



# Curso 4449

## Linux Beginners in Cloud

Versão 2017\_4.0

# Linux Beginners in Cloud

## Objetivos da Aula

# Raio-X

- Chamado Dexter;
  - Realizar o Mapemanto do Servidor da Cloud;
  - Entender a Estrutura do ShellScript.



## Anotações:

## Chamado Dexter

### Problema #2

Acessar o servidor da Dexter na Cloud;

Realizar a coleta das seguintes informações: Distribuição do servidor, Memoria RAM, Processador, Espaço em Disco e Portas abertas.

Essas informações deverão ser armazenadas dentro de /home/tux/mapeamento/, em arquivos separados;

Compactar o diretório mapeamento em /home/tux/mapeamento.tar.gz;

Remover o diretório /home/tux/mapeamento;

O dono do arquivo /home/tux/mapeamento.tar.gz deverá ser o tux.

3

4LINUX

### Chamados/Problemas

Para atender este chamado, o aluno deverá aprender os seguintes conceitos:

- Onde o sistema armazena informações de Hardware? E como obter essas informações.
- Como criar arquivos com as saídas de comandos.
- Entender a ferramenta tar e gzip.

### Anotações:

---

---

---

---

---

---

---

---

---

---

## Passo 1 – Acessando o WebServer

- Usaremos um cliente SSH para acessar o Servidor da Dexter que está na Cloud;
- Escolha um Cliente SSH de sua preferência:

**Servidor:** cloud.4linux.com.br

**Porta:** 22

**Login e Senha:** Credenciais do EaD

```
1# ssh -l email@dominio.com.br cloud.4linux.com.br
```

## Acessando o servidor na Cloud

### Instalando o termius no Google Chrome:

O termius é uma aplicação do Google Chrome para acessar servidores remotos utilizando o SSH. Para executar esta aula, você pode usar o Cliente SSH de sua preferência, caso não queira instalar nada no seu equipamento, pode seguir os passos abaixo:

- No navegador, acesse a seguinte URL: “chrome://extensions/”;
- Clique em “Obter mais extensões”;
- Busque por “Termius” ou “ssh” e selecione “+ USAR NO CHROME”.

Para acessar o aplicativo, basta acessar pelo navegador:  
chrome://apps  
Selecione Terminus

## Laboratório Dexter

### Socorro e agora?

**Pronto, já estou conectado. Mas onde pego esse monte de informações?**

O arquivo que possui informações da distribuição é o /etc/os-release:

```
1# less /etc/os-release  
2# cp /etc/os-release /home/tux/mapeamento/  
  
Já para obter informações de processos e recursos do equipamento,  
pode ser consultado o "/proc":  
3# less /proc/meminfo  
4# less /proc/cpuinfo
```

**Mas esse arquivo da memória mostra tudo em kByte!**

Sim, por isso temos comandos melhores para verificar esse recurso:

```
5# free -h  
6# free -h > /home/tux/mapeamento/memoria
```



5

4LINUX

### Mapeando o Sistema

O diretório /proc possui 3 tipos de conteúdo:

1. Subdiretórios de processos;
2. Arquivos Informativos;
3. Parâmetros alteráveis.

Para consultar as informações de processo, nós utilizamos ferramentas como ps ou top ao invés de investigar cada subdiretório dos processos.

Quanto aos arquivos informativos, também é possível pegar as informações utilizando outros binários, como o próprio top trás informações da memória e do processador.

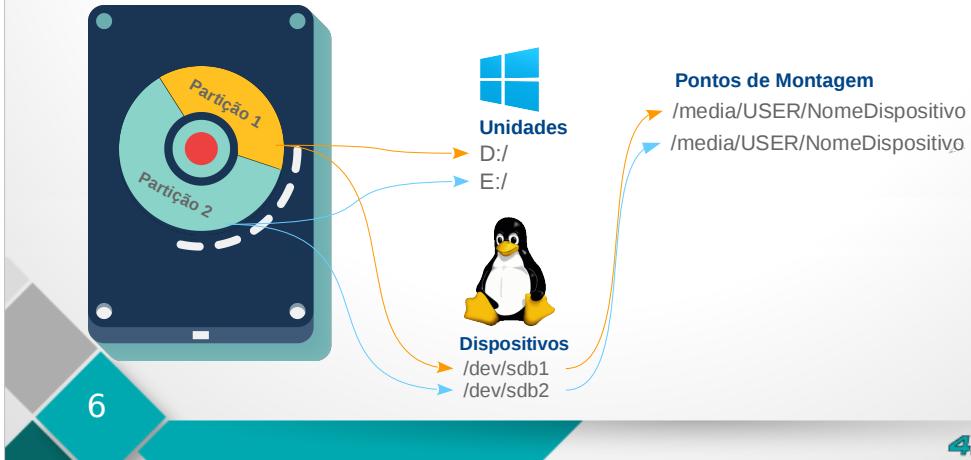
É muito comum termos que consultar informações do sistema utilizando comandos, normalmente a saída do comando trás informações que não são usuais para nós humanos, para isso muitos comandos possuem parâmetros como “-h” de Human, no caso do “free” ele sempre vai trazer o formato adaptado para Mega, Giga, Tera, etc.

# Laboratório Dexter

Certo certo... e onde fica o “C:” ou “D:” aqui no Linux?

Atenção:

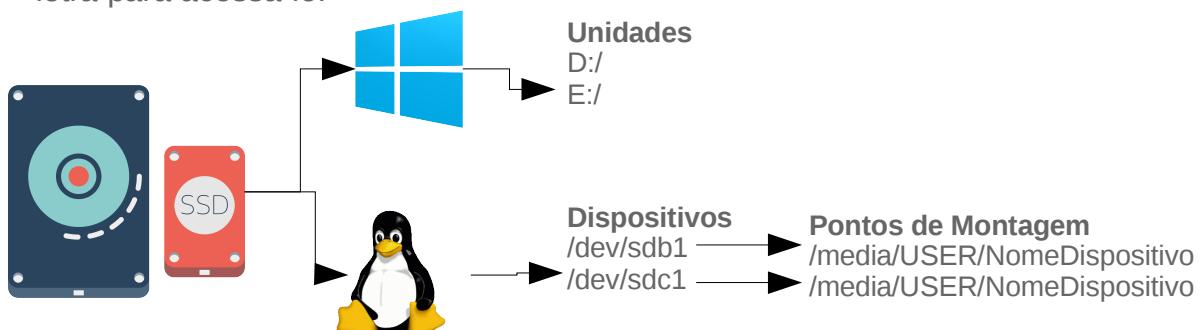
O Linux não possui Unidades, tudo é ponto de Montagem!



4LINUX

## Mapeamento de Servidores

Quando conectamos um dispositivo de bloco em um linux, ele é reconhecido e interpretado como um arquivo dentro de /dev, porém para esse dispositivo ser acessado é necessário que esteja montado em algum diretório, diferente do Windows que reconhece os dispositivos como uma unidade e atribuindo uma letra para acessá-lo.



Logo, no Linux é necessário acessar o ponto de montagem para poder visualizar os arquivos do disco. O comando abaixo lista todos os discos conectados no equipamento, mesmo os que não estão montados:

```
# lsblk
NAME          MAJ:MIN RM   SIZE RO TYPE MOUNTPOINT
sda            8:0    0 931,5G  0 disk
└─sda1         8:1    0   487M  0 part /boot
└─sda2         8:2    0     1K  0 part
└─sda5         8:5    0 931G  0 part
```

## Laboratório Dexter

**Agora eu entendi!**  
**Mas como eu pego o espaço livre em cada partição?**

Para ver todos os dispositivos montados em seu espaço livre, utilize o comando Disk Free:

```
1# df -h  
2# df -h > /home/tux/mapeamento/disco
```



### Mapeamento de Servidores

O comando df apresenta apenas o relatório dos dispositivos montado, e o parâmetro -h serve para apresentar os valores em formatos “humanos” como no comando free.

Caso exista a necessidade, você pode consultar o tamanho de determinado arquivo utilizando o comando “du”:

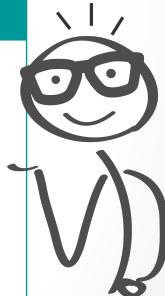
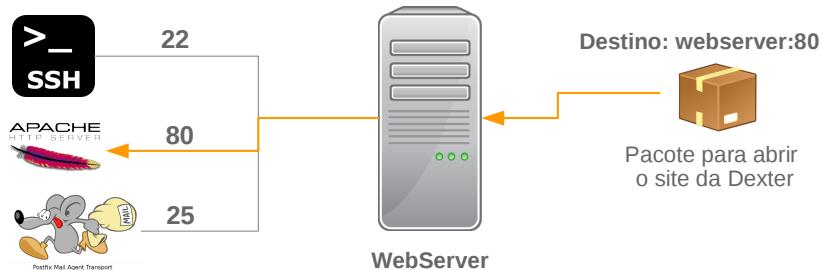
```
# du -hs /etc  
3.6M /etc/
```

O parametro -h tem o mesmo significado dos demais (human), já o parametro -s vem de summarizar, serve para ele exibir apenas o tamanho total de cada arquivo.

# Laboratório Dexter

## Espera! Para que serve essas portas ?

As portas servem para encaminhar os pacotes (tcp/udp) para os seus respectivos serviços:



Para listar todas as portas que estão sendo “ouvidas”, você pode executar:

```
# ss -nltpu  
# ss -nltpu > /home/tux/mapeamento/portas
```

8

4LINUX

## Mapemaneto de Servidores

O binário ss faz parte do pacote iproute2 (mesmo pacote que contém o binário ip), esse comando serve para apresentar e filtrar as portas que estão abertas e conexões que estão ativas no servidor no momento atual.

Os parâmetros mais utilizados são:

n → Para não resolver os nomes, quando o comando é executado sem este parâmetro ele tende a demorar mais, uma vez que ele irá tentar resolver cada IP para nome nos servidores DNS conhecidos;

l → Limita a saída do comando, mostrando apenas as portas que estão sendo ouvidas (listening), não apresentando as conexões ativas no momento;

t → Mostra as conexões TCP;

p → Exibe o ID do processo vinculado à esta conexão;

u → Mostra as conexões UDP.

Além do comando ss também é possível utilizar o netstat, porém este comando faz parte do pacote net-tools que vem sendo retirado das distribuições, os parâmetros utilizados são os mesmos:

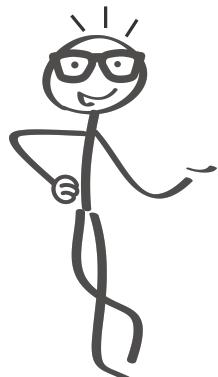
```
# ss -nltpu  
# netstat -nltpu
```

## Laboratório Dexter

### E agora como a gente usa o zip/winrar?

Para empacotar e compactar utilize:

```
1# tar -czf /home/tux/mapeamento.tar.gz /home/tux/mapeamento
```



4LINUX

9

### Compactação

No Linux é muito comum ver arquivos empacotados e compactados no formato “tar.gz”, diferente do windows a ferramenta tar serve para “empacotar” arquivos e podemos utilizar outras ferramentas para aplicar compressão como o “gz”, “xz”, “bzip”, etc...

O empacotador serve para unificar em apenas um arquivo diversos arquivos.

Para empacotar um diretório você pode utilizar o tar:

```
# tar -cf arquivo.tar /diretório/a/ser/compactado/
```

O parâmetro -cf significa respectivamente, “create” e “file”, logo sempre após o comando -f deve ser citado o arquivo que será criado.

Com o arquivo criado, já é possível compactá-lo utilizando, por exemplo, a ferramenta gzip:

```
# gzip arquivo.tar
```

Porém o comando tar possui parâmetros para manipular os arquivos utilizando os compressores, logo podemos criar o arquivo utilizando:

```
# tar -czf arquivo.tar.gz /diretório/a/ser/compactado/
```

Para descompactar arquivos com a extensão “.tar”, basta utilizar o parametro -X

```
# tar -xf arquivo.tar
```

# Laboratório Dexter

## Agora sim, vamos remover a pasta!

O comando utilizado para remover é o rm:

```
1# rm -rf /home/tux/mapemanto
```

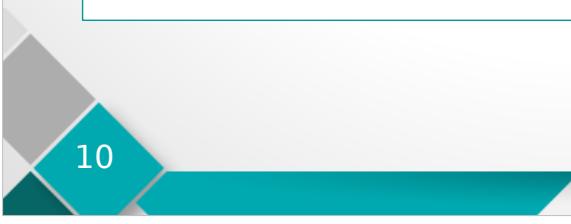
## Mas como eu sei quem é o dono do arquivo agora?

O comando ls trás esse tipo de detalhes, inclusive quem é o dono.

```
1# ls -l /home/tux/
```

Já para mudar o dono, basta usar o comando change owner.

```
1# chown tux /home/tux/mapemanto
```



10

4LINUX

## Removendo dados

O comando “rm” serve para remover arquivos ou diretórios. Os parâmetros apresentados no exemplo acima, servem para:

-r Remover de forma recursiva, logo se não for passado este parâmetro ele não remove diretórios.

-f Força a remoção.

## Listando com detalhes

Muitas vezes é importante vermos os detalhes dos arquivos de um diretório, para isso podemos usar o parâmetro -l do ls. Este parâmetro serve para listar com detalhes.

```
# ls -l /home/tux  
-rw-r--r-- 1 root root 675 Apr 21 12:36 mapeamento.tar.gz
```

Usuário  
Dono

Grupo  
Dono

## Trocando os donos

Além do comando chown para trocar o Dono do Arquivo, temos também o chgrp para trocar o Grupo Dono:

```
# chgrp tux /home/tux/mapeamento.tar.gz  
# ls -l /home/tux/mapeamento.tar.gz  
-rw-r--r-- 1 root root 675 Apr 21 12:36 mapeamento.tar.gz
```

# Laboratório Dexter



## Terminamos, agora é só relaxar!

Ainda não! Podemos automatizar a captura dessas informações utilizando um ShellScript simples.

Dê uma olhada em /root/mapeamento.sh

```
1# cat /root/mapeamento.sh
```

11

4LINUX

## ShellScript

O shellscript é uma maneira de automatizar determinadas tarefas repetitivas. Atualmente no mundo de DevOps, existem outras diversas ferramentas para este tipo de automatização, porém no caso do Shellscript ele basicamente executa os comandos que estão descritos nele.

```
root@webservercloud:/# cat /root/mapeamento.sh
#!/bin/bash
#Script para criar o mapemento do servidor cloud.4linux.com.br

#Cria o diretorio /home/tux/mapemaneto.
mkdir /home/tux/mapeamento

##
## Colhendo informacoes do sistema
##

cp /etc/os-release /home/tux/mapeamento/ #distribuicao do Sistema Operacional.
free -h > /home/tux/mapeamento/memoria #Memoria RAM
df -h > /home/tux/mapeamento/disco      #Espaco em disco livre.
ss -nltpu > /home/tux/mapeamento/portas #Portas/Servicos sendo executados.

#Compactar todo o conteudo
tar -czf /home/tux/mapeamento.tar.gz /home/tux/mapeamento

#Removendo o diretório de Origem
rm -rf /home/tux/mapeamento

#Trocando o dono do arquivo
chown tux /home/tux/mapeamento.tar.gz
```

## ShellScript

Dentro de um script em Shell, encontramos sempre na primeira linha, qual será o interpretador dos comandos apresentados no decorrer do documento. A sintaxe dele é “#!/bin/bash” porém, poderia ser chamado outro interpretador diferente do bash, como por exemplo o python, php, perl, etc... O nome dado a este campo é *shebang*.

Outra característica importante do ShellScript são os comentários, tudo que vier após o caractér “#” dentro do documento será sempre desconsiderado durante a execução do script. É deveras importante documentar dentro do script para quê ele serve, e o que os comandos estão fazendo. Sem está documentação o script funcionará normalmente, porém ao fazer uma manutenção no script, será sempre mais complicado, uma vez que não há comentários.

O shell suporta alguns argumentos de programação, logo para um melhor aproveitamento dele, é bom conhecer um pouco de lógica de programação. Ele suporta expressões como: for, while, if, case, function, etc...

Você pode executar um Shellscrip de diversas maneiras. Caso esteja no diretório onde está o script, basta executar:

```
# cd /root/  
# ./mapeamento.sh
```

Se o script não estiver no seu diretório, pode ser digitado o caminho absoluto do script:

```
# /root/mapeamento.sh
```

Também é possível especificar qual é o interpretador que irá executá-lo (neste caso não é necessário o script ter permissão de execução):

```
# bash /root/mapeamento.sh
```

### Anotações:

---

---

---

---

---

---

## Laboratório Dexter

### E por que você não me mostrou isso antes !?!

Calma! Ainda aprenderemos mais sobre ShellScript antes de usá-lo.

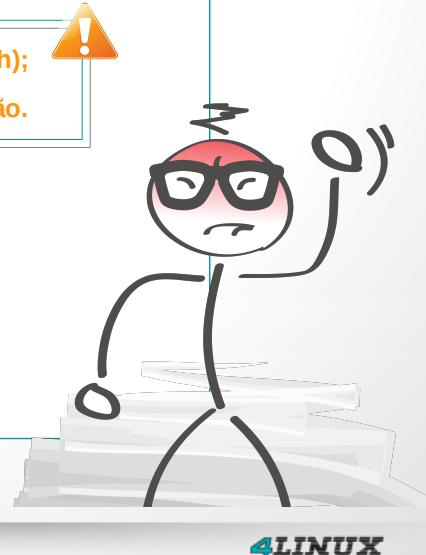
- 1 – Ele deve ter o shebang (`#!/bin/bash`);
- 2 – Ele deve ter permissão de execução.

Para verificar a permissão de execução:

```
1# ls -l /root/mapeamento.sh
```

Para executá-lo:

```
1# cd /root  
1# ./mapeamento.sh
```



13

4LINUX

## ShellScript

```
# ls -l /root/mapeamento.sh  
-rwxr-xr-x 1 root root 537 Apr 21 12:36 /root/mapeamento.sh
```

O primeiro campo apresenta o permissionamento do arquivo listado, ele é dividido da seguinte forma:

<b>rwx</b>	<b>r-x</b>	<b>r-x</b>
Usuário Dono	Usuários do Grupo Dono	Outros Usuários

Sendo que o significado de cada letra é:

r – read (leitura), representado por 4;

w – write (escrita), representado por 2;

x – execute (execução), representado por 1.

Por padrão, todos os arquivos que são criados no Linux, não recebem permissão de execução para aplicarmos. Para tal permissão basta executar:

```
# chmod +x /root/mapeamento.sh
```

Desta forma, é acrescentado o “x” de execute para todos os usuários. Neste caso, ele irá executar todos os comandos que vimos nessa aula de forma automática.

Também é possível aplicar o permissionamento de execução utilizando a forma octal, uma vez que sabemos que gostaríamos de aplicar, por exemplo quando o root cria um arquivo, por padrão ele recebe o permissionamento:

<b>rW-</b>	<b>r--</b>	<b>r--</b>
r = 4 w = 2	r = 4	r = 4

Neste caso este arquivo possui o permissionamento 644, se quisermos adicionar execução para o usuário dono, o grupo dono e os outros bastaria acrescentar 1 em cada campo:

```
# chmod 0755 /root/mapeamento.sh
```

### Permissionamento Especial

Além do permissionamento comum de leitura, escrita e gravação também existe o permissionamento especial que são utilizados para as finalidades abaixo:

**SUID bit** – Utilizado em binários, serve para herdar as permissões do dono do arquivo ao executar. Exemplo:

```
-rwsr-xr-x 1 root root 54192 Nov 20 2014 /usr/bin/passwd
```

O binário passwd deve ser executado por qualquer usuário, porém estes usuários não tem acesso ao arquivo /etc/shadow, logo eles precisam executar este binário com a herança do dono que é o “root”.

**SGID bit** – Usado em diretórios para que os novos arquivos herdem o mesmo grupo dono do diretório.

**STICKY bit** – Usado em diretórios para que apenas o dono dos arquivos possam deletar os mesmos.

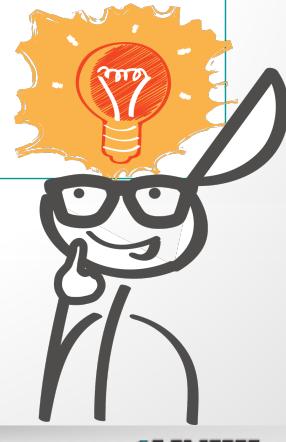
<b>rws</b>	<b>r-s</b>	<b>r-t</b>
SUID bit 4	SGID bit 2	STICKY bit 1

## Laboratório Dexter

### Hey! Eu consigo executar esse script como se fosse um comando?

Sim! Você pode criar um “alias” ou um “link”:

```
1# alias mapeamento="/root/mapeamento.sh"
1# mapeamento
Ou...
1# ln -s /root/mapeamento.sh /bin/mapeamento
1# mapeamento
```



15

4LINUX

### Transformando um shellscript em comando

Por padrão, algumas distribuições já possuem alguns alias criados, como por exemplo quando você utiliza um ls e ele apresenta a saída do comando colorido, pois possui um alias da seguinte forma:

```
# alias ls="ls --color=auto"
```

Porém, sempre que o equipamento for reiniciado ele perderá a configuração do alias. Para que seja carregado sempre que acessar o sistema, você deverá inserir esse comando nos arquivos que são carregados durante o login do usuário (~/.bashrc).

O comando ln, serve para criação de links. Usado junto com o parâmetro -s, é criado um link simbólico apenas, semelhante aos atalhos do windows.

```
# ln -s /root/mapeamento.sh /bin/mapeamento
```

Quando adicionado o atalho em /bin, qualquer usuário que possua o diretório /bin na sua PATH poderá utilizar o comando “mapeamento” que ele irá executar o script.

## Recapitulando...

# Raio-X

- Chamado Dexter;
  - Realizar o Mapemanto do Servidor da Cloud;
  - Entender a Estrutura do ShellScript.

## Anotações:

## Comandos aprendidos nesta aula

Comando	Função	Exemplo
lsblk	Lista todos os dispositivos conectados no sistema;	#lsblk
df	Exibe o espaço disponível em disco;	#df -h
du	Exibe o tamanho do arquivo ou diretório;	#du -hs /var/log/*
ss	Exibe as conexões ativas atualmente;	#ss -nltp
netstat	Exibe as conexões ativas atualmente;	#netstat -nltp
tar	Empacota e compacta determinado diretório ou arquivo;	#tar -czf logs.tgz /var/log
gzip	Compacta determinado arquivo;	#gzip arquivo
rm	Utilizado para remover arquivo;	#rm -rf /home/tux/mapeamento
chown	Altera o dono de determinado arquivo ou diretório;	#chown tux /home/tux
chgrp	Altera o grupo dono de determinado arquivo ou diretório;	#chgrp tux /home/tux
chmod	Altera o permissionamento de arquivos ou diretórios;	#chmod +x /root/mapeamento.sh
alias	Criar um “apelido” no Shell;	#alias ls="ls --color=auto"
ln	Cria um link.	#ln -s /root/mapeamento.sh /bin/mapeamento