XXX

XXX

Configuração de segurança de sistemas Linux em ambientes acadêmicos

Marco Aurélio Graciotto Silva

n. XXX

TECHNICAL REPORT

São Carlos - SP - Brasil

Nov./2006

XXX

XXX

ISSN: XXX

Configuração de segurança de sistemas Linux em ambientes acadêmicos

Marco Aurélio Graciotto Silva

n. XXX

TECHNICAL REPORT

São Carlos - SP - Brasil

Nov./2006

Prefácio

Resumo

A instalação de computadores com o sistema operacional Linux é uma atividade recorrente em ambientes acadêmicos, seja para fins administrativos (serviços de autenticação, arquivo e impressão) ou científicos (desenvolvimento de aplicativos, execução de experimentos, divulgação de pesquisas). No entanto, observa-se que não existe preocupação, durante a instalação desses sistemas, de configurá-los adequadamente, de modo a preservar sua integridade e segurança. Este relatório explica as falhas de segurança mais comuns e apresenta técnicas simples para proteger o sistema de tais falhas ou, pelo menos, diminuir a extensão dos danos causados pelas falhas de segurança.

Motivação

Ter uma máquina invadida é algo que ninguém deseja. E, em apenas uma semana, uma mesma máquina que utilizo foi invadida duas vezes (pelo menos). A primeira vez apenas para instalar um programa que coordena e participa de um *Denial of Service* (Negação de serviço) (DoS), o que foi simples de identificar e resolver. A segunda vez... até hoje não tenho certeza de como foi a invasão, só sei que o sistema foi suficientemente comprometido para exigir uma reinstalação. Despender uma tarde e uma noite inteiras para averiguar o problema não era exatamente o que eu tinha planejado para aquele dia. No entanto, por se tratar de uma máquina importante, você não tem muita opção: ou a máquina é arrumada ou diversas pesquisas do grupo precisam ser interrompidas.

A principal motivação para esse relatório técnico é evitar que mais pessoas percam preciosas horas na recuperação de tais incidentes.

Público alvo

Este relatório destina-se a alunos de graduação e pós-graduação que administram sistemas Linux. Supõe-se que o leitor tenha conhecimentos básicos de sistemas operacionais e redes, bem como de sistemas Linux.

Garantias

Seguir as instruções deste documento não é uma garantia de que um computador estará protegido. As instruções aqui contidas cobrem as técnicas que o autor conhece e impediriam o comprometimento da máquina por alguém como o mesmo, o que ainda deixa um grande universo de alternativas disponíveis. De fato, se uma pessoa capacitada realmente deseja invadir ou comprometer uma máquina, ela o fará, independentemente do quão bem configurada ela for (veja os relatos que existem na Internet sobre invasões à máquinas sensíveis de segurança, como as de militares).

Apesar de não serem oferecidas garantias, é certo que as informações contidas neste documento corrigem falhas óbvias de segurança, o que torna a máquina um alvo não trivial para invasão. Probabilisticamente, as chances da máquina ser invadida serão menores.

Sugestões de leitura

A leitura deste documento inteiro é recomendável. Assim, ter-se-á conhecimento não apenas das soluções adotadas, mas também o porquê delas serem como são e qual é o problema que elas tratam. No entanto, se o tempo for extremamente reduzido para melhorar a segurança do seu servidor, recomenda-se a leitura do capítulo 3, sobre as medidas de segurança que devem ser tomadas.

Agradecimentos

Primeiramente, eu gostaria de agradecer a sítios como `lwn.net`, que mostram uma visão simplificada deste complicado mundo que é o Linux e suas distribuições. Não sou leitor assíduo de sites de segurança, mas deveria. Eles também merecem méritos. O mesmo para os inúmeros desenvolvedores que compõem essa fantástica comunidade de software livre. Antes a liberdade de saber que problemas existem do que a falsa sensação de segurança que os fabricantes de programas de código fechado tanto defendem.

Sumário

1 Introdução **1**

2 Vulnerabilidades **3**

2.1 Natureza das ameaças . . . 3

2.2 Tipos de vulnerabilidade . . . 4

2.2.1 Ameaças acidentais e naturais . . . 4

2.2.2 Ameaças acidentais e de natureza humana . . . 4

2.2.3 Ameaças propositais . . . 4

2.3 Considerações finais . . . 5

3 Medidas **7**

3.1 Fornecimento de energia elétrica . . . 7

3.2 Cabeamento elétrico e de rede . . . 8

3.3 Disposição física do computador . . . 8

3.4 Definição de senha para a BIOS . . . 8

3.5 Inicialização do computador . . . 8

3.6 Grub . . . 9

3.7 Login do usuário root . . . 9

3.8 Restringir pessoas que podem executar comandos como super-usuário . . . 10

3.9 System Request . . . 10

3.10 Criação de novos usuários . . . 11

3.11 Desabilitar o login do usuário “root” pelo ssh . . . 11

3.12 Impedir ataques para quebra de senha por força bruta por SSH . . . 11

3.13 Máscara padrão para novos arquivos e diretórios . . . 12

3.14 Prevenir negação de serviços (DoS) . . . 12

3.15 Conexões indesejadas . . . 13

3.16 Configuração das contas utilizadas pelos serviços . . . 17

3.17 Configuração do PHP . . . 17

3.18 Configuração do Apache 18
3.19 Desabilitar serviços desnecessários 19
3.20 Considerações finais 19

4 Eventualidades **21**
4.1 Preservar o sistema afetado 21
4.2 Colete dados 22
4.3 Analisar os dados coletados 22
4.4 Projetar uma solução para a falha 22
4.5 Implementar a solução 23

Referências Bibliográficas **25**

Lista de Acrônimos

DoS *Denial of Service* (Negação de serviço) i

Capítulo 1

Introdução

A instalação de computadores com o sistema operacional Linux é uma atividade recorrente em ambientes acadêmicos, seja para fins administrativos (serviços de autenticação, arquivo e impressão) ou científicos (desenvolvimento de aplicativos, execução de experimentos, divulgação de pesquisas). Antigamente, a disponibilização de tais serviços era restrita aos setores específicos de informática dos departamentos, devido à complexidade da configuração e até mesmo a maior exigência de hardware para a execução dos serviços. Hoje o cenário é bem diferente: mesmo máquinas obsoletas (com mais de cinco anos de vida) são suficientes e o software é de fácil instalação.

Um cenário bem típico, e utilizado para este relatório, é de um projeto que conte com apoio financeiro de agências de fomento governamentais, com um professor (por conseqüência de reconhecido mérito científico) responsável pelo projeto e alguns alunos (de pós-graduação e graduação) que executam o projeto sob a orientação deste professor. Esses projetos, em sua maioria, não aprovam a contratação de uma pessoa para apoio operacional, ou seja, para a administração dos equipamentos por ventura adquiridos para a execução do projeto. Logo, o professor, muito provavelmente, delega as tarefas de instalação e manutenção dos servidores aos seus orientandos. Estes, por sua vez, dispõem de pouco tempo para essa tarefa: os de pós-graduação estão muito envolvidos com a pesquisa e divulgação dos resultados; os de graduação estão muito ocupados com as disciplinas e ainda não possuem o conhecimento adequado para manter um sistema em execução sob boas condições. Logo, a configuração das máquinas é feita da maneira simples, com a instalação de alguma distribuição Linux popular (Debian, Fedora, SUSE, Ubuntu, etc) e a configuração mínima dos serviços necessários (Apache, PHP, etc).

Observa-se que, tipicamente, não existe preocupação, durante a instalação desses computadores, de configurá-los adequadamente, de modo a prevenir o comprometimento da máquina. Seria correto supor que as distribuições Linux implementam medidas para aumentar o nível de segurança da máquina. Infelizmente, não é isso o que se observa. Por muitas vezes, elas privilegiam a facilidade de uso ao detrimento da sua segurança. Por mais que

elas lancem, freqüentemente, atualizações para seus programas, várias configurações dos mesmos mantêm-se inalteradas e, conseqüentemente, inseguras. E existe o problema de que algumas medidas de segurança não podem ser tomadas pelo sistema operacional, ficando sua execução a encargo da pessoa responsável pela máquina.

O objetivo deste relatório é explicar as falhas de segurança mais comuns e apresentar técnicas simples para proteger o sistema de tais falhas ou, pelo menos, diminuir o alcance dos danos decorrentes das falhas de segurança.

As seguintes convenções são adotadas neste texto: palavras em caracteres mono-espaçados são comandos de sistema; palavras em idioma estrangeiro (inglês) estão em itálico; nomes de programas estão em letras normais. Alguns termos em português serão acompanhados de sua tradução em inglês ou a nomenclatura utilizada em software livre: nesse caso, o texto estará em itálico e entre parênteses. Nos exemplos que contêm scripts (aqueles que se iniciam com o caracter '#'), o caracter '\', quando encontrado no final da linha, identifica que a linha continua na linha abaixo.

Capítulo

2

Vulnerabilidades

O que é uma falha de segurança e por que nos preocupamos com ela? Todos os sistemas, mecânicos ou biológicos, exibem falhas. Uma pessoa, quando contrai uma gripe, demonstra uma deficiência em seu sistema imunológico. Uma máquina, ao ser invadida, acusa um erro em algum software que controla o acesso, via rede ou terminal local, ao sistema. A grande diferença entre esses "sistemas" é que o primeiro consegue se recuperar da sua falha enquanto que o segundo não. Em outras palavras, o prejuízo da vulnerabilidade do primeiro são alguns dias de dor de cabeça; o do segundo é a perda irrecuperável do sistema

Os computadores, apesar de toda a evolução, ainda são sistemas demasiadamente simples e pouco tolerantes à ocorrência de problemas. Não é necessário muito para desativá-los ou degradar o funcionamento dos mesmos o suficiente para deixá-los inúteis. É utópico achar que é possível tornar um sistema computacional isento de vulnerabilidades. O que é factível é adotar medidas no sistema que o torne suficientemente resistente ao meio, através da correção dos erros e a execução de ações que reduzam os efeitos das falhas, pelo maior período possível (alguns poucos anos).

Para definir as medidas mais adequadas, é necessário, antes de tudo, conhecer as principais vulnerabilidades apresentadas por um sistema computacional típico. A partir da próxima seção, cada tipo de vulnerabilidade será explicada.

2.1. Natureza das ameaças

Uma ameaça, ou seja, ação que busca explorar as vulnerabilidades de um sistema computacional, pode ser de natureza acidental ou proposital:

• Ameaça acidental: Ação cuja causa é de natureza não humana ou humana sem intenção. Exemplos de ação

da natureza: quedas de energia, inundações, goteiras. Exemplos de ações de pessoas sem intenção de danificar o sistema: derrubar café no computador, provocar choques mecânicos na máquina, não identificar o comprometimento da máquina que mantém e deixá-la em funcionamento (potencialmente tal máquina servirá para atacar outros sistemas).

- Ameaça proposital: Ação cuja causa é de natureza humana, com intenção de danificar o sistema. Exemplos: violar o interior do computador, invadir a máquina por meio de uma falha de um software instalado no sistema.

2.2. Tipos de vulnerabilidade

Um sistema computacional é composto por hardware e software. Ambos possuem suas vulnerabilidades particulares. Por exemplo, sempre existe a possibilidade de alguém arrombar o gabinete ou, menos dramaticamente, colocar uma bolsa em cima do computador e causar uma falha no disco. Já os software podem conter erros que permitam explorar estouros de pilhas, execução arbitrária de código, o que os torna alvos fáceis.

2.2.1. Ameaças acidentais e naturais

Goteiras, marcas de infiltração, aparelhos de ar condicionado (podem expelir água ou gelo caso estejam defeituosos). A lista de problemas físicos no ambiente em que o computador está instalado é extensa. Ao mesmo tempo em que a probabilidade deles ocorrerem e afetarem o computador seja baixa, o dano causado é amplo, muitas vezes exigindo o envio do computador para manutenção e, não raramente, perda definitiva dos dados.

Outro problema, comum no Brasil, é a instabilidade do fornecimento de energia elétrica, com variações de voltagem além das especificações do equipamento, cabeamento de rede instalado paralelamente ao cabeamento elétrico. Infeliz e ironicamente, as redes das universidades não são, em sua imensa maioria, corretamente planejadas e executadas.

2.2.2. Ameaças acidentais e de natureza humana

Laboratórios são constantemente limpos e, neste processo, impactos dos objetos utilizados para limpeza (vassouras, rodos) são freqüentes. A utilização de produtos líquidos para a limpeza também é uma ameaça, visto que respingos podem atingir as máquinas ou a instalação elétrica e de rede das mesmas.

O uso do computador como suporte para livros e bolsas também é comum. Normalmente não seria o problema, não fosse o fato de existir um disco rígido, com pratos girando a 7200 RPM e percorridos por agulhas distantes em micrômetros de sua superfície. Qualquer aceleração imposta a esse delicado mecanismo pode causar danos.

Finalmente, existem os casos em que o usuário executa, sem intenção, um programa malicioso no computador, o vulgo “cavalo de Tróia”.

2.2.3. Ameaças propositais

São as vulnerabilidades encontradas no software e exploradas por pessoas má intencionadas. Não são o tipo mais comum de vulnerabilidade explorada, mas, certamente, são as que mais afligem os administradores de rede e, por

sorte, a que mais medidas facilmente implementáveis existem para correção e prevenção.

Os principais ataques a que os softwares são acometidos são aqueles que visam a execução de código arbitrário, vazamento de informação, negação de serviço e descoberta de senhas (e demais falhas de autenticação).

A execução de código arbitrário talvez seja o tipo de vulnerabilidade mais grave. A partir dele, obtém-se acesso local a máquina e seus dados, o que permite a exploração de falhas que, remotamente, não seria possível (escalada de privilégios, por exemplo).

As principais técnicas para injetar código arbitrário na máquina são: estouro de *buffer* e pilha, formatação de *strings*, código SQL, execução de comandos do sistema, etc. As distribuições Linux cumprem o papel de corrigir tais falhas nas aplicações por elas providas, mas o fato é que as máquinas utilizadas no laboratórios contém muitos softwares instalados ou criados pelo usuários (principalmente programas Web escritos em PHP). Para esses, a responsabilidade pela segurança é inteiramente dos respectivos usuários do sistema e autores dos softwares.

Em aplicações PHP, as maiores vítimas de ataques, é comum a exploração de falhas que permitam a injeção de código em consultas SQL e chamadas de comandos do sistema. Por exemplo, em `system('ls ' + dir)`, em que `dir` é uma variável cujo conteúdo é obtido de uma requisição HTTP, o usuário poderia atribuir à `dir` o valor `/tmp; rm -rf /`. Ou seja, insere-se um comando como parâmetro. Como não é feita uma verificação nos argumentos (por exemplo, utilizando a função `escapeshellcmd()`), a execução do comando apagaria todos os arquivos do sistema (ou, pelo menos, tentaria). E este é apenas um exemplo. Geralmente este artifício é utilizado para executar programas para escalar privilégios ou programar ataques a outras máquinas.

2.3. Considerações finais

Este capítulo apresenta uma classificação simplificada das vulnerabilidades, bem quais são as mais comuns. A intenção não é apresentar toda a fundamentação teórica sobre elas: existem livros especializados no assunto que atendem a esse propósito. O objetivo aqui é lembrar que existem várias vulnerabilidades, algumas das quais negligenciadas quando configurando um sistema. A quantidade de máquinas paradas em laboratórios devido a ameaças acidentais é mais elevado do que por ameaças propositais, para citar um fato.

O próximo capítulo descreve algumas medidas para corrigir as principais vulnerabilidades encontradas nos computadores utilizados em laboratórios acadêmicos, desde aquelas aparentemente tolas, como um cabo de força colocado ao alcance do pé, a falhas graves de configuração de softwares.

Capítulo 3

Medidas

As medidas descritas neste capítulo possuem quatro objetivos:

1. Garantir que o computador funcione.
2. Garantir a aplicação de diretivas de segurança pelo sistema operacional.
3. Evitar o comprometimento do sistema por acesso local à máquina.
4. Evitar o comprometimento do sistema por acesso externa à máquina.

Antes de tudo, é necessário que o computador funcione corretamente. Na prática, isso se traduz em simples cuidados na instalação da máquina e que prevenirão aborrecimentos.

Para garantir a aplicação das diretivas padrões de segurança, é essencial que o sistema operacional padrão do computador seja executado. Seguindo o lógica de inicialização do sistema, é preciso assegurar que o dispositivo de inicialização e o *kernel* padrões não sejam alterados.

Uma vez em operação o sistema operacional e aplicados os mecanismos e políticas de segurança do mesmo, é necessário prevenir que os usuários executem comandos com privilégios de super-usuário (`root`) ou tenham acesso a dados que exijam tal nível de privilégio.

Por fim, é preciso impedir que a máquina seja comprometida pela execução de falhas em serviços oferecidos pelo sistema.

3.1. Fornecimento de energia elétrica

Verifique quão freqüentemente as fontes de energia dos computadores nos laboratórios próximos queimam. Se existem ocorrências desse gênero, é provável que o fornecimento de energia elétrica seja irregular, com variações de

tensão além do suportado pelas fontes, além de picos de energia (por exemplo, após a interrupção de fornecimento devido a uma manutenção na rede elétrica da universidade). Invista, se for o caso, na compra de um estabilizador de tensão ou de um *no-break*.

3.2. Cabeamento elétrico e de rede

Não deixe cabos elétricos e de rede soltos no chão, ao alcance de pés e vassouras. Aparentemente esta é uma recomendação inútil, mas o usuário não pensará o mesmo quando, em um momento de desatenção, ele puxar o cabo do teclado ou mouse e o conector da placa-mãe estragar.

Verifique se os diferentes cabos do computador não estão emaranhados. Isso vale principalmente para cabos elétricos e de rede. Tente, sempre que possível, instalá-los ortogonalmente, ou seja, eles não podem seguir paralelos e juntos.

Certifique-se que a instalação elétrica suporta os equipamentos instalados. Em outras palavras, não sobrecarregue as tomadas, ligando vários dispositivos em um único ponto. Lembre-se que os computadores gastam, em média, 300-400 W. Se existir uma caixa de disjuntores próxima ao laboratório, verifique a carga que os disjuntores suportam e calcule quantas máquinas podem ser instaladas confortavelmente no laboratório, sem sobrecarregar o sistema elétrico.

3.3. Disposição física do computador

Coloque o computador em uma superfície que não sofra muitos impactos. Evite mesas em que o teclado e o gabinete ficam no mesmo plano (não é raro encontrar usuários que golpeiam o teclado e, por tabela, o gabinete).

Oriente os usuários dos computadores para não colocar pesos em cima do computador: bolsas, livros, etc. O problema, em si, não é o peso adicional e sim o ato de colocar, que acaba por aplicar uma força indesejável no computador, no pior sentido possível (perpendicular ao movimento da cabeça de leitura e escrita do disco rígido).

Certifique-se que gabinete do computador está estabilizado na mesa. Se necessário, coloque calços no gabinete, minimizando o balançar do mesmo.

3.4. Definição de senha para a BIOS

Defina uma senha para a BIOS. Isto evitará que as pessoas modifiquem as configurações da BIOS (mais especificamente para alterar as configurações de inicialização do computador). Geralmente a opção da BIOS para alterar a senha encontra-se no item `Set Supervisor Password`.

3.5. Inicialização do computador

O primeiro passo é impedir que uma pessoa má intencionada desabilite os recursos de segurança de seu computador, ou seja, impedir que outro sistema operacional, que não o instalado no computador, seja executado. Por exemplo, se a opção de inicialização da máquina por um dispositivo externo, como o leitor de CD-ROM, estiver habilitada, seria possível inicializar a máquina com um *Live CD* ou similar, o que permitiria o uso irrestrito do computador.

Para desativar a inicialização do computador por dispositivos externos, altere os parâmetros da BIOS, habilitando apenas o disco rígido para a inicialização. Geralmente a BIOS oferece opções de `Fist Boot Device`, `Second Boot Device`, `Thrid Boot Device` e `Boot Other Device`. Altere o `First Boot Record` para `HDD-0` ou equivalente. Deixe todos os demais itens em branco ou desativados (`Disabled`).

Verifique também se o sistema oferece a funcionalidade de `Boot Menu`. O `Boot Menu` permite ao usuário a escolha do dispositivo de inicialização sem a necessidade de acessar as configurações da BIOS, bastando apertar, geralmente, a tecla `ESC`. Esse recurso deve ser desativado na configuração da BIOS.

3.6. Grub

A maioria das distribuições Linux utiliza o `Grub` para selecionar o dispositivo a ser utilizado para inicialização do sistema e o sistema operacional a ser executado. O problema é que o `Grub` não permite apenas a seleção, mas também a alteração das opções já existentes. Isso permite que, por exemplo uma pessoa má intencionada altere os parâmetros de inicialização do kernel para permitir o uso do sistema diretamente como usuário "root".

O `Grub` permite a atribuição de uma senha para evitar as alterações de suas opções. Para configurar a senha, altere o parâmetro `password` do arquivo `/boot/grub/menu.lst`. O valor a ser atribuído pode ser uma palavra em texto limpo ou uma mensagem MD5. O ideal é que se utilize uma mensagem MD5, de modo a não comprometer a senha caso o arquivo `/boot/grub/menu.lst` seja lido do por um usuário não autorizado. A senha, como mensagem MD5, é gerada pelo comando `grub-md5-crypt`. Por exemplo, para a senha `teste123`, teríamos o seguinte resultado:

```
# grub-md5-crypt
Password:
Retype password:
$1$O65.W1$j.fvWLvEpD039K4sPAfpv1
```

Copie a mensagem (no caso do exemplo, `\$1\$O65.W1\$j.fvWLvEpD039K4sPAfpv1`) para o parâmetro `password` do arquivo `/boot/grub/menu.lst`:

```
password --md5 $1$O65.W1$j.fvWLvEpD039K4sPAfpv1
```

Finalmente, altere as permissões de acesso ao arquivo, restringindo-as para o proprietário do arquivo e as negando para os demais (`chmod 0600 /boot/grub/menu.lst`).

3.7. Login do usuário root

O arquivo `/etc/securetty` define os dispositivos a partir de quais o usuário `root` pode entrar no sistema. Reduza ao mínimo, para não dizer eliminar, o conteúdo do mesmo (restrinja ao dispositivo `tty1`, por exemplo). Utilize, sempre que possível, o `sudo` para permitir que um usuário adquira, temporariamente, direitos de super-usuário.

3.8. Restringir pessoas que podem executar comandos como super-usuário

Restrinja as pessoas que podem executar comandos como super-usuário (`root`). Para isso, desative a conta de `root`:

- Apague a senha do usuário `root` em `/etc/shadow`, substituindo-a por um asterisco (`*`). A senha, no `/etc/shadow`, fica entre o primeiro e segundo sinal de "`:`".
- Altere o `shell` do usuário `root`, no arquivo `/etc/passwd`, de `/bin/bash` para `/bin/false`.

A seguir, configure os usuários normais do sistema que poderão adquirir privilégios de super-usuário para executar programas na máquina. Isso é possível com o `sudo` (leia como "*as super-user, do*" - "como super-usuário, faça"). Veja um exemplo de utilização do `sudo`:

```
# sudo ls /root
Password:
Desktop
.bashrc
```

A configuração do `sudo` é realizada pelo arquivo `/etc/sudoers`. Ele deve ser editado com o comando `visudo` (nenhum parâmetro, como o nome do arquivo, é necessário). Sugere-se a seguinte configuração:

```
Defaults env_reset
Defaults log_host
Defaults requiretty
Defaults syslog_goodpri=info
Defaults syslog_badpri=alert
Defaults lecture=always
Defaults syslog=authpriv

%admin ALL=(ALL) ALL
```

O significado de cada parâmetro é:

• `env_reset`: Apaga todas as variáveis de ambiente.
• `log_host`: Registra o nome do computador (`hostname`) a partir do qual utilizou-se o comando `sudo`.
• `requiretty`: Exige que o usuário tenha um terminal (`tty`) associado à sessão para executar o comando.
• `syslog_goodpri`: Controla o registro de autenticações bem sucedidas.
• `syslog_badpri`: Controla o registro de autenticaçãoes má sucedidas.
• `lecture`: Mostra um texto sobre o uso do `sudo` sempre que a senha for solicitada.

3.9. System Request

O `System Request` (`SysRq`) permite a interrupção da execução de qualquer programa. Ele é geralmente utilizado para realizar a manutenção do sistema, mas também pode ser utilizado para desligar o sistema e obter outros acessos privilegiados ao mesmo. Para desabilitá-lo, edite o arquivo `/etc/sysctl.conf` e realize a seguinte alteração:

```
# Disable the magic-sysrq key
kernel/sysrq = 0
```

3.10. Criação de novos usuários

Edite o arquivo `/etc/adduser.conf` e altere a permissão padrão configurada para o diretório principal (`home`) de um usuário no momento de criação de sua conta, negando as permissões de leitura e execução para os demais usuários do sistema. O parâmetro em questão é o `DIR_MODE` e o valor sugerido é `0750`. Por exemplo:

```
# If DIR_MODE is set, directories will be created with the specified
# mode. Otherwise the default mode 0755 will be used.
DIR_MODE=0750
```

Essa configuração permite a leitura e escrita ao proprietário, a leitura para os usuários que pertençam ao mesmo grupo e nenhum direito para os demais usuários.

3.11. Desabilitar o login do usuário "root" pelo ssh

Para impedir que o servidor de SSH, o `OpenSSH`, autentique o usuário "root", altere a diretiva `PermitRootLogin`, em `/etc/ssh/sshd_config` para `No`:

```
PermitRootLogin no
```

3.12. Impedir ataques para quebra de senha por força bruta por SSH

Sessões seguras são imperativas para o uso e administração de sistemas acadêmicos. No entanto, elas também expõem a máquina ao acesso externo. Na prática, a menos que o suposto invasor tenha conhecimento sobre as pessoas que utilizam a máquina e consigam descobrir a senha, trata-se de um meio de ataque ineficaz. Apesar disso, muitos administradores prefeririam que mesmo esta chance remota fosse afastada. As regras abaixo bloqueiam tal ataques de força bruta contra o SSH. Elas precisam ser recarregadas a cada inicialização da máquina, então sugere-se que elas sejam inseridas no final do arquivo `/etc/rc.local`.

```
# Identify an SSH brute-force login attack
iptables -N ssh-attack-phase1
iptables -N ssh-attack-phase2
iptables -A ssh-attack-phase1 -m recent --name ssh-attack \
    --rcheck --seconds 60 --hitcount 3 -j ssh-attack-phase2
iptables -A ssh-attack-phase1 -m recent --set --name ssh-attack

# Block a host trying an SSH brute-force login attack
iptables -A ssh-attack-phase2 -m recent --name ssh-attack --update
iptables -A ssh-attack-phase2 -m limit -j LOG --log-prefix="SSH attack" \
    --log-level alert
iptables -A ssh-attack-phase2 -j DROP

# Look initial connections at port 22 (TCP) and look for SSH brute force attacks.
```

```
iptables -A INPUT -p tcp --dport 22 --syn -j ssh-attack-phase1
iptables -A FORWARD -p tcp --dport 22 --syn -j ssh-attack-phase1
```

3.13. Máscara padrão para novos arquivos e diretórios

Desabilite a permissão de leitura para novos arquivos e diretórios para todos os usuários do sistema que não pertençam ao grupo a que pertence o arquivo. Altere o arquivo `/etc/bashrc` ou `/etc/bash.bashrc` e acrescente (ou altere) a linha:

```
umask 127
```

Consulte o endereço `http://en.wikipedia.org/wiki/Umask` para informações mais detalhadas sobre a configuração do `umask`.

3.14. Prevenir negação de serviços (DoS)

Altere o arquivo `/etc/sysctl.conf` e acrescente as linhas seguintes (ou, caso elas já existam, certifique-se que os valores definidos estão corretos):

```
# Report bogus responses to broadcast frames (sent by routers that violates RFC1122)
net/ipv4/icmp_ignore_bogus_error_responses = 0

# Disables replies to broadcast ICMP echo (ping), a common DoS attack
net/ipv4/icmp_echo_ignore_broadcasts = 1

# Enable the 'Source Address Verification' (protect against IP spoofing)
net/ipv4/conf/default/rp_filter=1
net/ipv4/conf/all/rp_filter = 1

# Enable TCP SYN Cookie Protection
net/ipv4/tcp_syncookies = 1

# Turn on the tcp_timestamps
net/ipv4/tcp_timestamps = 1
```

Modifique o arquivo `/etc/rc.local` e acrescente os seguintes comandos:

```
#!/bin/bash

for f in /proc/sys/net/ipv4/conf/*/{log_martians,rp_filter};
do
    echo 1 > $f
done

for f in /proc/sys/net/ipv4/conf/*/{accept_redirects,accept_source_route};
do
    echo 0 > $f
done
```

O significado de cada parâmetro das interfaces de rede é:

- `accept_redirects`: Controla os pedidos para redirecionamento de pacotes.
- `accept_source_route`: Controla o uso de pacotes *source-routed*.
- `log_martians`: Registra a ocorrência de números IP inválidos.
- `rp_filter`: Valida a origem do pacote IP, utilizando a técnica de caminho inverso (*reversed path*), conforme especificado no RFC 1812.
- `tcp_timestamps`: Habilita a proteção contra *wrapped sequences* (PAWS) e *round trip time measurement* (RTTM).

3.15. Conexões indesejadas

Evitar o universo de máquinas que podem conectar em determinados serviços, provavelmente a um conjunto de máquinas consideradas seguras, não apenas evita ataques como, na ocorrência de um, facilita a descoberta da origem do ataque.

Muitos programas utilizam a biblioteca `tcpwrappers` para definir que computadores podem se conectar aos serviços disponibilizados. A vantagem é que é possível definir, em apenas dois arquivos, as configurações de vários programas, o que facilita a administração do sistema. Os dois arquivos envolvidos são o `/etc/hosts.allow` e `/etc/hosts.deny`. Primeiramente, a biblioteca `tcpwrappers` verifica se a máquina que deseja se conectar consta em `/etc/hosts.allow`. Caso positivo, a conexão será permitida. Caso contrário, o arquivo `/etc/hosts.deny` é verificado. Se a máquina constar nesse arquivo, a conexão não é permitida. Finalmente, se o endereço da máquina não constar em nenhum desses arquivos, a conexão é permitida.

Por padrão, nega-se a conexão a qualquer computador. Para isso, altera-se o arquivo `/etc/hosts.deny`:

```
ALL: ALL
```

Então, configura-se o arquivo `/etc/hosts.allow` com os endereços IPs das máquinas cujas conexões serão permitidas (no lugar de `192.168.1.0/255.255.255.0`, informe a configuração da sub-rede à qual o seu computador pertence):

```
ALL:      192.168.1.0/255.255.255.0
```

Não são todos os programas que utilizam a biblioteca `tcpwrappers`. Por exemplo, o Apache não a utiliza. Nesse caso, deve-se recorrer ao *firewall* do Linux, o *iptables*. O *script* a seguir configura um *firewall* que bloqueia todas as portas menos a 22 (SSH) e 80 (HTTP). Edite o *script* de acordo com suas necessidades:

```
# Reset the firewall configuration
iptables -F

# Drop every IN connection unless told otherwise
iptables -P INPUT DROP

# Deny output and forward connections from anywhere
iptables -P OUTPUT DROP
iptables -P FORWARD DROP

# Allow loopback connections
iptables -A INPUT -s 127.0.0.0/8 -d 127.0.0.0/8 -i lo -j ACCEPT
```

```
iptables -A OUTPUT -s 127.0.0.0/8 -d 127.0.0.0/8 -o lo -j ACCEPT

# Allow communication initially established by this computer
#
iptables -A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
iptables -A OUTPUT -m state --state NEW,ESTABLISHED,RELATED -j ACCEPT

# Allow DNS access
#
iptables -A INPUT -p tcp --sport 53 --dport 1024:65535 -j ACCEPT
iptables -A INPUT -p udp --sport 53 --dport 1024:65535 -j ACCEPT
iptables -A OUTPUT -p tcp --dport 53 --sport 1024:65535 -j ACCEPT
iptables -A OUTPUT -p udp --dport 53 --sport 1024:65535 -j ACCEPT

# Allow input connections to some services
#
iptables -A INPUT -p tcp --dport 22 --sport 1024:65535 -m state --state NEW -j ACCEPT # SSH
iptables -A INPUT -p tcp --dport 80 --sport 1024:65535 -m state --state NEW -j ACCEPT # HTTP
iptables -A INPUT -p tcp --dport 111 -j ACCEPT  # RPC (portmapper)
iptables -A INPUT -p udp --dport 111 -j ACCEPT  # RPC (portmapper)
iptables -A INPUT -p tcp --dport 389 -j ACCEPT  # LDAP
iptables -A INPUT -p tcp --dport 636 -j ACCEPT  # LDAPS
iptables -A INPUT -p tcp --dport 2049 -j ACCEPT # NFS
iptables -A INPUT -p udp --dport 2049 -j ACCEPT # NFS


# Disable output connection from ports above 1024 (useful for servers)
#
iptables -A OUTPUT -p tcp --sport 1024:65535 -j DROP
iptables -A OUTPUT -p tcp --dport 1024:65535 -j DROP
iptables -A OUTPUT -p udp --sport 1024:65535 -j DROP
iptables -A OUTPUT -p udp --dport 1024:65535 -j DROP

# Syn-flood protection
#
iptables -A FORWARD -p tcp --syn -m limit --limit 1/s -j ACCEPT

# Furtive port scanner
#
iptables -A FORWARD -p tcp --tcp-flags SYN,ACK,FIN,RST RST -m limit \
    --limit 1/s -j ACCEPT

# Ping of death
iptables -A FORWARD -p icmp --icmp-type echo-request -m limit \
    --limit 1/s -j ACCEPT



#
# Handle bad packets
#
iptables -N bad-packets

# Log bad packets
#
iptables -A bad-packets -m limit -j LOG --log-prefix="Bad packet" \
    --log-level alert

# Drop bad packets
#
iptables -A bad-packets -j DROP

# Invalid packets
iptables -A INPUT -m state --state INVALID -j bad-packets
```

```
iptables -A FORWARD -m state --state INVALID -j bad-packets

# Malformed packets
#
iptables -A INPUT -p tcp -m tcp --tcp-flags FIN,SYN,RST,PSH,ACK,URG NONE -j bad-packets
iptables -A INPUT -p tcp -m tcp --tcp-flags FIN,SYN FIN,SYN -j bad-packets
iptables -A INPUT -p tcp -m tcp --tcp-flags SYN,RST SYN,RST -j bad-packets
iptables -A INPUT -p tcp -m tcp --tcp-flags FIN,RST FIN,RST -j bad-packets
iptables -A INPUT -p tcp -m tcp --tcp-flags FIN,ACK FIN -j bad-packets
iptables -A INPUT -p tcp -m tcp --tcp-flags ACK,URG URG -j bad-packets
iptables -A INPUT -m unclean -j bad-packets


#
# Handle spammers, hackers, and so on
#
iptables -N spammer

# Log spammers
#
iptables -A spammer -m limit -j LOG --log-prefix="Spammer" --log-level alert

# Drop packets from spammers
#
iptables -A spammer -j DROP
iptables -A INPUT -j REJECT --reject-with icmp-host-prohibited

# Anythong related to SMTP (port 25) should be dropped.
#
iptables -A INPUT -p tcp -m tcp --dport 25 -j spammer

# Block unattended connections at port 22 (ssh)
#
iptables -A INPUT -p tcp -m tcp --dport 22 -m limit --limit 1/sec -j spammer

# Block pseudo-hackers
#
iptables -A INPUT -s scum.spammers.org -j spammer
iptables -A INPUT -s script.kiddies.com -j spammer
```

Uma observação importante sobre o uso de NFS e Iptables. O NFS utiliza o serviço de RPC e esse designa, por padrão, portas aleatórias para alguns serviços (execute o comando `rpcinfo -p` para descobrir as portas em uso). Para configurar as regras do firewall para o NFS, é necessário configurar portas fixas para o RPC. As instruções aqui contidas foram criadas a partir do tutorial disponível em `http://howtos.rlworkman.net/NFS_Firewall_HOWTO`.

Edite o arquivo `/etc/services` e adicione as seguintes linhas:

```
mountd          861/udp                 # NFS mountd
mountd          861/udp                 # NFS mountd
rquotad         863/udp                 # NFS rquotad
rquotad         863/tcp                 # NFS rquotad
status          865/udp                 # NFS status (listen)
status          865/tcp                 # NFS status (listen)
status          866/udp                 # NFS status (send)
status          866/tcp                 # NFS status (send)
lockd           4045/udp                # NFS lock daemon/manager
lockd           4045/tcp                # NFS lock daemon/manager
```

No Ubuntu, edite o arquivo `/etc/default/nfs-common` e altere a variável `STATDOPTS`, configurando a porta

de entrada para 865 e de saída para 866. Observe que os valores assumidos por essa variável são parâmetros repassados para o `statd`. Portanto, para utilizar a porta 865, o valor da variável deve ser `--port 865`:

```
# Options for rpc.statd.
#   Should rpc.statd listen on a specific port?
#   If so, set this variable to a statd argument like: "--port 1000".
STATDOPTS=--port 865 --outgoing-port 866
```

Edite também o arquivo `/etc/default/nfs-kernel-server` e altere a variável `RPCMOUNTDOPTS` de modo semelhante à configuração do `statd`:

```
# Options for rpc.mountd
RPCMOUNTDOPTS=--port 861
```

Edite também o arquivo `/etc/default/quota` e altere a variável `RPCRQUOTADOPTS` de modo semelhante à configuração do `statd`:

```
# Add options to rpc.rquotad here
RPCRQUOTADOPTS=--port 863
```

Configure o gerenciador de travas do NFS para uma porta fixa. Crie o arquivo `/etc/modprobe.d/nfs` e insira o seguinte conteúdo:

```
options lockd nlm_udpport=4045 nlm_tcpport=4045
```

Finalmente, adicione as novas regras para permitir o acesso a esses serviços:

```
iptables -A INPUT -p udp --dport 865:866 -j ACCEPT  # RPC (status)
iptables -A INPUT -p tcp --dport 865:866 -j ACCEPT  # RPC (status)
iptables -A INPUT -p tcp --dport 861 -j ACCEPT  # RPC (mountd)
iptables -A INPUT -p udp --dport 861 -j ACCEPT  # RPC (mountd)
iptables -A INPUT -p tcp --dport 863 -j ACCEPT  # RPC (quotad)
iptables -A INPUT -p udp --dport 863 -j ACCEPT  # RPC (quotad)
iptables -A INPUT -p tcp --dport 4045 -j ACCEPT # RPC (lock manager)
iptables -A INPUT -p udp --dport 4045 -j ACCEPT # RPC (lock manager)
```

Um nível extra de segurança pode ser alcançado caso se verifique os números IP e os números MAC atribuídos às interfaces de redes dos computadores. O script abaixo realiza tal tarefa:

```
#!/bin/bash
declare -a hosts
hosts=('192.168.1.2 32:4D:9B:5F:C2:4F')

iptables -N suspect_packets
iptables -A suspect_packets -m limit -j LOG --log-level alert \
    --log-prefix "Possible spoofing detected"
iptables -A suspect_packets -j DROP

for i in $(seq 0 $((${#hosts[@]} - 1)))
do
        set -- ${hosts[$i]}
        IP=$1
```

```
        MAC=$2
        iptables -A INPUT -s $IP -m mac --mac-source ! $MAC \
        --match limit -j suspect_packets
done

exit 0
```

3.16. Configuração das contas utilizadas pelos serviços

Os serviços não devem possuir uma *shell* válida associada. Configure-os para utilizar o `/bin/false` como *shell*. Isso faz-se pela alteração do arquivo `/etc/passwd` e a substituição do último parâmetro (geralmente `/bin/bash`) para `/bin/false`. O exemplo abaixo contém um serviço, `www-data` e uma conta de usuário, `magsilva`, devidamente configurados.

```
www-data:x:33:33:www-data:/var/www:/bin/false
magsilva:x:1000:1000:Marco Aurélio Graciotto Silva,,,:/home/magsilva:/bin/bash
```

3.17. Configuração do PHP

Por padrão, o PHP permite a execução de programas a partir dos scripts. Essa chamada pode ser realizada por funções específicas (`exec`, `passthru`, `proc_open`, `shell_exec` e `system`) ou pelo operador de execução (aspas simples invertidas). Infelizmente, esses comandos e, principalmente, o operador de execução podem ser utilizados de má fé e devem, portanto, ser desabilitados. Para isso, altere a configuração do PHP (`php.ini`[1]) e habilite a opção `safe_mode`:

```
safe_mode = On
```

Caso deseje flexibilizar o uso do `safe_mode`, configure o parâmetro `safe_mode_exec_dir`, definindo nele os diretórios para os quais o `safe_mode` será ignorado:

```
safe_mode_exec_dir = /usr/local/bin/
```

Caso os programas em PHP instalados na máquina não executem corretamente com o `safe_mode` ativado, é possível desabilitar as funções individualmente por meio do parâmetro `disable_functions`. Desabilite, pelo menos, o operador de execução (que é controlado pela função `system_shell`):

```
disable_functions system_shell
```

Caso deseje informar mais funções, separe-as com vírgula na definição do parâmetro `disable_functions`.

Além disso, é necessário limitar o acesso do PHP aos arquivos do sistema. Por padrão, ele tem acesso a todos os arquivos do sistema (ao menos àqueles que o usuário do servidor Web tem acesso). Esse acesso pode ser

[1] Observe que isso deve ser feito em todos os arquivos de configuração do PHP (módulos para o Apache, executável preparado para CGI e para linha de comando). Algumas distribuições Linux adotam um arquivo de configuração único para todas as variações de PHP instaladas, mas algum sistemas, como o Debian e o Ubuntu, separam a configuração em vários arquivos (por exemplo, `/etc/php5/apache2/php.ini`, `/etc/php5/cgi/php.ini` e `/etc/php5/cli/php.ini`.

limitado pelo parâmetro `open_basedir`. Altere-o para conter apenas o diretório do servidor Web e o diretório temporário do sistema. Por exemplo, para limitar o acesso aos diretórios `/var/www` e `/tmp`, o `php.ini` seria alterado da seguinte forma:

```
open_basedir = "/var/www:/tmp"
```

Certifique-se que o parâmetro `register_globals` está desabilitado. Ele transforma, automaticamente, parâmetros recebidos por requisições HTTP em variáveis nos programas PHP. Eventualmente, para programas não bem escritos, isto pode ser explorado para executar operações indesejadas:

```
register_globals = Off
```

3.18. Configuração do Apache

A configuração do Apache, em termos de segurança, é relativamente simples. Ela consiste em restringir a execução de programas e impedir que o usuário altere essa configuração nos diretórios sob seu controle.

Encontre o arquivo de configuração do Apache. Ele geralmente possui o nome `httpd.conf`, presente no diretório `/etc/apache2/conf`. Algumas distribuições armazenam as configurações em vários arquivos. Por exemplo, no Ubuntu, as configurações a serem realizadas com as instruções desta seção estão no arquivo `/etc/apache2/sites-available/default`.

Identificado o arquivo de configuração do Apache, procure a diretiva de configuração do diretório raiz do seu servidor. O nome deste diretório está configurado no parâmetro `DocumentRoot` do arquivo de configuração. A maioria das distribuições Linux utilizam o diretório `/var/www`. Provavelmente será algo semelhante a isto:

```
<Directory /var/www>
    Options FollowSymLinks
    AllowOverride None
</Directory>
```

Todas as configurações a seguir devem ser realizadas no interior da configuração do diretório raiz. Primeiro, desabilite a listagem de diretórios, acrescentando a opção `-Indexes` ao parâmetro `Options`. Caso já exista o parâmetro `Options`, acrescente a nova opção ao final da linha, utilizando um espaço em branco como separador dos parâmetros. Caso já exista a opção, acrescente o sinal de `-` na frente da opção (substituindo o sinal `+` se o mesmo existir).

```
Options -Indexes
```

Desabilite a inclusão de código no lado do servidor (*server side includes*):

```
Options -Includes
```

Desabilite a execução de programas CGI:

```
Options -ExecCGI
```

Todas as opções acima desabilitaram a execução de programas. Agora resta impedir que o usuário altere tais configurações. Para isso, configure o parâmetro `AllowOverride` para `None`:

```
AllowOverride None
```

3.19. Desabilitar serviços desnecessários

Um último passo para a minimizar os riscos de segurança na máquina é a desabilitação de serviços desnecessários. Não existe uma receita do que deve ser ou não habilitado: depende de cada máquina e seus propósitos. A maioria das distribuições Linux utiliza um esquema denominado SYSV para a inicialização de serviços. Esse esquema estabelece níveis de execução, de 0 a 6:

- 0: Desligamento.
- 1: Mono-usuário.
- 2: Multi-usuário sem rede ou multi-usuário.
- 3: Multi-usuário.
- 4: Multi-usuário.
- 5: Multi-usuário com ambiente gráfico.
- 6: Reinialização.

Novamente, aqui depende da distribuição Linux. O Ubuntu, por exemplo, utiliza o nível 2 para sua execução normal (ambiente gráfico multi-usuários). Distribuições como Mandriva utilizam o nível 5. Observe o nível padrão de execução no arquivo `/etc/inittab`. Por exemplo, segundo o exemplo abaixo, nível de execução padrão é 2.

```
# The default runlevel.
id:2:initdefault:
```

Identificado o nível, acesse o diretório `/etc/rcN.d`, substituindo o `N` pelo número que identifica o nível. Nesse diretório existirão arquivos cujos nomes começam com a letra `S` ou `K`, seguidos de um número de dois dígitos. A letra `S` identifica os serviços que serão iniciados e a letra `K` os que serão encerrados ao adentrar no nível em questão. Os números determinam a ordem com que os serviços serão iniciados.

3.20. Considerações finais

As medidas especificadas neste capítulo fornecem um grau de segurança adequado para a maioria dos servidores de uso acadêmico, mas sempre será possível burlar as proteções impostas. O próximo capítulo trata da (incômoda) situação de detectar e diagnosticar futuras violações do sistema, de modo a acrescentar novas medidas a estas já adotadas.

Capítulo

4

Eventualidades

Inevitavelmente, uma falha de segurança será explorada no sistema. No momento em que esta eventualidade for detectada, deve-se agir para diagnosticar e isolar o problema, bem como providenciar a restauração do sistema.

A primeira medida de contingência, na verdade, sequer é de contingência: tenha sempre uma cópia de segurança dos dados. Foge ao escopo deste relatório a criação de tais cópias, mas é vital que elas existam: nem todas as falhas podem ser recuperadas (na verdade, a maioria não pode). A presença da cópia permite que todo o processo ao redor da eventualidade transcorra em maior tranqüilidade.

No entanto, não se deve assumir que, com a cópia de segurança em mãos, basta restaurá-la e continuar o uso do sistema. Se a falha que foi utilizada para comprometer o sistema não for detectada e corrigida, a chance dela ser novamente explorada é elevada.

Ao detectar uma eventualidade, contate o responsável pelos problemas de segurança em sua unidade de pesquisa. Eles dirão os procedimentos a serem tomados. Caso você não disponha desse instrumento, siga as instruções das próximas seções. Apesar delas serem superficiais e simples, elas são essenciais para qualquer processo de diagnóstico de falhas.

4.1. Preservar o sistema afetado

A primeira ação a ser tomada é isolar o sistema:

- Não desligue o computador. Caso isso seja feito, talvez a falha seja desativada (o que impede o seu diagnóstico). Também existe a chance da falha afetar o sistema de tal modo a evitar o reinício do sistema.
- Desconecte-o da rede.

4.2. Colete dados

Isolado o sistema, procede-se ao diagnóstico do problema. Para isso, é necessário coletar dados do estado do sistema. Anote todo e qualquer comando digitado, bem como a hora em que foi digitado. Tendo isso em mente, execute os seguintes passos:

- Anote a data e a hora do sistema. Isso é importante para determinar a validade dos dados armazenados nos registros.
- Verifique quais os processos que estão em execução. Procure processos suspeitos. Observe detalhes dos processos em execução, acessando os dados disponíveis no diretório `/proc`.
- Analise os arquivos de registro, configuração e inicialização do sistema:
  - `/var/log`
  - `/etc`
  - `/boot`

4.3. Analisar os dados coletados

A seguir, analise os dados em questão e procure anomalias:

- Identifique os programas que foram aparentemente afetados.
- Determine, mesmo que com uma grande margem de erro, a data e o horário em que a falha provavelmente ocorreu.
- Identifique as pessoas que utilizaram o sistema na data e horário em que a falha provavelmente ocorreu.
- Pergunte às pessoas identificadas se elas executaram algum programa que poderia causar o problema em questão.

Observe que o objetivo é encontrar a falha. Se uma pessoa foi responsável pelo problema, isso é algo a ser julgado pelas pessoas responsáveis pelo laboratório. Não se busca a incriminação dos usuários do sistema e sim a causa do problema.

4.4. Projetar uma solução para a falha

Avalie a falha encontrada (se encontrada) e estude soluções para a mesmo. Em especial, pense em:

- Como corrigir a falha.
- Como prevenir falhas do mesmo tipo.
- Como reduzir as conseqüências de falhas deste tipo.

Documente esse estudo. Talvez essas informações não lhe pareçam úteis nesse momento, mas, com o tempo, esse conhecimento, principalmente se compartilhado com os demais laboratórios, permitirá que todos os computadores da unidade de pesquisa fiquem mais seguros (e a segurança de um computador depende da segurança de todos os computadores próximos ao mesmo).

4.5. Implementar a solução

Finalmente, implemente a solução. Teste o sistema e certifique-se que a falha foi realmente sanada. Então, restaure as cópias de segurança e desfaça o isolamento da máquina (feito no início do tratamento da eventualidade).

Referências Bibliográficas

BAKER, Richard H. *The Computer Security Handbook*. 1. ed. [S.l.]: TAB Professional and Reference Books, 1985.

MOURANI, Gerhard; "MADDY", Madhu. *Securing and Optimizing Linux: RedHat Edition - A Hands on Guide*. Open Network Architecture, 2000. Disponível em: `http://www.faqs.org/docs/securing/index.html`.

PUSCHITZ, Werner. *Securing and Hardening Linux Production Systems: A Practical Guide to Basic Linux Security in Production Enterprise Environments*. 2006. Artigo.

SCHUMACHER, Markus. Ontology development. In: _____. [S.l.]: Springer, 2003. cap. Ontology Development, p. 179–184.

SCHUMACHER, Markus. *Security Engineering with Patterns*. [S.l.]: Springer, 2003. (Lecture Notes in Computer Science, v. 2754).

SCHUMACHER, Markus. Toward a securit core ontology. In: _____. [S.l.]: Springer, 2003. cap. Toward a Security Core Ontology, p. 87–96.

THE /proc filesystem documentation. [s.d.]. Disponível em: `http://linux.inet.hr/proc_sys_net_ipv4.html`.