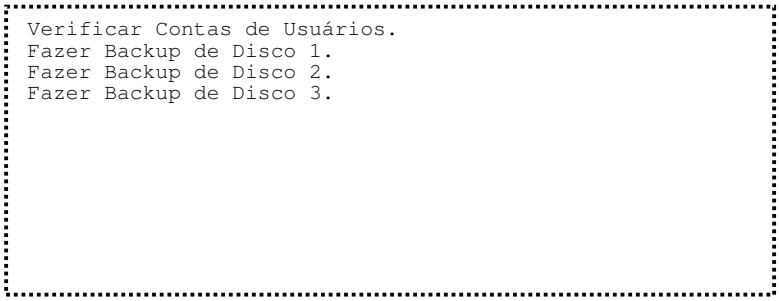

Figura 3.1: Exemplo do arquivo AG_DIA.DB

Cada linha é separada por sinais de fim de linha ”\n”.

3.3.2 A base AG_SEM.DB

Responsável pelo armazenamento das informações que serão exibidas semanalmente.

O conectivo baseia-se em um numeral que varia de zero a seis (0-6), representando, assim, os dias da semana, sendo 0 - Domingos, 1 – Segundas, 2 – Terças, 3 – Quartas, 4 – Quintas, 5 – Sextas, 6 - Sábados.

A aplicação irá varrer esse arquivo à procura dos conectivos que satisfizerem a combinação entre o dia da semana armazenado e o dia da semana corrente, devolvendo ao usuário apenas mensagens do dia da semana atual. As mensagens ali armazenadas serão exibidas periodicamente a cada sete dias, no dia da semana escolhido pelo usuário, caracterizando o período semanal entre amostragens de lembretes.

```
5:Hoje é sexta-feira, faça backup semanal.
6:Reunião com operadores.
1:Verificar logs.
1:Verificar quotas de Usuários.
3:Favor Realizar Limpeza de Disco.
```

Figura 3.2: Exemplo do arquivo AG_SEM.DB

3.3.3 A base AG_MES.DB

Responsável pelo armazenamento das informações que serão exibidas mensalmente.

O conectivo baseia-se em um numeral cardinal que varia de um a trinta e um (1-31), representando os dias do mês.

A aplicação irá varrer esse arquivo a procura dos conectivos que satisfizerem a combinação entre o dia do mês armazenado e o mês corrente, devolvendo ao usuário apenas mensagens do dia do mês atual. As mensagens aqui armazenadas serão exibidas periodicamente a cada trinta ou trinta e um dias dependendo do mês, fevereiro, pela sua excepcionalidade, irá obedecer ao total de dias levando em conta se o ano é bissexto ou não.

```
31:Reformulação e Atualização WebSite - Sinal de 2 meses.
22:Reunião Mensal com Diretoria.
10:Dia do Pagamento.
23:Verificar Atualizações de Segurança.
```

Figura 3.3: Exemplo do arquivo AG_MES.DB

3.3.4 A base AG_ANO.DB

Responsável pelo armazenamento das informações que serão exibidas anualmente.

O conectivo baseia-se em um numeral que varia de um a doze (1-12), representando os meses do ano sendo, 1 - Janeiro, 2 – Fevereiro, 3 – Março,..., 12 - Dezembro.

A aplicação irá varrer esse arquivo a procura dos conectivos que satisfizerem a combinação entre o mês armazenado e o mês corrente, devolvendo ao usuário apenas mensagens do mês atual. As mensagens aqui armazenadas serão exibidas periodicamente a cada doze meses, sendo apresentada durante todos os dias do mês em questão, caracterizando o período anual proposto entre amostragem de lembretes.

```
6:Férias de Operador 1, Remanejamento.
7:Férias de Operador 2, Remanejamento.
12:Férias do Administrador - Emergências: (34)9119-7512.
1:Providenciar hardware e fazer Update de Bastion Server.
2:Providenciar hardware e fazer Update de Router Server.
2:Vencimento de Contrato FAPESP.
```

Figura 3.4: Exemplo do arquivo AG_ANO.DB

3.4 Instalação

3.4.1 Conteúdo

A instalação dos arquivos que constituem a aplicação é compactada e distribuída através de um arquivo *tar.gz*, que copia os arquivos necessários para

a execução de *agendaexec* e *agendaconf*, como também instala arquivos de suporte adicionais. Segue abaixo descrição completa da estrutura contida no pacote que representa a aplicação.

```
# tar -xzvf auto_agenda.tar.gz
/autoagenda/agendaconf
/autoagenda/agendaexec
/autoagenda/source/agendaconf.c
/autoagenda/source/agendaexec.c
/autoagenda/source/gpl.txt
/autoagenda/source/agendaexecpam.c
/autoagenda/README
/autoagenda/pam_agendaexec.so
/autoagenda/install_ag_pam
```

Figura 3.5: Conteúdo de arquivo compactado tar.gz

3.4.2 Tutorial (README)

Um tutorial de instalação (README) é disponibilizado juntamente com os arquivos da aplicação através do *tar.gz* mostrado na seção anterior. Segue abaixo uma breve descrição das informações contidas no arquivo README anexado ao pacote final do aplicativo aqui estudado:

Instalação passo a passo referente ao arquivo README.

Como usuário *root* execute:

- **`# tar -xzvf auto_agenda.tar.gz`**

Esse procedimento irá criar o subdiretório *./autoagenda* no diretório corrente. Conteúdo Figura 3.5. Após a devida descompactação dos arquivos. Execute:

- **`# cd autoagenda`**
- **`# ./agendaconf`**

Essa será sua primeira execução de *agendaconf*, sua chance de começar a inserir registros na agenda. A primeira execução de *agendaconf* é de grande importância, pois nesse momento a aplicação cria automaticamente os arquivos que compõem a base de dados descritos detalhadamente na seção 3.3. Esse tutorial sugere que se faça algumas inserções de lembretes para dar continuação ao processo de instalação que se segue.

A aplicação já está funcionando. Para ver suas mensagens do dia corrente, execute:

- **`# ./agendaexec`**

Esse aplicativo irá executar a agenda exibindo os registros programados em *agendaconf* para o dia corrente. Até o momento, a agenda ainda não está instalada como módulo PAM de iniciação em tempo de login. Para instalar o módulo e começar a visualizar a agenda no login, execute o script em Anexo C:

- **`# ./install_ag_pam`**

Esse script é o responsável pela instalação do módulo PAM. Após esse passo o usuário poderá abrir uma nova seção <*CTRL*+*ALT*+*F2*> e testar o funcionamento do módulo em seu sistema, realizando novo login. Não

há lógica em rodar *./install_ag_pam* e fazer novo login sem antes ter executado, pelo menos uma vez, o aplicativo *agendaconf*, ele gerará a base para que se possa fazer consultas, portanto realize *agendaconf* antes de instalar a agenda como módulo.

3.4.3 GPL - General Public License

Essa aplicação está vinculada a Licença Pública Geral GNU (*GNU General Public License GPL*) [GPL (2003)] através do anexo exibido abaixo, contido no cabeçalho das aplicações aqui apresentadas.

```
/*
Agenda de Tarefas Auxiliando a Administração de Servidores Linux
Copyright (C) 2004  Rafael Reis Bezerra

This program is free software; you can redistribute it and/or
modify it under the terms of the GNU General Public License
as published by the Free Software Foundation; either version 2
of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program; if not, write to the Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA  02111-
1307, USA.
*/
```

GPL é a licença que acompanha os pacotes distribuídos pelo Projeto *GNU*, e mais uma grande variedade de software, incluindo o núcleo do sistema operacional *GNU/Linux*. A formulação da *GPL* é tal que ao invés de limitar a distribuição do software por ela protegido, ela de fato impede que este software seja integrado em software proprietário. A *GPL* é baseada na legislação internacional de copyright, o que deve garantir cobertura legal para o software licenciado com a *GPL*, segundo [HEXSEL (2003)]

Software Livre (Free Software) é o software disponível com a permissão para qualquer um usá-lo, copiá-lo, e distribuí-lo, seja na sua forma original ou com modificações, seja gratuitamente ou com custo. Em especial, a possibilidade de modificações implica em que o código fonte esteja disponível. Se um programa é livre, potencialmente ele pode ser incluído em um sistema operacional também livre. É importante não confundir software livre com software grátis porque a liberdade associada ao software livre de copiar, modificar e redistribuir, independe de gratuidade. Existem programas que podem ser obtidos gratuitamente, mas que não podem ser modificados, nem redistribuídos. Por outro lado, existe a possibilidade de uso não-gratuito em todas as categorias listadas no que segue, segundo [HEXSEL (2004)].

Para maiores informações esse trabalho sugere a leitura da definição de software livre pela *Free Software Foundation* publicada em [FSF (2004)].

Capítulo 4

Análise e funcionamento da agenda

4.1 Considerações inicias

Basicamente, a aplicação gira em torno de comparações entre medidas de tempo. A agenda se guia através da requisição do tempo do relógio local da máquina em que está sendo executada..

Diferentemente da *cron* e do comando *at GNU/Linux,* a agenda aqui apresentada possui meios instintivos usando menus de controle, para a manipulação de eventos anexados ao tempo. Sendo assim o aplicativo se diferencia na manipulação e no foco em eventos de mensagens, enquanto o *cron* caracteriza-se por ser um '*scheduler'* de arquivos executáveis, configurado através da edição de arquivo.

4.2 Função Time.h

Requisições de tempo são realizadas através do uso de funções contidas na biblioteca C time.h [C-LIBRARY (2004)]. A biblioteca time.h fornece diversas funções úteis à leitura e conversão de hora e data corrente. Segue abaixo um exemplo de requisição de tempo como sugerido pela aplicação, juntamente com o significado do código linha a linha.

1) *struct tm mytime;*
2) *time_t tempo;*

3) *tempo = time(NULL);*

4) *mytime = *localtime(&tempo);*

5) *mes = mytime.tm_mon + 1;*

6) *dia = mytime.tm_mday;*

7) *ano = mytime.tm_year + 1900;*

8) *sem = mytime.tm_wday;*

Linha 1

***struct tm*:** Estrutura usada para abrigar hora e data.

Linha 2

time_t: `Tipo de variável para armazenar calendário.`

Linha 3

tempo = time(NULL);

Adquire o calendário corrente e codifica-o dentro do formato time_t. Um valor do tipo time_t é retornado para a variável tempo do tipo time_t.

Linha 4

mytime = *localtime(&tempo);

O valor da variável time_t tempo popula a estrutura tm de nome mytime, é retornado um ponteiro para a estrutura.

Linhas 5, 6, 7, 8

mes = mytime.tm_mon + 1;

dia = mytime.tm_mday;
ano = mytime.tm_year + 1900;
sem = mytime.tm_wday;

As variáveis de estrutura usadas nessa aplicação são: (essa estrutura permite usar mais variáveis que as aqui exibidas):

int tm_mday: Dia do mês (1 a 31).
int tm_mon: Meses desde janeiro (0 a 11).
int tm_year: Anos desde 1900.
int tm_wday: Dias desde domingo (0 a 6 Domingo=0).

Algumas alterações são feitas nos valores obtidos diretamente da estrutura tm com o objetivo de ajuste de determinadas informações, haja vista que determinadas variáveis de estrutura como a *int tm_mon* e a *int tm_year* (linhas 5 e 7) retornam valores fora do padrão usado pela aplicação.

A biblioteca time.h [C-LIBRARY (2004)] e suas funções são utilizadas tanto em *agendaconf*, como em *agendaexec*.

4.3 Inserção de Lembretes.

A inserção de lembretes se dá de duas maneiras: pelo modo interativo baseado em menus com o uso do arquivo de gerenciamento *agendaconf* ou através da manipulação e edição da base de dados diretamente.

Ao se executar a aplicação *agendaconf* o menu principal (nível zero) de funções é exibido; pertencendo a ele a opção de número um (1) – Inserir

Lembrete. Essa opção é responsável pela exibição de um segundo menu (sub-menu nível um) chamado pela função interna *menuinsert()*, com quatro novas opções de inserção em função à periodicidade escolhida pelo usuário. Segue abaixo exemplo do segundo sub-menu de inserção (nível um) de lembretes.

```
INSERINDO LEMBRETE EM AGENDA

(1)-Inserir lembrete diário.
(2)-Inserir lembrete semanal.
(3)-Inserir lembrete mensal.
(4)-Inserir lembrete anual.
(5)-Voltar ao menu principal.

Escolha uma opcao:<1,2,3,4,5>:
```

Figura 4.1: Menu de inserção de lembrete

As opções de um a quatro acionam a abertura de novos sub-menus, que possuem características diferentes entre si. São eles: "(1)-Inserir lembrete diário", "(2)-Inserir lembrete semanal", "(3)-Inserir lembrete mensal" e "(4)-Inserir lembrete anual". As opções de número dois, três e quatro abrirão novos sub-menus (nível dois) cada um, esses pedirão mais detalhes sobre informações adicionais.

4.3.1 Lembretes semanais.

A inserção de lembretes semanais (opção dois) abre um novo sub-menu (nível dois) chamado pela função interna *menusemanal()*. Esse menu fornecerá opções de escolha ao usuário em relação ao dia da semana que deseja estar exibindo sua mensagem semanal. O usuário deverá escolher um numeral

correspondente ao dia da semana. Segue abaixo exemplo do sub-menu de inserção de lembretes semanais (nível dois).

```
INSERINDO LEMBRETE SEMANAL EM AGENDA

(1)-Segunda-feira.
(2)-Terça-feira.
(3)-Quarta-feira.
(4)-Quinta-feira.
(5)-Sexta-feira.
(6)-Sábado.
(7)-Domingo.
(8)-Voltar.

Escolha o dia da semana que deseja, para exibir o lembrete
semanalmente.
Por favor, escolha uma opção:<1,2,3,4,5,6,7,8>:
```

Figura 4.2: Menu de inserção de lembrete semanal

4.3.2 Lembretes mensais.

A inserção de lembretes mensais (opção três), abre um novo sub-menu (nível dois) chamado pela função interna *menumensal()*. Esse menu fornece um pequeno exemplo e informações sobre como proceder, finalizando com a opção voltada ao usuário, para que se digite um numeral correspondente ao dia do mês elegido. Segue abaixo exemplo do sub-menu de inserção de lembretes mensais (nível dois).

```
INSERINDO LEMBRETE MENSAL EM AGENDA

Escolha um dia do mês, para exibir o lembrete mensalmente.
Por exemplo:
Se digitar o número 10, o lembrete será exibido a cada dia 10 de
todos os meses do ano.

Digite o dia: Numeral de <1 a 31>:
```

Figura 4.3: Menu de inserção de lembrete mensal

4.3.3 Lembretes anuais.

A inserção de lembretes anuais (opção quatro) abre um novo sub-menu (nível dois) chamado pela função interna *menuanual()*. Esse menu fornecerá opções de escolha ao usuário em relação ao mês do ano que deseja estar exibindo sua mensagem anual. O usuário deverá escolher um numeral correspondente ao mês escolhido. Segue abaixo exemplo do sub-menu de inserção de lembretes anuais (nível dois).

```
INSERINDO LEMBRETE ANUAL EM AGENDA

(1)-Janeiro
(2)-Fevereiro.
(3)-Março.
(4)-Abril.
(5)-Maio.
(6)-Junho.
(7)-Julho.
(8)-Agosto.
(9)-Setembro.
(10)-Outubro.
(11)-Novembro.
(12)-Dezembro.
(13)-Sair

Escolha o mês do ano que deseja, para exibir o lembrete
anualmente.
Por favor, escolha uma opção:<1-13>:
```

Figura 4.4: Menu de inserção de lembrete anual.

4.3.4 Lembretes diários e a função menudig().

Todas as três opções comentadas acima convergem para uma mesma função comum, *menudig()*, responsável pela confirmação da inserção. A função *menudig()* também é responsável pela primeira opção do menu de inserção nível um. A opção de "(1)-Inserir lembrete diário", não possui conectivos nem delimitadores a serem armazenados na base de dados, então é desnecessário

possuir um sub-menu auxiliar para identificar conectivos como dia da semana, dia do mês, ou mês do ano que será exibida, pois as mensagens serão exibidas diariamente, sem depender de mais nada. Segue abaixo exemplo do sub-menu de confirmação de inserção de lembretes. Para opções dois, três e quatro (nível três) e para opção um (nível dois).

```
(1)-Digitar lembrete.
(2)-Voltar.

Escolha uma opcao:<1,2>:
```

Figura 4.5: Menu de confirmação de inserção de lembrete.

Após a confirmação de que realmente se quer digitar o lembrete, uma função chamada *insere()* toma posse do processo e exibe um *prompt* com a mensagem *"---> Entre com a mensagem a ser gravada*:", nesse momento aplicação também informa ao usuário sobre a quantidade máxima de cento e cinqüenta (150) caracteres permitidos na constituição da mensagem. Digitada a mensagem, a função se encarrega de registrar no arquivo correto da base de dados, as informações provenientes dos sub-menus que antecederam à escrita da mensagem.

Segue abaixo um esquema gráfico ilustrando o funcionamento da inserção de lembretes na base, via aplicativo *agendaconf*:

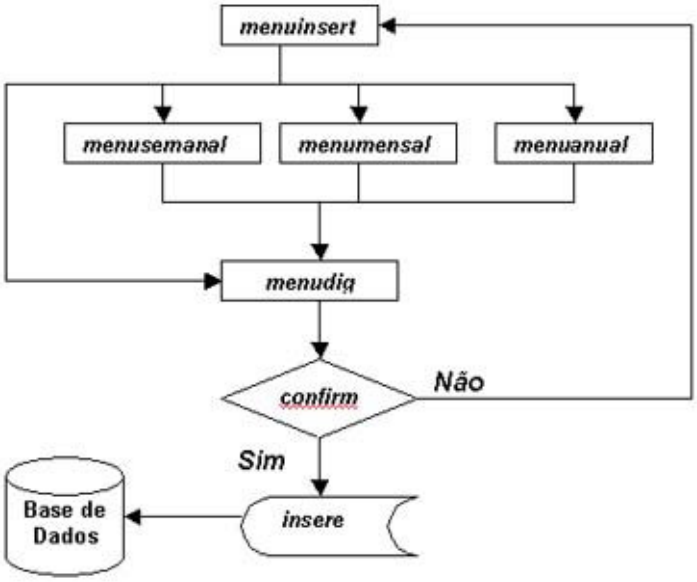

Figura 4.6: Esquema gráfico sobre fluxo de dados em *agendaconf*

4.3.5 Inserindo via edição da base.

A segunda maneira de estar realizando a inserção de lembretes na agenda, se dá através da manipulação e edição da base de dados diretamente. A agenda aceita quatro tipos de inserção, e para cada delas existe um arquivo de armazenamento que juntos constituem a base de dados completa da agenda.

A edição pode ser feita usando qualquer tipo de editor que trabalhe com texto puro, sem alterar o tipo original do texto.

A inserção de lembretes semanais, mensais e anuais seguem um mesmo padrão do tipo <*conectivo*><*delimitador*><*mensagem*>. Para adicionar:

lembrete semanal (AG_SEM.DB):

<conectivo> 0, 1, 2, 3, 4, 5 ou 6.
<delimitador> :
<mensagem> Uma mensagem textual a critério do usuário.

Exemplo: *5:Hoje é sexta-feira, faça backup semanal.*

lembrete mensal (AG_MES.DB):

<conectivo> Intervalo de 1 a 31.
<delimitador> :
<mensagem> Uma mensagem textual a critério do usuário.

Exemplo: *23:Verificar Atualizações de Segurança.*

lembrete anual (AG_ANO.DB):

<conectivo> Intervalo numérico de 1 a 12.
<delimitador> :
<mensagem> Uma mensagem textual a critério do usuário.

Exemplo: *1:Providenciar hardware e fazer Update de Bastion Server.*

A inserção de lembrete diário segue um padrão diferente do tipo *<mensagem>*, com a ausência do conectivo e delimitador. Para adicionar:

lembrete diário (AG_DIA.DB):

<mensagem> Uma mensagem textual a critério do usuário.

Cada lembrete deve estar contido em linhas diferentes entre si. Não é necessário classificar as linhas por ordem numérica de conectivos. Para que haja um bom funcionamento da aplicação é imprescindível que o usuário se certifique de qual arquivo está manipulando e que obedeça a suas regras de padrão e aos intervalos de conectivos aceitos por cada base. Após sair e salvar as respectivas alterações nos arquivos, a aplicação valida as novas mudanças realizadas na base, imediatamente.

4.4 Visualização da Base

A opção de número dois "(2)-Visualizar a base" do menu principal realiza através de uma função interna chamada *exibe()*, uma varredura nos quatro arquivos da base de dados, capturando e exibindo de forma organizada toda a informação contida nesses. Segue abaixo exemplo de uma chamada à função *exibe()* através do menu principal, exibindo os dados em formato de relatório:

```
VISUALIZANDO TODOS OS REGISTROS:

[REGISTROS DIARIOS] - 4 registros encontrados.
[1][DIARIO]:: Verificar Contas de Usuários.
[2][DIARIO]:: Fazer Backup de Disco 1.
[3][DIARIO]:: Fazer Backup de Disco 2.
[4][DIARIO]:: Fazer Backup de Disco 3.

[REGISTROS SEMANAIS] - 5 registros encontrados.
[SEMANAL][SEXTAS]:: Hoje é sexta-feira, faça backup semanal.
[SEMANAL][SABADOS]:: Reunião com operadores.
[SEMANAL][SEGUNDAS]:: Verificar logs.
[SEMANAL][SEGUNDAS]:: Verificar quotas de Usuários.
[SEMANAL][QUARTAS]:: Favor Realizar Limpeza de Disco.

[REGISTROS MENSAIS] - 4 registros encontrados.
[MENSAL][31]:: Reformulação e Atualização WebSite - Sinal de 2 meses.
[MENSAL][22]:: Reunião Mensal com Diretoria.
[MENSAL][10]:: Dia do Pagamento.
[MENSAL][23]:: Verificar Atualizações de Segurança.

[REGISTROS ANUAIS] - 6 registros encontrados.
[ANUAL][JUNHO]:: Férias de Operador 1, Remanejamento.
[ANUAL][JULHO]:: Férias de Operador 2, Remanejamento.
[ANUAL][DEZEMBRO]:: Férias do Administrador - Emergências: (34)9119-
7512.
[ANUAL][JANEIRO]:: Providenciar hardware e fazer Update de Bastion.
[ANUAL][FEVEREIRO]:: Providenciar hardware e fazer Update de Router
[ANUAL][FEVEREIRO]:: Vencimento de Contrato FAPESP.
```

Figura 4.7: Exemplo de tela emitindo relatório sobre base de dados.

É possível perceber, através do exemplo acima, como a aplicação trata as informações da base, separando-as em relação à periodicidade dos registros e exibindo-as em formato de tópicos, sendo que o número total de registros encontrados para cada tópico também é informado. Todos os conectivos encontrados são traduzidos para o seu significado semântico e separados por sinais de colchetes, tornando a leitura da base fácil e organizada.

4.5 Exclusão de Lembretes

A exclusão de lembretes pode ser feita via aplicação *agendaconf* ou por edição dos arquivos da base de dados.

4.5.1 Exclusão via *agendaconf*

A opção de número três *"(3)–Exclusão de registros"* a partir do menu principal (nível zero), é responsável pela abertura de um novo sub-menu (nível um) chamado pela função *menuexlui()*. Essa função fornece um menu com opções de exclusão de registros diários, semanais, mensais e anuais.
A exclusão é feita através da remoção de todo o conteúdo da base escolhida. Todos os dados contidos na base escolhida para remoção serão perdidos, o aplicativo fornece informação sobre o risco da operação caso o usuário venha agir de forma despretensiosa veja Figura 3.8.

O aplicativo até essa versão ainda não implementava a remoção de registros linha a linha. A exclusão se dá através da remoção do arquivo contendo a base de dados sugerida pelo usuário à exclusão. A função *menuexclui()* não chama outra função para realizar o *remove()* [UFMG (2004)], essa mesmo é responsável por todo o processo. Segue abaixo exemplo do menu de exclusão de registros (nível um).

```
EXCLUINDO REGISTROS

(1)-Excluir registros diários.
(2)-Excluir registros semanais.
(3)-Excluir registros mensais.
(4)-Excluir registros anuais.
(5)-Voltar ao menu principal.

ATENCAO: *Todos os lembretes do registro escolhido serão
perdidos!*
NAO PROSSIGA CASO NAO TENHA CERTEZA! - Escolha a opção 5 para
sair.

Escolha uma opção:<1,2,3,4,5>:
```

Figura 4.8: Menu de exclusão de registros.

4.5.2 Exclusão via edição de arquivos.

O usuário pode efetuar exclusões de registros isolados através da remoção da linha dentro do arquivo base que contenha o lembrete indesejado. Para o procedimento deve-se excluir a linha por inteiro. Por exemplo, usando o editor de textos *vi*:

- Posicionar o cursor na linha em que será removida.
- <*ESC + dd*> Teclar ESC, e depois a tecla d, duas vezes. Isso removerá a linha, ou seja, o registro não mais desejado será eliminado da agenda.

A funcionalidade de remoção de registros isolados via aplicação *agendaconf* , através de menus dedicado à remoção de registros, até o presente momento encontrava-se em implementação. A remoção completa já se encontra disponível na versão atual liberada.

É sugerido que a remoção completa seja feita apenas através do aplicativo *agendaconf*, a exclusão física do arquivo pelo usuário via linha de comando do sistema operacional, pode vir a causar problemas de execução em *agendaexec* tornando-o instável.

4.6 Funcionamento de agendaexec

É o executável responsável pela funcionalidade da agenda automática.

O seu *layout* trás informações sobre lembretes pertinentes ao dia corrente. Uma comparação entre valores numéricos de tempo é realizado entre o

relógio da máquina e os conectivos da base de dados vigente. Assim como um relógio, o aplicativo *agandaexec* funciona ciclicamente varrendo a estrutura da base à procura de combinações temporais.

As comparações são realizadas nos três arquivos de registros que possuem conectivos, são eles: *AG_SEM.DB, AG_MES.DB, AG_ANO.DB*. O conteúdo do arquivo *AG_DIA.DB* será sempre exibido que o programa for executado, pois não há restrições de tempo para essa funcionalidade, é executado diariamente. Segue abaixo o *layout* da exibição dos lembretes por *agendaexec*:

```
AGENDA AUTOMATICA - Sábado, 18 de setembro de 2004.

Lembretes registrados para hoje.

[DIARIO]
:: Verificar Contas de Usuários.

[DIARIO]
:: Fazer Backup Diário de /home.

[SEMANAL][Sabado]
:: Reunião com operadores.
```

Figura 4.9: Tela de exibição de lembretes do dia por agendaexec

Como observado no exemplo acima, apenas as mensagens respectivas ao dia corrente são exibidas. Esse aplicativo pode ser executado diretamente pelo usuário, via linha de comando, ou ao fazer login no sistema.

4.7 Iniciando no login usando PAM

4.7.1 Linux-PAM.

A aplicação *agendaexec* possui a característica de ser executada após o login de usuário. Isso é implementado a partir do uso de módulos plugáveis de autenticação ("*Linux-PAM - Pluggable Authentication Modules for Linux*") [PAM (2004)].

Linux-PAM (Pluggable Authetication Modules for Linux) é o sistema de autenticação pelo qual o *GNU/Linux* faz todas suas autenticações. Um módulo *PAM* é repassado a cada aplicação que deseja fazer autenticação através de um arquivo de configuração exclusivo, em */etc/pam.d*. Nesse diretório estão todos os arquivos de configuração de aplicações que fazem autenticação via *PAM* [BEZERRA (2003)].

4.7.2 Criando o módulo pam_agendaexec.so.

O módulo é criado a partir do arquivo de execução original *agendaexec.* Algumas alterações de sintaxe são feitas no corpo do código, para que a biblioteca *sys/pam_modules.h* o reconheça e compile da forma apropriada Abaixo segue conversão do código de *agendaexec* para módulo *PAM*.

- Inclusão da nova biblioteca:

```
#include <security/pam_modules.h>
```

- Troca do código '`main() {`' por:

```
PAM_EXTERN

int pam_sm_open_session(pam_handle_t *pamh,int flags,int
argc,const char **argv)
{
```

Após a compilação, o módulo (arquivo *pam_agendaexec.so*) deve ser copiado para */lib/security*, diretório padrão dos módulos PAM. O arquivo */etc/pam.d/login* é editado e a linha "*session optional /lib/security/ pam_agendaexec.so*" inserida no final desse arquivo. Um script de instalação (*install_ag_pam*) que acompanha o pacote do aplicativo é encarregado da realização dessas tarefas, caso o administrador deseje realizar a execução da agenda em tempo de autenticação. O script (*install_ag_pam*) deve ser executado com privilégios de super-usuário. A agenda poderá ser instalada para exibição no *login* apenas pelo usuário que possua privilégios de acesso como *root.*

4.8 Resultados

Na fase de testes a agenda automática apresentou bom funcionamento em sua funcionalidade de um modo geral, correspondendo às expectativas iniciais de exibição de lembretes diários. Todos os erros encontrados nessa fase foram devidamente sanados.

Levando em conta que todas as funcionalidades do aplicativo foram detalhadamente analisadas no cap 4, esse trabalho sugere ao leitor que acompanhe as figuras desse capítulo, como mostra de resultados diretos.

A partir do momento em que a aplicação passou a ser usada, novas idéias surgiram a respeito de possíveis melhorias e refinamentos. Algumas novas

características já se encontram em fase de implementação em uma nova versão do aplicativo. São elas:

- Adição da funcionalidade de exclusão de registros isolados, fornecendo a possibilidade do usuário poder fazer suas exclusões de lembretes via software.

- Versão gráfica GTK+ do aplicativo.

O uso da aplicação se tornou de grande utilidade, ultrapassando o domínio do uso inicialmente voltado apenas para administradores, mostrando-se também um ótimo "acionador de lembretes" de situações do cotidiano, usado por usuários comuns.

Em testes realizados no laboratório de mestrado em Ciências da Computação da Universidade Federal de Uberlândia, foi constatado, inclusive o uso da agenda automática para envio de pequenas mensagens de um dia para o outro, a outros usuários de uma mesma máquina.

Capítulo 5
Conclusão

O aplicativo aqui apresentado fornece ao usuário, meios de tornar a administração de um servidor mais organizada, permitindo que tarefas técnicas elementares ou complexas não sejam deixadas de serem realizadas. Conhecido um cronograma de tarefas, esse pode ser facilmente ajustado em forma de lembretes pela aplicação estudada, embutindo, assim, um caráter de disciplina na atuação do responsável pelo bom funcionamento da rede.

Levando em conta que as mensagens inseridas na aplicação são pessoais e de responsabilidade única do usuário, o objetivo inicial do aplicativo pode ser facilmente burlado, podendo ser usada para diversos fins, além do caráter técnico inicial, como, por exemplo, durante o uso da aplicação por diversas pessoas, lembretes administrativos foram sugeridos, assim com lembretes pessoais e recados a outros usuários que porventura usassem uma mesma máquina.

Devido a sua simplicidade em armazenar um conjunto de dados relativamente pequeno, a ferramenta satisfaz conceitos de desempenho pelo uso de arquivos textos como sua base de dados, ao invés de utilizar-se de um gerenciador de banco de dados mais robusto. A portabilidade em se trabalhar com arquivos texto como base de dados se torna também fator relevante tratando-se de sistemas *UNIX-like*.

O trabalho atual se restringe a certos parâmetros de acesso à base local e periodicidade de mensagens. Um trabalho futuro refere-se a novas funcionalidades e melhorias como:

Agendamento e execução automática de programas escolhidos pelo usuário: como o cron do Linux, mas com interface amigável, onde o usuário possa agendar tarefas de modo instintivo, guiado por menus.

Implementação da inserção de lembretes por data isolada: nesse caso, o usuário entraria com uma data qualquer para exibição de sua mensagem.

Esse tipo de funcionalidade pode ser útil na exibição de mensagens programadas para uma certa data específica, sem desejar que essa seja rotacionada e exibida de tempo em tempo. Sendo assim expirada após sua única exibição.

Criação de uma base de dados centralizada e remota, para que aplicativos clientes futuros possam lê-la via rede: nesse caso usuários possuiriam aplicativos clientes responsáveis apenas pela leitura da base sem ter acesso de escrita na mesma, enquanto o administrador ficaria responsável por programar tarefas rotineiras ou não, em uma aplicação centralizada com acesso à base central, usando a agenda para organizar cronogramas de trabalho em grupo dentro de um ambiente computacional de rede. Sendo assim o administrador teria acesso irrestrito a gerenciar os lembretes administrativos fixos ou cíclicos.

Usuários teriam acesso à base central destinada a eles através de autenticação com senha, tendo cada um acesso à sua base separadamente. Usuários comuns além do acesso à leitura aos registros da base centralizada,

possuiriam bases locais para que pudessem realizar seus próprios agendamentos. Veja abaixo esquema gráfico sobre o funcionamento da nova funcionalidade:

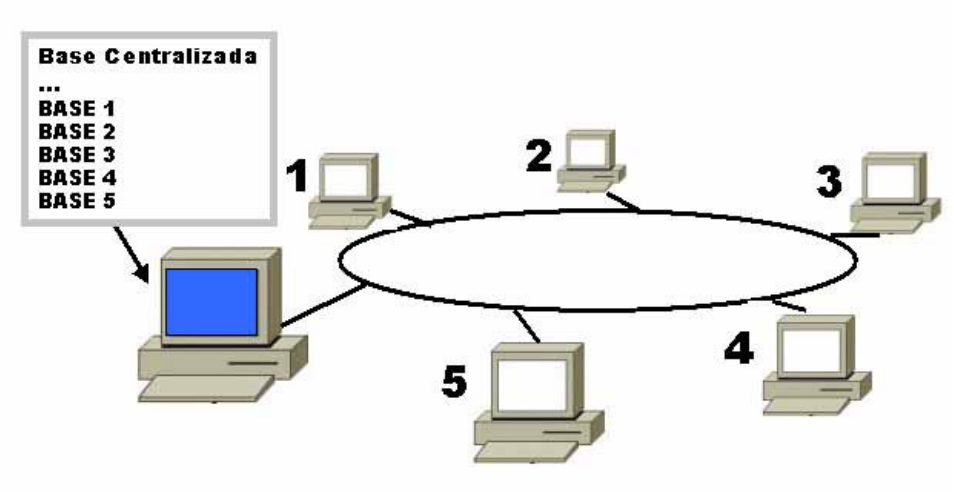

Figura 5.1: Esquema de funcionamento agenda remota.

Na Figura 5.1 é exibido um esquema onde o computador central (computador de tela azul), operado pelo administrador, possui o domínio das bases dos clientes (computadores de tela branca). A figura esquematiza computadores em redes lendo à base centralizada, gerenciada pelo administrador.

Esse trabalho está sob *GPL* [GPL (2003)], portanto o código da aplicação em anexo A e B, estão inteiramente disponíveis ao acesso do público, para redistribuição e/ou fazer modificações respeitando os termos da *GPL* como publicado por *Free Software Foundation* [FSF (2004)].

Referências Bibliográficas

[C-LIBRARY (2004)] *The C Library Reference Guide*. [on-line]. Disponível na Internet via www url: `http://www.acm.uiuc.edu/webmonkeys/book/c_guide/` Arquivo capturado em 07 de setembro de 2004.

[UFMG (2004)] *Curso de linguagem C Engenharia Elétrica - UFMG*. [on-line]. Disponível na Internet via www url: `http://ead1.eee.ufmg.br/cursos/C/c.html.` Arquivo capturado em 12 de agosto de 2004.

[PAM (2004*)] The Linux-PAM System Administrators' Guide*. [on-line]. Disponível na Internet via www url: `http://www.kernel.org/pub/linux/libs/pam/Linux-PAM-html/pam-.html.` Arquivo capturado em 9 de setembro de 2004.

[BEZERRA (2003)] *Bezerra,R. R. Biblioteca GINUX* [on-line]. Disponível na Internet via www url: `http://ginux.comp.ufla.br/documentacao/bibginux/Rafael-Pam.pdf`. Arquivo capturado em 15 de dezembro de 2003.

[CONECTIVA (2004)] *Conectiva. Arquivos Compactados*. [on-line]. Disponível na Internet via www url:

http://www.conectiva.com/doc/livros/online/10.0/usuario/pt_BR/ch08s05.html. Arquivo capturado em 3 de setembro de 2004.

[GPL (2003)] *GNU General Public License*. [on-line]. Disponível na Internet via www url: http://www.gnu.org/copyleft/gpl.html. Arquivo capturado em 8 de maio de 2003.

[HEXSEL (2004)] *Hexsel. Portal Software Livre*. [on-line]. Disponível na Internet via www url: http://www.softwarelivre.gov.br/SwLivre/. Arquivo capturado em 10 de setembro de 2004.

[SFBR (2004)] *Portal Software Livre*. [on-line]. Disponível na Internet via www url: http://www.softwarelivre.gov.br. Arquivo capturado em 11 de setembro de 2004.

[INFO (2004)] *INFO Online*. [on-line]. Disponível na Internet via www url: http://info.abril.com.br/ Arquivo capturado em 12 de setembro de 2004.

[FSF (2004)] *Fundação para Software Livre*. [on-line]. Disponível na Internet via www url: http://www.fsf.org/philosophy/free-sw.pt.html. Arquivo capturado em 10 de setembro de 2004.

[SOURCE (2004)] *Source Forge Net*. [on-line]. Disponível na Internet via www url: `http://sourceforge.net`. Arquivo capturado em 09 de setembro de 2004.

[NORTON (2000)] NORTON, Peter:Guia Completo do Linux. São Paulo, Berkeley, 2000.

[NEMETH (2002)] NEMETH, Evi:Manual de administração do sistema UNIX. Porto Alegre, Bookman, 2002.

[SCHNEIDER (2002)], Bruno de Oliveira Schneider. Linguagens de Programação II / Bruno de Oliveira Schneider. Lavras: UFLA/FAEPE, 2002. 83 p. : il. - Curso de Pós-Graduação "Lato Sensu" (Especialização) a Distância: Administração em Redes Linux.

[UCHOA (2003)], Joaquim Quinteiro Uchoa. Administração de Redes Linux / Fernando Cortês Sica, Joaquim Quinteiro Uchoa, Luiz Eduardo Simeone. Lavras: UFLA/FAEPE, 2003. 92 p. : il. - Curso de Pós-Graduação "Lato Sensu" (Especialização) a Distância: Administração em Redes Linux.

Anexo A:

Código Fonte do aplicativo agendaconf

```
/*
Agenda de Tarefas Auxiliando a Administração de Servidores
Linux - Gerenciador
Copyright (C) 2004  Rafael Reis Bezerra

This program is free software; you can redistribute it
and/or modify it under the terms of the GNU General Public
License as published by the Free Software Foundation;
either version 2 of the License, or (at your option) any
later version.

This program is distributed in the hope that it will be
useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.  See the GNU General Public License for more
details.

You should have received a copy of the GNU General Public
License along with this program; if not, write to the Free
Software Foundation, Inc., 59 Temple Place - Suite 330,
Boston, MA  02111-1307, USA.
*/

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>

int main()
{
 char op;
 struct tm mytime;
 time_t tempo;
 int  mes, dia, ano;

 tempo = time(NULL);
 mytime = *localtime(&tempo);

 mes = mytime.tm_mon + 1;
 dia = mytime.tm_mday;
 ano = mytime.tm_year + 1900;
 printf("\nPAINEL DE CONTROLE DA AGENDA \n");
```

```
 printf("\nMes Atual: %d\nDia: %d\nAno: %d\n",mes,dia,ano);
 criarqs();
 menumain();
}

menumain()
{
char op;
while(1)
     {
      printf("\n(1)-Inserir lembrete.\n");
      printf("(2)-Visualizar a base.\n");
      printf("(3)-Excluir registros.\n");
      printf("(4)-Sair.\n\n");
      printf("Escolha uma opcao:<1,2,3,4>:");
      scanf("%s",&op);

        switch(op)
             {
                 case '1':
                 menuinsert();
               break;
                 case '2':
                 printf("\nVISUALIZANDO TODOS OS
REGISTROS:...\n\n");
                 exibe();
               break;
                 case '3':
               menuexclui();
               break;
               case '4':
               printf("\nSaindo...\n");
                 exit(0);
                 break;
                 default:
                 printf("...Operador Invalido.");
                 break;
              }
     }
}

menuinsert()
{
 char op;
 while(1)
       {
           printf("\nINSERINDO LEMBRETE EM AGENDA\n");
```

```
          printf("\n(1)-Inserir lembrete diario.");
          printf("\n(2)-Inserir lembrete semanal.");
        printf("\n(3)-Inserir lembrete mensal.");
        printf("\n(4)-Inserir lembrete anual.");
        printf("\n(5)-Voltar ao menu principal.\n\n");
          printf("Escolha uma opcao:<1,2,3,4,5>:");
          scanf("%s",&op);
          switch(op)
                {
                     case '1':
                     printf("\nINSERCAO DE LEMBRETES
DIARIOS\n");
                  menudig("AG_DIA.DB",1);
                  break;
                  case '2':
                  printf("\nINSERCAO DE LEMBRETES
SEMANAIS\n");
                  menusemanal();
                  break;
                  case '3':
                  printf("\nINSERCAO DE LEMBRETES
MENSAIS\n");
                  menumensal();
                  break;
                  case '4':
                  printf("\nINSERCAO DE LEMBRETES
ANUAIS\n");
                  menuanual();
                  break;
                  case '5':
                      menumain();
                      break;
                      default:
                      printf("...Operador Invalido.");
                      break;
                }
        }
}

menuexclui()
{
char op;
while(1)
     {
         printf("\nEXCLUINDO REGISTROS\n");
         printf("\n(1)-Excluir registros diarios.");
         printf("\n(2)-Excluir registros semanais.");
```

```
         printf("\n(3)-Excluir registros mensais.");
         printf("\n(4)-Excluir registros anuais.");
         printf("\n(5)-Voltar ao menu principal.\n\n");
       printf("ATENCAO: *Todos os lembretes do registro
escolhido serao perdidos!*\n");
       printf("NAO PROSSIGA CASO NAO TENHA CERTEZA! -
Escolha a opcao 5 para sair.\n\n");
       printf("Escolha uma opcao:<1,2,3,4,5>:");
         scanf("%s",&op);
       criarqs();
         switch(op)
              {
                case '1':
            remove("AG_DIA.DB");
            printf("\nOK...\nTodo o conteúdo de registros
diarios foram apagados.\n");
                break;
                case '2':
            remove("AG_SEM.DB");
            printf("\nOK...\nTodo o conteúdo de registros
semanais foram apagados.\n");
                break;
                case '3':
                remove("AG_MES.DB");
                printf("\nOK...\nTodo o conteúdo de
registros mensais foram apagados.\n");
                break;
                case '4':
                remove("AG_ANO.DB");
                printf("\nOK...\nTodo o conteúdo de
registros anuais foram apagados.\n");
                break;
                case '5':
                menumain();
                break;
                default:
                printf("...Operador Invalido.");
                break;
              }
     }
}


menusemanal()
{
 int op;
 while(1)
```

```
 {
      printf("\nINSERINDO LEMBRETE SEMANAL EM AGENDA\n");
        printf("\n(1)-Segunda-feira.");
        printf("\n(2)-Terca-feira.");
        printf("\n(3)-Quarta-feira.");
        printf("\n(4)-Quinta-feira.");
        printf("\n(5)-Sexta-feira.");
      printf("\n(6)-Sabado.");
      printf("\n(7)-Domingo.");
      printf("\n(8)-Voltar.\n\n");
      printf("Escolha o dia da semana que deseja, para
exibir o lembrete semanalmente.\n");
      printf("Por favor, escolha uma
opcao:<1,2,3,4,5,6,7,8>:");
      scanf("%d",&op);
      if(op==8)menuinsert();
      while(op > 8 || op < 1){
              printf("\n\nDesculpe, voce digitou um numero
invalido\nPor favor, tente outra vez\n\n");
              printf("Por favor, escolha uma opcao:
<1,2,3,4,5,6,7,8>:");
              scanf("%d",&op);
      }
      if(op==7)op=0;
      menudig("AG_SEM.DB",op);
 }
}

menumensal()
{
 int op;
 while(1)
  {
        printf("\nINSERINDO LEMBRETE MENSAL EM AGENDA\n");
        printf("\nEscolha um dia do mes, para exibir o
lembrete mensalmente.\n");
      printf("Por exemplo:\n\n");
      printf("Se digitar o numero 10, o lembrete sera
exibido a cada dia 10 de todos os meses do ano\n\n");
        printf("Digite o dia: Numeral de <1 a 31>:");
        scanf("%d",&op);
        while(op > 31 || op < 1){
            printf("\n\nDesculpe, voce digitou um numero
invalido\nPor favor, tente outra vez\n\n");
            printf("Por favor, digite o dia: Numeral de <1
a 31>:");
            scanf("%d",&op);
```

```
      }
      menudig("AG_MES.DB",op);
  }
}


menuanual()
{
  int op;
  while(1)
   {
        printf("\nINSERINDO LEMBRETE ANUAL EM AGENDA\n");
        printf("\n(1)-Janeiro.");
        printf("\n(2)-Fevereiro.");
        printf("\n(3)-Marco.");
        printf("\n(4)-Abril.");
        printf("\n(5)-Maio.");
        printf("\n(6)-Junho.");
        printf("\n(7)-Julho.");
      printf("\n(8)-Agosto.");
      printf("\n(9)-Setembro.");
      printf("\n(10)-Outubro.");
      printf("\n(11)-Novembro.");
      printf("\n(12)-Dezembro.");
        printf("\n(13)-Voltar.\n\n");
        printf("Escolha o mes do ano que deseja, para
exibir o lembrete anualmente.\n");
        printf("Por favor, escolha uma opcao:<1-13>:");
        scanf("%d",&op);
      if(op==13)menuinsert();
      while(op > 13 || op < 1){
              printf("\n\nDesculpe, voce digitou um numero
invalido\nPor favor, tente outra vez\n\n");
              printf("Por favor, Escolha uma opcao: <1 a
12>:");
              scanf("%d",&op);
      }
      menudig("AG_ANO.DB",op);
  }
}

menudig(char fileAGDB[10],int conectempo)
{
 char op;
 while(1)
     {
        printf("\n(1)-Digitar lembrete.\n");
```

```
          printf("(2)-Voltar.\n\n");
          printf("Escolha uma opcao:<1,2>:");
        scanf("%s",&op);

        switch(op)
                  {
                      case '1':
                      insere(fileAGDB,conectempo);
                  printf("\nOK.. mensagem gravada..\n");
                      break;
                      case '2':
                      menuinsert();
                      break;
                      default:
                      printf("...Operador Invalido.<Por
favor, tente uma opcao valida>");
                      break;
                  }
     }
}

insere(char arq[10],int conectempo)
{
    FILE *p;
    char str[150],strconect[30];
    printf("\n\nA mensagem devera ter no maximo 150
caracteres.");
    printf("\nEntre com a mensagem a ser gravada:\n--> ");
    fgets(str,149,stdin);
    fgets(str,149,stdin);
    if (!(p = fopen(arq,"a")))
    {
         printf("Erro! Impossivel abrir o arquivo!\n");
         exit(1);
    }
    sprintf(strconect, "%d", conectempo);
    strcat(strconect,":");
    if(strcmp("AG_SEM.DB",arq)==0)fputs(strconect,p);
    if(strcmp("AG_MES.DB",arq)==0)fputs(strconect,p);
    if(strcmp("AG_ANO.DB",arq)==0)fputs(strconect,p);
    fputs(str,p);
    fclose(p);
}

exibe()
{
 FILE *p;
```

```
 int i,j=0,cont=0,ic2;
 struct tm mytime;
 time_t tempo;
 int  mes, dia, ano, sem;
 char
carac,caracsem[10],caracmes[15],semstr[15],messtr[15],conte
udo[150],carac2[2];
 tempo = time(NULL);
 mytime = *localtime(&tempo);

 sem = mytime.tm_wday;

 static char arq[4][10]={
 "AG_DIA.DB",
 "AG_SEM.DB",
 "AG_MES.DB",
 "AG_ANO.DB"};

 while(cont<4) {
       if((p = fopen(arq[cont], "r")) == NULL)  {
           printf("Erro na abertura do arquivo");
           exit(1);
        }

       while(!feof(p)){
             fgets(conteudo,149,p);
             j++;
       }
       j--;
       if(cont==0)printf("[REGISTROS DIARIOS] ");
       if(cont==1)printf("[REGISTROS SEMANAIS] ");
       if(cont==2)printf("[REGISTROS MENSAIS] ");
       if(cont==3)printf("[REGISTROS ANUAIS] ");
       if(j==0)printf("- Nenhum registro encontrado.\n");
       else  if(j==1)printf("- %d registro
encontrado.\n",j);
             else printf("- %d registros
encontrados.\n",j);
       fclose(p);
       if((p = fopen(arq[cont], "r")) == NULL)  {
           printf("Erro na abertura do arquivo");
           exit(1);
       }
       if(cont!=1 && cont!=2 && cont!=3){
       for(i=0;i<j;i++){
              fgets(conteudo,149,p);
```

```
             if(cont==0)printf("[%d][DIARIO]::
%s",i+1,conteudo);
        }
       }
       if(cont==1){
             sprintf(caracsem,"%d",sem);
             while((carac=getc(p)) != EOF)
                 {
                     if(carac ==
'0')printf("[SEMANAL][DOMINGOS]:: ");
                     if(carac ==
'1')printf("[SEMANAL][SEGUNDAS]:: ");
                     if(carac ==
'2')printf("[SEMANAL][TERCAS]:: ");
                     if(carac ==
'3')printf("[SEMANAL][QUARTAS]:: ");
                     if(carac ==
'4')printf("[SEMANAL][QUINTAS]:: ");
                     if(carac ==
'5')printf("[SEMANAL][SEXTAS]:: ");
                     if(carac ==
'6')printf("[SEMANAL][SABADOS]:: ");
                       while((carac=getc(p)) != '\n'){
                            if(carac !=
':')printf("%c",carac);
                          }
                  if(carac == '\n')printf("\n");
             }
        }

       if(cont==2){
            ic2=0;
            while((carac2[ic2]=getc(p)) != EOF){
                    if((carac=getc(p)) == ':'){
                            printf("[MENSAL][%c]::
",carac2[ic2]);
                            while((carac=getc(p)) !=
'\n')printf("%c",carac);
                            printf("\n");
                            ic2=-1;
                     }
                    else{
                            carac2[ic2+1]=carac;
                            printf("[MENSAL][%c%c]::
",carac2[ic2],carac2[ic2+1]);
                            while((carac=getc(p)) != '\n'){
```

```
                                    if(carac !=
':')printf("%c",carac);
                          }
                              printf("\n");
                              ic2=-1;
                           }
                 ic2++;
               }
         }

      if(cont==3){
             ic2=0;
               while((carac2[ic2]=getc(p)) != EOF){
                   if((carac=getc(p)) == ':'){

      if(carac2[ic2]=='1')printf("[ANUAL][JANEIRO]:: ");

if(carac2[ic2]=='2')printf("[ANUAL][FEVEREIRO]:: ");

if(carac2[ic2]=='3')printf("[ANUAL][MARCO]:: ");

if(carac2[ic2]=='4')printf("[ANUAL][ABRIL]:: ");

if(carac2[ic2]=='5')printf("[ANUAL][MAIO]:: ");

if(carac2[ic2]=='6')printf("[ANUAL][JUNHO]:: ");

      if(carac2[ic2]=='7')printf("[ANUAL][JULHO]:: ");

if(carac2[ic2]=='8')printf("[ANUAL][AGOSTO]:: ");

if(carac2[ic2]=='9')printf("[ANUAL][SETEMBRO]:: ");
                      while((carac=getc(p)) !=
'\n')printf("%c",carac);
                        printf("\n");
                        ic2=-1;
                    }
               else{
                        carac2[ic2+1]=carac;

      if(carac2[ic2+1]=='0')printf("[ANUAL][OUTUBRO]:: ");

if(carac2[ic2+1]=='1')printf("[ANUAL][NOVEMBRO]:: ");

if(carac2[ic2+1]=='2')printf("[ANUAL][DEZEMBRO]:: ");
                      while((carac=getc(p)) != '\n'){
```

```
                          if(carac !=
':')printf("%c",carac);
                        }
                       printf("\n");
                       ic2=-1;
                    }
               ic2++;
            }
       }

 fclose(p);
 cont++;
 j=0;
 i=0;
 printf("\n");
 }
}

criarqs()
{
 FILE *p;
 p = fopen("AG_DIA.DB","a");
 fclose(p);
 p = fopen("AG_SEM.DB","a");
 fclose(p);
 p = fopen("AG_MES.DB","a");
 fclose(p);
 p = fopen("AG_ANO.DB","a");
 fclose(p);
}
```

Anexo B:

Código Fonte do aplicativo agendaexec

```
/*
Agenda de Tarefas Auxiliando a Administração de Servidores
Linux - Executor
Copyright (C) 2004  Rafael Reis Bezerra

This program is free software; you can redistribute it
and/or modify it under the terms of the GNU General Public
License as published by the Free Software Foundation;
either version 2 of the License, or (at your option) any
later version.

This program is distributed in the hope that it will be
useful, but WITHOUT ANY WARRANTY; without even the implied
warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR
PURPOSE.  See the GNU General Public License for more
details.

You should have received a copy of the GNU General Public
License along with this program; if not, write to the Free
Software Foundation, Inc., 59 Temple Place - Suite 330,
Boston, MA  02111-1307, USA.
*/


#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>

int main()
{
  struct tm mytime;
  time_t tempo;
  int  mes, dia, ano, sem, semt, j=0, i=0, reg=0,
ic2,inicio;
  FILE *p;
  char
carac,caracsem[10],caracmes[10],caracdia[1],carac2[2],semst
r[15],messtr[15],conteudo[150];
  tempo = time(NULL);
```

```
mytime = *localtime(&tempo);

mes = mytime.tm_mon + 1;
dia = mytime.tm_mday;
ano = mytime.tm_year + 1900;
sem = mytime.tm_wday;

if(sem==0)strcpy(semstr,"Domingo");
if(sem==1)strcpy(semstr,"Segunda-feira");
if(sem==2)strcpy(semstr,"Terca-feira");
if(sem==3)strcpy(semstr,"Quarta-feira");
if(sem==4)strcpy(semstr,"Quinta-feira");
if(sem==5)strcpy(semstr,"Sexta-feira");
if(sem==6)strcpy(semstr,"Sabado");

if(mes==1)strcpy(messtr,"Janeiro");
if(mes==2)strcpy(messtr,"Fevereiro");
if(mes==3)strcpy(messtr,"Marco");
if(mes==4)strcpy(messtr,"Abril");
if(mes==5)strcpy(messtr,"Maio");
if(mes==6)strcpy(messtr,"Junho");
if(mes==7)strcpy(messtr,"Julho");
if(mes==8)strcpy(messtr,"Agosto");
if(mes==9)strcpy(messtr,"Setembro");
if(mes==10)strcpy(messtr,"Outubro");
if(mes==11)strcpy(messtr,"Novembro");
if(mes==12)strcpy(messtr,"Dezembro");
printf("\nAGENDA AUTOMATICA - ");
printf("%s, %d de %s de %d\n",semstr,dia,messtr,ano);
printf("\nLembretes registrados para hoje.\n");
criarqs();

if (!(p = fopen("AG_DIA.DB","r")))
    {
          printf("Erro! Impossivel abrir o arquivo!\n");
          exit(1);
    }

while(!feof(p)){
          fgets(conteudo,149,p);
          j++;
    }
j--;
fclose(p);
if((p = fopen("AG_DIA.DB", "r")) == NULL)
    {
          printf("Erro na abertura do arquivo");
```

```
          exit(1);
    }
for(i=0;i<j;i++)
    {
    fgets(conteudo,149,p);
    printf("\n[DIARIO]::\n:: %s",conteudo);
    reg++;
    }
fclose(p);


if (!(p = fopen("AG_SEM.DB","r")))
     {
          printf("Erro! Impossivel abrir o arquivo!\n");
          exit(1);
     }

sprintf(caracsem,"%d",sem);
inicio=1;
while((carac=getc(p)) != EOF)
{
      if(carac == caracsem[0] && inicio==1){
            printf("\n[SEMANAL][%s]::\n:: ",semstr);
            reg++;
            while((carac=getc(p)) != '\n'){
                if(carac != ':')printf("%c",carac);
            }
      if(carac == '\n')printf("\n");
      }
inicio=0;
if(carac == '\n')inicio=1;
}
fclose(p);

if (!(p = fopen("AG_MES.DB","r")))
    {
          printf("Erro! Impossivel abrir arquivo!\n");
          exit(1);
    }

ic2=0;
sprintf(caracdia,"%d",dia);
while((carac2[ic2]=getc(p)) != EOF)
{
   if(carac2[ic2]!='\n'){
      if((carac=getc(p)) == ':')
      {
```

```
             if(caracdia[0]==carac2[ic2]){
                   printf("\n[MENSAL][%c]::\n::
",carac2[ic2]);
                     while((carac=getc(p)) !=
'\n')printf("%c",carac);
                     printf("\n");
             }
             ic2=-1;
        }
        else
        {
             carac2[ic2+1]=carac;
             if(carac2[ic2]==caracdia[0] &&
carac2[ic2+1]==caracdia[1]){
                 printf("\n[MENSAL][%c%c]::\n::
",carac2[ic2],carac2[ic2+1]);
                 while((carac=getc(p)) != '\n'){
                       if(carac != ':')printf("%c",carac);
                 }
                   printf("\n");
             }
             ic2=-1;
        }
        ic2++;
     }

 }
 fclose(p);

 if (!(p = fopen("AG_ANO.DB","r")))
       {
            printf("Erro! Impossivel abrir arquivo!\n");
            exit(1);
       }

 ic2=0;
 sprintf(caracmes,"%d",mes);
 while((carac2[ic2]=getc(p)) != EOF){
      if(carac2[ic2]!='\n'){
           if((carac=getc(p)) == ':'){
              if(caracmes[0]==carac2[ic2]){


if(carac2[ic2]=='1')printf("\n[ANUAL][JANEIRO]::\n:: ");

if(carac2[ic2]=='2')printf("\n[ANUAL][FEVEREIRO]::\n:: ");
```

```
if(carac2[ic2]=='3')printf("\n[ANUAL][MARCO]::\n:: ");

if(carac2[ic2]=='4')printf("\n[ANUAL][ABRIL]::\n:: ");

if(carac2[ic2]=='5')printf("\n[ANUAL][MAIO]::\n:: ");

if(carac2[ic2]=='6')printf("\n[ANUAL][JUNHO]::\n:: ");

if(carac2[ic2]=='7')printf("\n[ANUAL][JULHO]::\n:: ");

if(carac2[ic2]=='8')printf("\n[ANUAL][AGOSTO]::\n:: ");

if(carac2[ic2]=='9')printf("\n[ANUAL][SETEMBRO]::\n:: ");
                  while((carac=getc(p)) !=
'\n')printf("%c",carac);
                  printf("\n");
             }
                 ic2=-1;
         }
           else{
                 carac2[ic2+1]=carac;
                 if(carac2[ic2]==caracmes[0] &&
carac2[ic2+1]==caracmes[1]){

if(carac2[ic2+1]=='0')printf("\n[ANUAL][OUTUBRO]::\n:: ");

if(carac2[ic2+1]=='1')printf("\n[ANUAL][NOVEMBRO]::\n:: ");

if(carac2[ic2+1]=='2')printf("\n[ANUAL][DEZEMBRO]::\n:: ");
                      while((carac=getc(p)) != '\n'){
                            if(carac !=
':')printf("%c",carac);
                      }
                      printf("\n");
                 }
                 ic2=-1;
           }
           ic2++;
       }
   }


  printf("\n");
  if(reg==0)printf("\nNenhum registro armazenado em sua
agenda,\npara o dia de hoje.\n");
  fclose(p);
```

```
}

criarqs()
{
 FILE *p;
 p = fopen("AG_DIA.DB","a");
 fclose(p);
 p = fopen("AG_SEM.DB","a");
 fclose(p);
 p = fopen("AG_MES.DB","a");
 fclose(p);
 p = fopen("AG_ANO.DB","a");
 fclose(p);
}
```

Anexo C

Script de instalação do módulo PAM (install_ag_pam).

```
#!/bin/bash

echo "Voce deve estar logado como root para executar esse
script."
echo "Instalando Agenda como Modulo PAM..."
cp pam_agendaexec.so /lib/security/
echo "session    optional
/lib/security/pam_agendaexec.so" >> /etc/pam.d/login
echo "OK!"
```