

Sessão 10: Configuração segura de servidores Linux



As atividades desta sessão serão realizadas na máquina virtual *LinServer-G*.

Nesta seção iremos fazer uma série de configurações de segurança básica em um servidor Linux, especificamente a máquina *LinServer-G*. O estabelecimento de um *baseline* de segurança, como o que faremos aqui, é um passo importante na definição de uma fundação segura para a implantação de diferentes serviços de rede e, no caso da virtualização, de *templates* de máquinas virtuais.

1) Análise de *rootkits*

1. As ferramentas `chkrootkit` e `rkhunter` podem ser utilizadas para buscar por *rootkits* em um sistema Linux. *Rootkits*, como vimos na teoria, são conjuntos de programas de computador desenhados para permitir acesso continuado a área não-autorizadas de um sistema, usualmente com permissões elevadas.

Instale os pacotes `chkrootkit` e `rkhunter` na máquina *LinServer-G*, e verifique se existem *rootkits* instalados. Antes de executar o `rkhunter`, comente a linha `SCRIPTWHITELIST=/usr/bin/lwp-request` no arquivo `/etc/rkhunter.conf`.

```
# hostname  
LinServer-A
```

```
# apt-get install chkrootkit rkhunter
```

Execute o `chkrootkit`, e verifique seus resultados:

```
# chkrootkit  
  
(...)
```

Vamos comentar a linha solicitada pelo enunciado da atividade:

```
# sed -i 's/^(SCRIPTWHITELIST=\usr\bin\lwp-request\)/#\1/' /etc/rkhunter.conf
```

E, em seguida, rodar o `rkhunter`:

```
# rkhunter -c

(...)

System checks summary
=====

File properties checks...
  Required commands check failed
  Files checked: 139
  Suspect files: 0

Rootkit checks...
  Rootkits checked : 377
  Possible rootkits: 0

Applications checks...
  All checks skipped

The system checks took: 43 seconds

All results have been written to the log file: /var/log/rkhunter.log

One or more warnings have been found while checking the system.
Please check the log file (/var/log/rkhunter.log)
```

2) Inserção de senha no *bootloader*

O cuidado com a segurança física das máquinas deve ser amplo, indo desde o acesso à sala dos servidores até a adição de senha na BIOS dos sistemas (impedindo, por exemplo, alteração do dispositivo de *boot*).

Um aspecto que não pode ser esquecido é o *bootloader*, que faz a carga inicial do kernel—se desprotegido, um atacante com acesso físico à máquina pode utilizá-lo para alterar a senha do usuário **root** e ter acesso irrestrito ao sistema, dentre outras possibilidades.

O *bootloader* em uso pela grande maioria das distribuições Linux atualmente é o GRUB (*GRand Unified Bootloader*), e o Debian não é exceção. Vamos configurar uma senha de acesso ao GRUB para impedir que um atacante consiga ter acesso indevido ao sistema.

1. Usando o comando **grub-mkpasswd-pbkdf2**, gere um hash para a senha **rnpesr123**.

```
# echo -e 'rnpesr123\nrnpesr123' | grub-mkpasswd-pbkdf2 | awk  
'/grub.pbkdf2/{print$NF}'  
grub.pbkdf2.sha512.10000.D8258FF5554EB31945F1AB64026CFE276601804B0CFA82F14B0F61F9A9  
025ACA07C99BFF82F41912AAD897BA0DA9F0EB286FEB6E61332AEF2AB844952923FCB1.4CAC30EFD8FB  
C8070D52C52C5CA5A9C54A090881755EF9AE5A6B7077399B0641DE2E3B966A909F5F3A87A7FE0889492  
BF13FA2C017CDF54AB0025FB4BD92613E
```

2. Edite o arquivo `/etc/grub.d/40_custom` e insira o superusuário `admin`, com senha idêntica ao hash gerado no passo anterior.

```
# echo 'set superusers="admin"' >> /etc/grub.d/40_custom
```

```
# ghash="$( echo -e 'rnpesr123\nrnpesr123' | grub-mkpasswd-pbkdf2 | awk  
'/grub.pbkdf2/{p rint$NF}' )" ; echo "password_pbkdf2 admin ${ghash}" >>  
/etc/grub.d/40_custom ; unset ghash
```

```
# tail -n2 /etc/grub.d/40_custom  
set superusers="admin"  
password_pbkdf2 admin  
grub.pbkdf2.sha512.10000.5196BE001F91BB595600DC25D37F5F1448266CEF199431D9B80DA7ADF  
255685A43CCDE19C06DD132066DF945885D479A55E0A3BB8CFF6457B199606977BE85D.2E60E0918849  
65BF4BD86C736D95F3B3490BA906CF8E725F81D1FC3BD35A29B5799DDB6ED03CA661C3483974A57429E  
5DA6B253F9E3FA124A47B21ED1015C659
```

3. Reconfigure o GRUB com a nova combinação usuário/senha e reinicie a máquina. Verifique o funcionamento da sua configuração.

```
# grub-mkconfig -o /boot/grub/grub.cfg  
Generating grub configuration file ...  
Found linux image: /boot/vmlinuz-3.16.0-6-amd64  
Found initrd image: /boot/initrd.img-3.16.0-6-amd64  
Found linux image: /boot/vmlinuz-3.16.0-4-amd64  
Found initrd image: /boot/initrd.img-3.16.0-4-amd64  
done
```

```
# reboot
```

Após o *boot* da máquina, o menu do GRUB nos apresenta a possibilidade de editar a configuração apertando a tecla **e**:

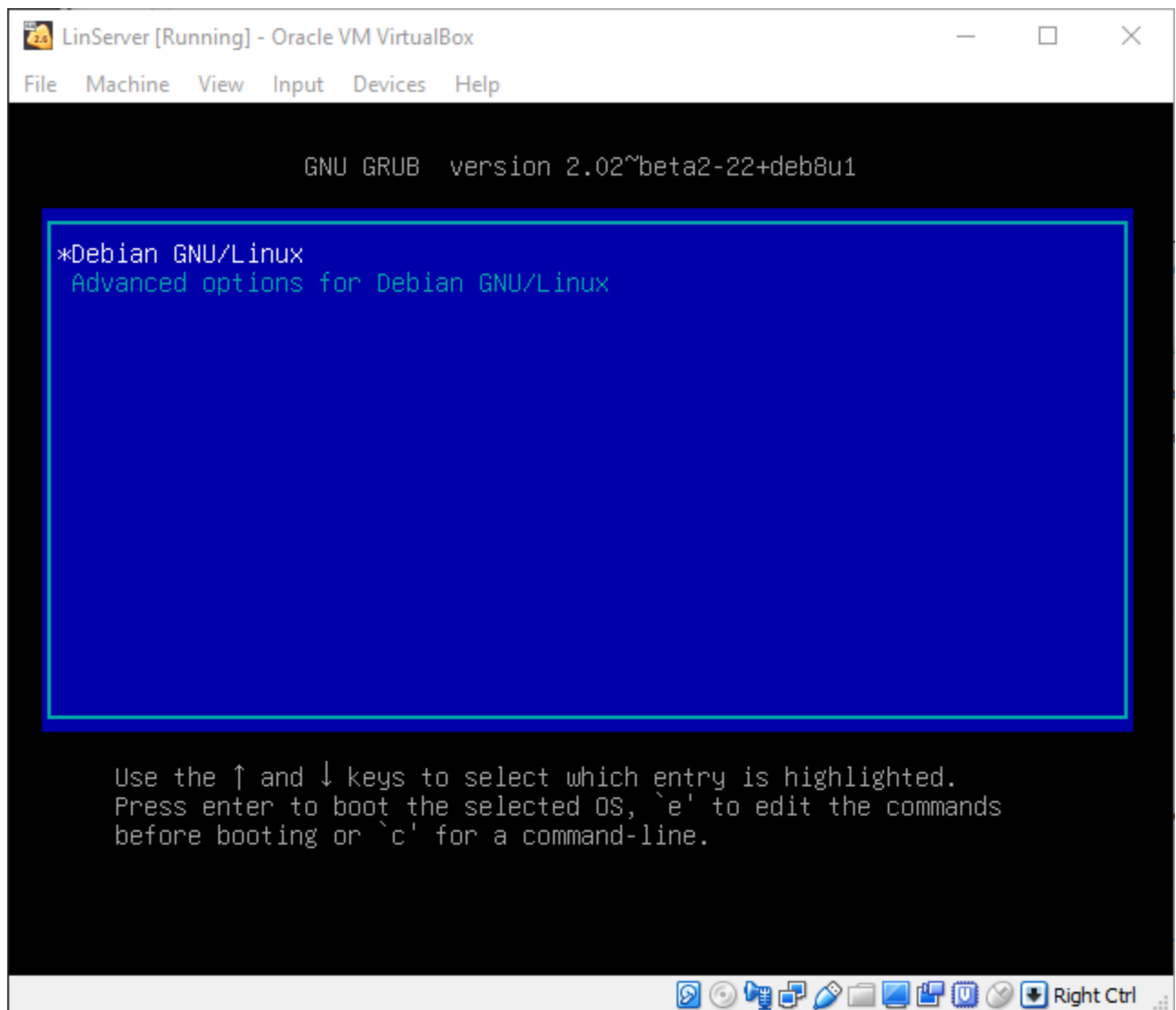


Figura 1. Edição de opções no GRUB

Apertando **e** sobre a primeira opção, imediatamente o sistema requisita a combinação usuário/senha configurada anteriormente:

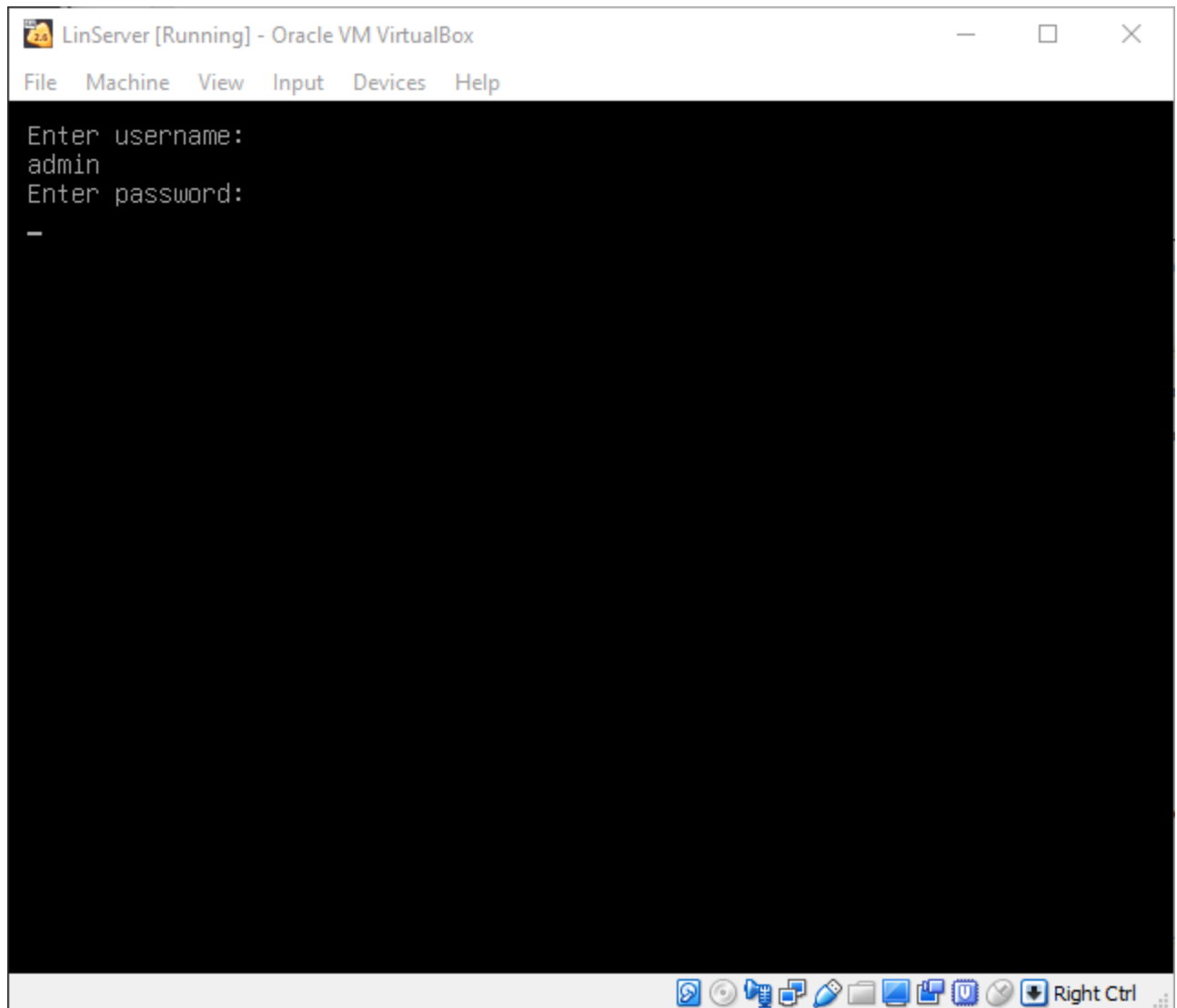


Figura 2. Inserção de usuário/senha no GRUB

Mediante a inserção da combinação correta, o menu de edição de opções de *boot* é mostrado, como se segue.

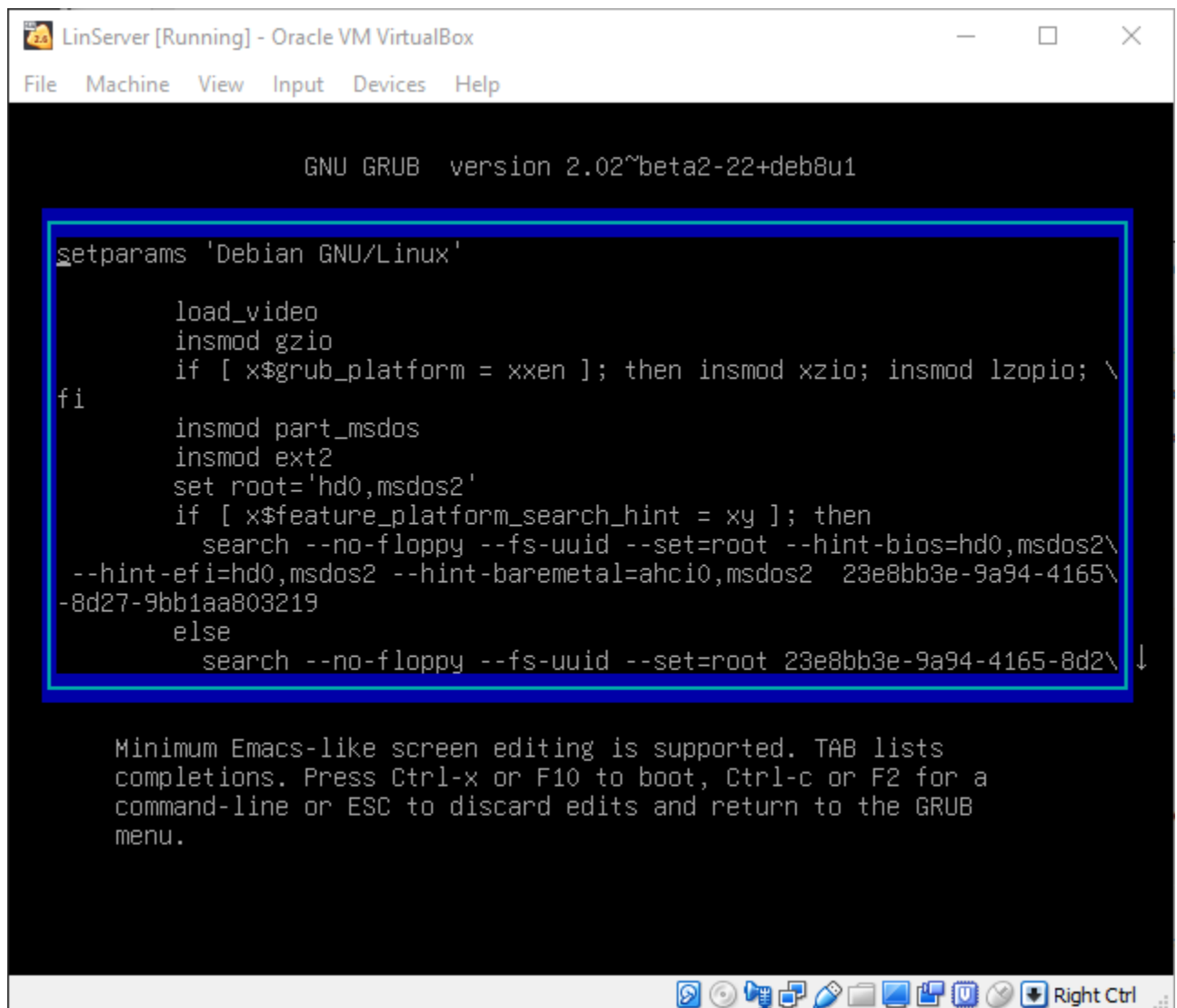


Figura 3. Edição de opções de boot no GRUB

Note, ainda, que mesmo o *boot* do sistema prossegue apenas se a combinação de usuário/senha correta é inserida no GRUB.

4. Edite a configuração do GRUB para que ele solicite senha **apenas** em caso de edição de entradas do menu, e que o *boot* normal do sistema prossiga sem que haja necessidade de interação.

Para conseguir o efeito desejado, é necessário editar o arquivo `/etc/grub.d/10_linux`. Na função `linux_entry()`, edite as duas linhas `echo "menuentry (···)"`, inserindo a flag `--unrestricted` antes da variável `${CLASS}`.

Vamos ver um antes/depois para ficar mais claro. Veja como estão as linhas 130-132 do arquivo `/etc/grub.d/10_linux` antes da edição:

Listagem 1. /etc/grub.d/10_linux

```
130     echo "menuentry '$(echo "$title" | grub_quote)' ${CLASS}"  
\$menuentry_id_option 'gnulinux-$version-$type-$boot_device_id' {" | sed  
"s/^/$submenu_indentation/"  
131     else  
132     echo "menuentry '$(echo "$os" | grub_quote)' ${CLASS}"  
\$menuentry_id_option 'gnulinux-simple-$boot_device_id' {" | sed  
"s/^/$submenu_indentation/"
```

Após a edição, elas devem ficar assim:

Listagem 2. /etc/grub.d/10_linux

```
130     echo "menuentry '$(echo "$title" | grub_quote)' --unrestricted ${CLASS}"  
\$menuentry_id_option 'gnulinux-$version-$type-$boot_device_id' {" | sed  
"s/^/$submenu_indentation/"  
131     else  
132     echo "menuentry '$(echo "$os" | grub_quote)' --unrestricted ${CLASS}"  
\$menuentry_id_option 'gnulinux-simple-$boot_device_id' {" | sed  
"s/^/$submenu_indentation/"
```

Note a adição da flag `--unrestricted` na antes de `${CLASS}` nas linhas 130 e 132.

Refaça a configuração do GRUB, reinicie a máquina e teste o funcionamento.

```
# grub-mkconfig -o /boot/grub/grub.cfg  
Generating grub configuration file ...  
Found linux image: /boot/vmlinuz-3.16.0-6-amd64  
Found initrd image: /boot/initrd.img-3.16.0-6-amd64  
Found linux image: /boot/vmlinuz-3.16.0-4-amd64  
Found initrd image: /boot/initrd.img-3.16.0-4-amd64  
done
```

```
# reboot
```

3) Remoção de serviços desnecessários

A remoção de serviços que não estão sendo utilizados em um servidor é premissa básica de segurança, pois reduz a superfície de ataque disponível para um agente malicioso. Deve-se fazer esse trabalho de forma diligente e constante, de forma a manter apenas aqueles serviços absolutamente necessários em operação.

1. Descubra quais serviços estão escutando por conexões TCP na máquina *LinServer-G*. Em seguida, faça o mesmo para o protocolo UDP.

Primeiro, vamos verificar os serviços TCP:

```
# netstat -tnlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
tcp        0      0 127.0.0.1:3306          0.0.0.0:*               LISTEN
854/mysql
tcp        0      0 0.0.0.0:22              0.0.0.0:*               LISTEN
416/sshd
tcp        0      0 127.0.0.1:25            0.0.0.0:*               LISTEN
1316/exim4
tcp6       0      0 :::80                   :::*                     LISTEN
513/apache2
tcp6       0      0 :::22                   :::*                     LISTEN
416/sshd
tcp6       0      0 :::1:25                  :::*                     LISTEN
1316/exim4
```

Depois, UDP:

```
# netstat -unlp
Active Internet connections (only servers)
Proto Recv-Q Send-Q Local Address           Foreign Address         State
PID/Program name
udp        0      0 0.0.0.0:514             0.0.0.0:*
374/syslog-ng
udp        0      0 172.16.1.10:123         0.0.0.0:*
427/ntpd
udp        0      0 127.0.0.1:123           0.0.0.0:*
427/ntpd
udp        0      0 0.0.0.0:123             0.0.0.0:*
427/ntpd
udp6       0      0 fe80::a00:27ff:fec1:123 :::*
427/ntpd
udp6       0      0 :::1:123                 :::*
427/ntpd
udp6       0      0 :::123                   :::*
427/ntpd
```

- Usando o comando **lsof**, descubra mais detalhes sobre o processo escutando na porta 25/TCP.

```
# lsof -i tcp:25 -n
COMMAND  PID   USER   FD  TYPE DEVICE SIZE/OFF NODE NAME
exim4    1316 Debian-exim 4u  IPv4 11627    0t0  TCP 127.0.0.1:smtp (LISTEN)
exim4    1316 Debian-exim 5u  IPv6 11628    0t0  TCP [::1]:smtp (LISTEN)
```


3. Descubra o nome do pacote escutando na porta 25/TCP. Em seguida, remova-o juntamente com seus arquivos de configuração.

```
# dpkg -l | grep exim
ii exim4                        4.84.2-2+deb8u5      all
metapackage to ease Exim MTA (v4) installation
ii exim4-base                  4.84.2-2+deb8u5      amd64
support files for all Exim MTA (v4) packages
ii exim4-config                4.84.2-2+deb8u5      all
configuration for the Exim MTA (v4)
ii exim4-daemon-light          4.84.2-2+deb8u5      amd64
lightweight Exim MTA (v4) daemon
```

```
# apt-get purge exim4*
```

4. Verifique que a porta 25/TCP não está mais na lista de *sockets* em estado LISTEN.

```
# netstat -tnlp | grep ':25 '
```

4) Controle granular de acesso a comandos

O **sudo** é uma importante ferramenta no controle de permissionamento em sistemas Linux. Ele permite que um usuário execute comandos como outro usuário do sistema, mas apenas aqueles previamente autorizados pelo usuário **root**. Dessa forma, pode-se permitir controle parcial do sistema a um colaborador, sem que ele tenha que ter acesso irrestrito à conta de superusuário.

1. Instale o pacote **sudo**, e verifique sua configuração padrão.

```
# apt-get install sudo
```

```
# visudo
```

2. Adicione o usuário **aluno** ao grupo **sudo**, e verifique quais comandos ele pode utilizar a partir de então. Adicionalmente, faça com que não seja necessário digitar senha para executar comandos privilegiados.

```
# adduser aluno sudo
Adding user `aluno' to group `sudo' ...
Adding user aluno to group sudo
Done.
```

```
# su - aluno
```

```
$ whoami  
aluno
```

```
$ sudo -l
```

We trust you have received the usual lecture from the local System Administrator. It usually boils down to these three things:

- #1) Respect the privacy of others.
- #2) Think before you type.
- #3) With great power comes great responsibility.

[sudo] password for aluno:

Matching Defaults entries for aluno on LinServer-A:

env_reset, mail_badpass,
secure_path=/usr/local/sbin\:/usr/local/bin\:/usr/sbin\:/usr/bin\:/sbin\:/bin

User aluno may run the following commands on LinServer-A:
(ALL : ALL) ALL

```
$ sudo visudo  
(...)
```

```
$ sudo cat /etc/sudoers | grep authenticate  
Defaults      !authenticate
```

3. Suponha que um novo colaborador, **mcfly**, acaba de entrar em seu setor e ficou responsável pela edição das regras de firewall dos servidores.
- a. Crie um novo usuário para esse colaborador, e configure sua senha como **rnpesr**.
 - b. Edite as regras de **sudo** para que ele possa editar as regras de firewall da máquina *LinServer-G* como o usuário **root**, e apenas isso.
 - c. Teste sua configuração.

Vamos primeiramente adicionar o usuário e configurar sua senha.

```
# useradd -m -s /bin/bash mcfly
```

```
# echo 'mcfly:rnpesr' | chpasswd
```

Em seguida, vamos editar o arquivo `/etc/sudoers` (para mais detalhes sobre sua sintaxe, consulte `man 5 sudoers`) e dar permissão para o usuário `mcfly` executar o comando `/sbin/iptables`.

```
# visudo  
(...)
```

```
# grep '^mcfly ' /etc/sudoers  
mcfly    ALL=(root) /sbin/iptables
```

Finalmente, vamos testar a configuração.

```
# su - mcfly
```

```
$ whoami  
mcfly
```

```
$ sudo iptables -L  
Chain INPUT (policy ACCEPT)  
target      prot opt source                destination  
  
Chain FORWARD (policy ACCEPT)  
target      prot opt source                destination  
  
Chain OUTPUT (policy ACCEPT)  
target      prot opt source                destination
```

```
$ sudo reboot  
Sorry, user mcfly is not allowed to execute '/sbin/reboot' as root on LinServer-A.
```

5) Controle de uso do binário `su`

Por padrão, através do comando `su` o Linux permite que qualquer usuário possa se tornar o superusuário `root`, se a senha correta for digitada. Para evitar esse comportamento, temos duas opções básicas:

- Desabilitar a conta do usuário `root`, e controlar o acesso a comandos através do `sudo`, como fizemos na atividade (4), ou

- b. Implementar um grupo especial, **wheel**, e permitir que apenas membros desse grupo possam utilizar o binário **su**.

Vamos testar esse segundo controle.

1. Crie um novo usuário, **docbrown** com senha **rnpesr**, e também um novo grupo de sistema, **wheel**. Adicione o novo usuário a esse grupo e edite o arquivo **/etc/pam.d/su** e implemente o controle de acesso ao binário **su**. Teste sua configuração.

Vamos criar o usuário, grupo, e fazer a adição solicitada.

```
# useradd -m -s /bin/bash docbrown
# echo 'docbrown:rnpesr' | chpasswd
```

```
# groupadd -r wheel
```

```
# adduser docbrown wheel
Adding user 'docbrown' to group 'wheel' ...
Adding user docbrown to group wheel
Done.
```

A seguir, vamos editar o arquivo **/etc/pam.d/su** e restringir o uso do **su** ao membros do grupo **wheel**.

```
# nano /etc/pam.d/su
(...)
```

```
# grep '^auth *required *pam_wheel.so' /etc/pam.d/su
auth      required    pam_wheel.so
```

Vamos testar a configuração com um usuário que não é membro desse grupo, como o **aluno**.

```
# su - aluno
```

```
$ whoami
aluno
```

```
$ su -
Password:
su: Permission denied
```

```
$ logout
```

Agora, vamos testar com um membro do grupo, como o usuário **docbrown** criado anteriormente.

```
# su - docbrown
```

```
$ whoami  
docbrown
```

```
$ su -  
Password:
```

```
# whoami  
root
```

6) Controle de acesso à console do sistema

Agora vamos restringir a quantidade de usuários que podem autenticar no console da máquina. Para tal, vamos configurar o módulo **pam_access** nos principais sistemas de autenticação: **ssh**, console texto, console gráfico (se instalado) e, opcionalmente, para os demais subsistemas.

1. Habilite o módulo **pam_access** para logins **ssh**, editando o arquivo **/etc/pam.d/sshd**.

Basta descomentar a linha **account required pam_access.so**, como mostrado a seguir.

```
# nano /etc/pam.d/sshd  
(...)
```

```
# grep '^account *required *pam_access.so' /etc/pam.d/sshd  
account required pam_access.so
```

2. Habilite o módulo **pam_access** para logins em console texto, editando o arquivo **/etc/pam.d/login**.

Basta descomentar a linha **account required pam_access.so**, como mostrado a seguir.

```
# nano /etc/pam.d/login  
(...)
```

```
# grep '^account *required *pam_access.so' /etc/pam.d/login
account    required      pam_access.so
```

3. Edite o arquivo `/etc/security/access.conf` e restrinja o acesso à console local e logins `ssh` apenas para membros do grupo `wheel` que efetuem login local ou logins remotos oriundos da rede 172.16.0/24, especificamente. Teste sua configuração.

Vamos editar o arquivo `/etc/security/access.conf`:

```
# nano /etc/security/access.conf
(...)
```

```
# grep -v '^#' /etc/security/access.conf
+ : wheel : LOCAL 172.16.1.0/24
- : ALL : ALL
```

Vamos monitorar o arquivo `/var/log/auth.log`, e testar alguns cenários. Primeiro, vamos tentar um login via console texto usando o usuário `aluno`. Temos os seguintes eventos registrados:

```
Sep  8 11:16:07 LinServer-A login[418]: pam_access(login:account): access denied
for user `aluno' from `tty1'
Sep  8 11:16:07 LinServer-A login[418]: Permission denied
```

Ao tentar login na console texto com o usuário `docbrown`:

```
Sep  8 11:17:18 LinServer-A login[3787]: pam_unix(login:session): session opened
for user docbrown by LOGIN(uid=0)
```

Vamos testar um login remoto via `ssh` usando o usuário `aluno` a partir da máquina `FWGW1-G`:

```
Sep  8 11:18:34 LinServer-A sshd[3818]: pam_access(sshd:account): access denied for
user `aluno' from `172.16.1.1'
Sep  8 11:18:34 LinServer-A sshd[3818]: fatal: Access denied for user aluno by PAM
account configuration [preauth]
```

A configuração sequer permite que tentemos digitar a senha do usuário — o bloqueio é feito antes disso. A seguir, vamos testar um login remoto via `ssh` usando o usuário `docbrown`, porém a partir da máquina `WinClient-G`, que está na faixa de IP incorreta:

```
Sep  8 11:23:10 LinServer-A sshd[3891]: pam_access(sshd:account): access denied for
user 'docbrown' from '10.1.1.10'
Sep  8 11:23:10 LinServer-A sshd[3889]: error: PAM: User account has expired for
docbrown from 10.1.1.10
```

Finalmente, vamos testar o cenário de login remoto via **ssh** usando o usuário **docbrown** a partir da máquina *FWGW1-G*:

```
Sep  8 11:32:18 LinServer-A sshd[4027]: Accepted keyboard-interactive/pam for
docbrown from 172.16.1.1 port 34663 ssh2
Sep  8 11:32:18 LinServer-A sshd[4027]: pam_unix(sshd:session): session opened for
user docbrown by (uid=0)
```

Note, portanto, que foram autorizados os logins apenas nos casos em que:

- a. O usuário era membro do grupo **wheel** e o login era feito localmente, ou
- b. O usuário era membro do grupo **wheel** e o login era feito remotamente a partir da faixa de IPs autorizada.

4. Reverta as configurações realizadas nesta atividade.

Basta comentar as linhas inseridas nos passos (1), (2) e (3) desta atividade.

7) Exigência de parâmetros mínimos de senha

O uso de senhas fortes é um requisito de segurança básico em sistemas computacionais; em servidores, especialmente, o descuido com senhas pode ocasionar falhas de segurança graves. As bibliotecas **pwquality** e **pwhistory** possibilitam a checagem da qualidade das senhas dos usuários, impondo requisitos mínimos em termos de tamanho e complexidade, bem como a manutenção de histórico de senhas

1. Instale os pacotes **libpam-modules** e **libpam-pwquality**, e configure o sistema para que novas senhas tenham os seguintes requisitos mínimos:
 - Tamanho mínimo de 10 caracteres.
 - Ao menos uma letra maiúscula.
 - Ao menos um caractere numérico.
 - Ao menos um caractere especial.
 - As últimas seis senhas não possam ser repetidas.

Primeiro, vamos instalar os pacotes solicitados:

```
# apt-get install libpam-modules libpam-pwquality
```

Para impor os requisitos de qualidade de senha, basta editar o arquivo `/etc/security/pwquality.conf`:

```
# nano /etc/security/pwquality.conf
(...)
```

```
# grep -v '^#' /etc/security/pwquality.conf
minlen = 10
dcredit = -1
ucredit = -1
ocredit = -1
```

Para impor os requisitos de não-repetição de senha, é necessário incluir uma linha para o módulo `pam_pwhistory.so` no arquivo `/etc/pam.d/common-password`:

```
# nano /etc/pam.d/common-password
(...)
```

```
# grep 'pam_pwhistory.so' /etc/pam.d/common-password
password      requisite                                pam_pwhistory.so retry=3 remember=6
use_authtok
```

Especificamente, essa linha pode ficar após a checagem de qualidade de senha (via módulo `pam_pwquality.so`) e antes dos módulos padrão do sistema, como mostrado a seguir:

```
# grep -v '^#' /etc/pam.d/common-password | sed '/^$/d'
password      requisite                                pam_pwquality.so retry=3
password      requisite                                pam_pwhistory.so retry=3 remember=6
use_authtok
password      [success=1 default=ignore]               pam_unix.so obscure use_authtok
try_first_pass sha512
password      requisite                                pam_deny.so
password      required                                  pam_permit.so
```

2. Teste suas configurações. Tente alterar a senha de um usuário não-privilegiado sem respeitar os requisitos mínimos de qualidade estabelecidos. Depois, tente reutilizar senhas e verifique o comportamento do sistema.

Vamos tentar alterar a senha do usuário `mcfly`. Progressivamente, vamos tentar a senha `teste`, depois `teste1` e finalmente `Teste1`:

```
$ whoami
mcfly
```



```
$ passwd
Changing password for mcfly.
(current) UNIX password:
New password:
BAD PASSWORD: The password contains less than 1 digits
New password:
BAD PASSWORD: The password contains less than 1 uppercase letters
New password:
BAD PASSWORD: The password contains less than 1 non-alphanumeric characters
passwd: Have exhausted maximum number of retries for service
passwd: password unchanged
```

Veja que o sistema reclamou nas três ocasiões, pois é necessário que a senha possua mais do que 10 caracteres, e pelo menos um caractere numérico, letra maiúscula e caractere especial. Finalmente, decidimos alterar a senha do usuário **mcfly** para **Testes123!**.

Agora, vamos tentar alterar a senha para o mesmo valor, e depois para **Testes234!** e **Testes345!**:

```
$ passwd
Changing password for mcfly.
(current) UNIX password:
New password:
BAD PASSWORD: The password is the same as the old one
New password:
BAD PASSWORD: The password is too similar to the old one
New password:
BAD PASSWORD: The password is too similar to the old one
passwd: Have exhausted maximum number of retries for service
passwd: password unchanged
```

Nos três casos, o sistema detectou que a senha ou era idêntica à original, ou que era muito parecida. Finalmente, decidimos alterar a senha do usuário **mcfly** para **Dufus!456**.

Vamos agora tentar reutilizar a senha antiga, **Testes123!**:

```
$ passwd
Changing password for mcfly.
(current) UNIX password:
New password:
Retype new password:
Password has been already used. Choose another.
passwd: Authentication token manipulation error
passwd: password unchanged
```

O sistema reporta que a senha já havia sido utilizada anteriormente. Onde esse registro fica mantido? No arquivo **/etc/security/opasswd**, como visto abaixo:

```
# whoami  
root
```

```
# cat /etc/security/opasswd  
mcfly:1001:3:$6$hJ8atdxg$cHkLXvRIRs2VDqeJQ12iNSlteeGEXdMqwI2odVx2n3X0gGGjLKfhmf/Bx8  
dzkdDpVafT5z0LZNXPfwgcOvemD1,$6$yCY08kaZ$XKKQwStnM3P6EnNMst9BX1r2aj5lHNVm2dSXAI16Xo  
W.9ypTch/q4eGhBI6w9JEwkAtnpN7xQCFxqp54pWkKs0,$6$EbARjVI9$u.mPGxn.ibveMlSGSumiiy0/6U  
TYS1d0YAN4mgSZ7JAv.Jj0XP7X8XRDvX7XuB1PEGWt5C0w4ZwuQXQmoKlpW1
```

8) Controle de logoff automático

A opção de logoff automático evita o uso indevido da sessão de um administrador quando este, inadvertidamente, não faz o logoff manual. A variável **\$TMOUT** do *shell* controla, em segundos, o tempo máximo aceito pelo sistema sem que o usuário execute um comando ou aperte uma tecla. Decorrido esse tempo, a máquina vai, automaticamente, efetuar o logoff do usuário.

1. Edite o arquivo **/etc/profile** e ative o logoff automático de usuários para dez segundos. Teste sua configuração.

```
# tail -n1 /etc/profile  
export TMOUT=10
```

```
# su - aluno
```

```
$ whoami  
aluno
```

```
$ timed out waiting for input: auto-logout  
# whoami  
root
```

9) Desabilitando a combinação de teclas CTRL + ALT + DEL

1. Para evitar que o servidor Linux seja reiniciado quando o seu teclado for confundido com o de um servidor Windows, desabilite a combinação de teclas CTRL + ALT + DEL.

```
# ls -ld /lib/systemd/system/ctrl-alt-del.target
lrwxrwxrwx 1 root root 13 Apr  8 2017 /lib/systemd/system/ctrl-alt-del.target ->
reboot.target
```

```
# systemctl mask ctrl-alt-del.target
Created symlink from /etc/systemd/system/ctrl-alt-del.target to /dev/null.
```

```
root@LinServer-A:~# systemctl daemon-reload
```

```
root@LinServer-A:~# ls -ld /etc/systemd/system/ctrl-alt-del.target
lrwxrwxrwx 1 root root 9 Sep  8 19:17 /etc/systemd/system/ctrl-alt-del.target ->
/dev/null
```