# Linux para Infraestrutura

Prof. André Nasserala

nasserala@gmail.com

# GNU/Linux

- **O que é Linux?**
- Linux é o termo geralmente usado para designar qualquer sistema operacional que utilize o kernel Linux;
- **E que o que é kernel?**
- Entendido como o núcleo, ele representa a camada de software mais próxima do hardware, sendo responsável por gerenciar os recursos do sistema computacional como um todo.
- **Quem é o pai do Linux?**
- Foi desenvolvido por Linus Torvalds, inspirado no sistema Minix. O seu código fonte está disponível sob licença GPL para qualquer pessoa utilizar, estudar, modificar e distribuir de acordo com os termos da licença.

# Licença GNU

- A GPL é a licença com maior utilização por parte de projetos de software livre, em grande parte devido à sua adoção para o Linux.
- Em termos gerais, a GPL baseia-se em 4 liberdades:

1. A liberdade de executar o programa, para qualquer propósito;
2. A liberdade de estudar como o programa funciona e adaptá-lo para as suas necessidades;
3. A liberdade de redistribuir cópias de modo que você possa ajudar ao seu próximo;
4. A liberdade de aperfeiçoar o programa, e liberar os seus aperfeiçoamentos, de modo que toda a comunidade se beneficie deles.

# Distribuições Linux

- Sistema Operacional Linux ou GNU/Linux completo é uma coleção de software livre criados por indivíduos, grupos e organizações de todo o mundo, incluindo o núcleo/kernel Linux.
- Companhias como a Red Hat ou a SuSE, bem como projetos de comunidades como o Rocky Linux, Debian ou o Gentoo, compilam o software e fornecem um sistema completo, pronto para instalação e uso.  Exemplos de distribuições:
- Red Hat;
- Debian;
- Ubuntu;
- CentOS/
- Rocky Linux.

# Interfaces do Linux

- Os Interpretadores de comandos representam a interface de acesso no modo texto, ou CLI(Command Line Interface) do Linux, eles são a principal forma de controle remoto e gerenciamento do sistema.
- Exemplos de interfaces de linha comando, também chamados de Shell's ou interpretadores de comandos:
- Sh;
- Bash;
- Ssh.

# Login e Logout

- Em shell(CLI), temos a seguinte situação:

```
Fedora release 12 (Constantine)
Kernel 2.6.31.5-127.fc12.i686.PAE on an i686 (tty3)

localhost login: _
```

- Usuário Comum($):

```
[nasserala@cluster01 ~]$
[nasserala@cluster01 ~]$
[nasserala@cluster01 ~]$
[nasserala@cluster01 ~]$
```

- Super Usuário(#):

```
[nasserala@cluster01 ~]$
[nasserala@cluster01 ~]$
[nasserala@cluster01 ~]$
[nasserala@cluster01 ~]$ su -
Senha:
[root@cluster01 ~]#
```

# Desligar e Reiniciar

- Várias alternativas por shell:
- Comando: shutdown;
  - Opções:
  - -r : reinicia o computador;
  - -h : desliga o computador;
  - Com +10 : executa em 10 minutos;
  - Shutdown –r +10.
  - Shutdown –h now
- Comando: halt;
  - O comando halt apenas desliga o computador.
- Comando init(Níveis de 0 a 6):
  - Exemplo:
  - Init 6
  - Reinicia o computador
  - Init 0
  - Desliga o Computador
- Comando: reboot;
  - O comando reboot apenas reinicia a maquina.
- Teclas: Ctrl + Alt + Delete;
  - A seqüência apenas vale feita diretamente na console do sistema e apenas pode reiniciar o computador.

# Administração de Usuários

- Para criar um usuário de modo simples no Linux utilizamos os seguintes comandos:
- **adduser usuário**
- Cria o usuário
- **passwd usuário**
- Define senha do usuário
- Assim o usuário será criado, e por padrão já que não especificamos o diretório home, a shell, e nem o ID, esse usuário ficará em /home/usuario, o seu shell será o /bin/bash e um ID disponível foi criado para ele. Mas nós podemos específicar todos esses parâmetros com o seguinte comando:
- **adduser -m –c “Nome Completo”-d /home/usuario –s “/bin/sh” –g vasco usuario**

# Administração de Usuários

- Assim, estamos dizendo que:
- O diretório home do usuário será em /home/usuario,com o parâmetro –d;
- Que a shell padrão para esse usuário será o /bin/sh,com o parâmetro –s;
- Que ele será criado no grupo vasco, com o parâmetro –g;
- Deve ser criado a configuração do diretório de e-mail com –m;
- Com o parâmetro “–u” é possível definir o ID do usuário(Não recomendado):
- **adduser –u 501 nasserala**

# Administração de Usuários

- O adicionamento de usuários cria uma entrada para o usuário no arquivo /etc/passwd;
- Esta entrada tem o nome, a senha, UID, GID, Nome Completo , o diretório home do usuário e a shell padrão do usuário, todos separados com o sinal “: “.
- A sintaxe do arquivo, para cada linha, é a seguinte:
- **cat /etc/passwd**
- **nasserala:x:501:100:André Nasserala:/home/nasserala:/bin/bash**
- O comando cat lista o conteúdo de um arquivo.
- O x indica que as senhas estão criptografadas e salvas no arquivo(somente leitura) /etc/shadow.

# Administração de Usuários

- Para deletarmos um usuário utilizamos o seguinte comando:
- **userdel meu_user**
- **userdel -r meu_user**
- deleta o usuário meu_user junto com o diretório.
- Para criar um grupo digite o comando:
- **groupadd novo_grupo**
- para alterar o nome de um grupo ja criado:
- **groupmod nome_antigo –n novo_nome**
- O arquivo de configuração dos grupos fica em /etc/group.

# Administração de Usuários

- O comando usermod modifica as configurações de uma determinada conta de usuário.
- As opções mais usadas são:
- **-d diretório_home**
- **-c "nomedousuário”**
- **-g grupo**
- **-s shell**
- Exemplo:
- **usermod -c ”André Luiz Nasserala Pires” nasserala**

# Manipulando as Permissões

- O comando chmod pode ser usado para mudar os FLAGS 'rwx' dos arquivos e/ou diretórios, a sintaxe básica do comando é:
- **chmod [ugoa]{-+}[rwx] “nome do arquivo/diretório”**
- Então vamos à um exemplo. Se eu quero mudar a permissão para o grupo-dono do arquivo (g=group) poder ler e gravar (rw) no 'arquivo1.txt', faço o seguinte:
- **-rwx---r-- 1 killer users 1231 Mar 09 12:12 arquivo1.txt**
- **chmod g+rw arquivo1.txt**
- **-rwxrw-r-- 1 killer users 1231 Mar 09 12:12 arquivo1.txt**
- Caso você queira desfazer o comando, você faria: "chmod g-rw arquivo1.txt". Como se vê, o + ou - define se os FLAGS serão ativados ou desativados.

# Manipulando as Permissões

- Outros exemplos:
- **chmod a+r arquivo2.txt**
- Todos usuários (a=all) podem ler o arquivo2.txt
- **chmod o+w arquivo3.txt**
- Outros usuários (o=others) sem ser o dono e o grupo dono do arquivo, podem gravar no 'arquivo3.txt
- **chmod u-rwx arquivo4.txt**
- Retira as permissões de leitura, escrita e execução para o usuário dono do arquivo.

# Manipulando as Permissões

- O comando chmod pode também ser usado com números, em vez dos flags. Este método é chamado de octal, veja o exemplo abaixo:
- chmod 664 arquivo.txt
- O que quer dizer cada um desses números?
- Os números vão de 0 a 7, onde o primeiro se refere ao usuário dono, o segundo ao grupo dono e o terceiro aos outros usuários:

- 0 retira todas as permissões;
- 1 Atribui a permissão de execução;
- 2 Atribui a permissão de escrita;
- 3 Atribui as permissões de execução e escrita(1+2);
- 4 Atribui permissão de leitura;
- 5 Atribui a permissão de execução e leitura(1+4);
- 6 Atribui as permissões de leitura e escrita(2+4);
- 7 Atribui todas as pemissões(1+2+4).

# Manipulando as Permissões

- Assim o comando:
- **chmod 664 arquivo.txt**
- Dará permissão de Leitura e Escrita ao usuário dono(6), Leitura e Escrita ao grupo dono(6) e somente leitura aos demais usuários.
- **Obs**: O chmod na forma octal não é atributivo como na forma tradicional, assim se for preciso adicionar a permissão de escrita ao grupo, por exemplo, sendo que esse já tenha permissão de leitura, não podemos fazer assim:
- **chmod 020 arquivo**
- Pois assim as permissões do dono ficarão zeradas, do grupo dono apenas escrita e a dos demais zeradas.

# Manipulando as Permissões

- Exemplos:
- Atribua a permissão de leitura ao dono e remova todas as outras permissões do arquivo:
- **-rwxrw-r-- 1 killer users 1231 Mar 09 12:12 arquivo1.txt**
- **chomod 400 arquivo1.txt**
- **-r-------- 1 killer users 1231 Mar 09 12:12 arquivo1.txt**
- E se for necessário atribui a permissão de execução ao grupo dono e aos demais usuário sem alterar as permissões já atribuídas?
- **chmod 511 arquivo1.txt**
- **-r-x--x--x 1 killer users 1231 Mar 09 12:12 arquivo1.txt**

# Manipulando os Donos

- O comando chown é simples e pode ser usado para mudar o dono e o grupo dono de um arquivo/diretório. E é usado da seguinte maneira:
- **chown usuario:grupo “arquivo/diretorio”**
- Como exemplo, vamos definir que um arquivo 'teste1.txt' terá como dono killer' e como grupo 'users’:
- **chown killer:users teste1.txt**
- Outros exemplos(como –R o comando aplica a todo conteúdo de um diretório):
- **chown nasserala:visits ***
- **chown -R hugo:users arquivos**

# Estrutura de Diretórios

- Num sistema Linux geralmente temos uma estrutura de diretórios um pouco complexa;
- Os arquivos ali dispostos, a princípio, parecem estar jogados aleatoriamente nos diversos diretórios existentes;
- Mas, felizmente, há uma certa ordem e uma lógica fazendo com que cada diretório do sistema tenha uma finalidade específica;
- O sistema de arquivos mais importante em um sistema Linux é o sistema de arquivos raiz. Ele geralmente está montado no diretório “/”, também chamado de diretório raiz;
- Um sistema de arquivos raiz deve conter o necessário para suportar um sistema Linux completo, e para tanto deve incluir alguns requisito básicos.

# Estrutura de Diretórios

- Estrutura básica de diretórios :
- /bin
- /dev
- /home
- /proc
- /usr
- /boot
- /etc
- /lib
- /var
- /sbin

```
[root@localhost /]# ls -l
total 102
dr-xr-xr-x.   2 root root  4096 Set 10 18:04 bin
dr-xr-xr-x.   5 root root  1024 Set 10 18:06 boot
drwxr-xr-x.   2 root root  4096 Nov 11  2010 cgroup
drwxr-xr-x.  18 root root  3700 Set 10 18:08 dev
drwxr-xr-x. 107 root root 12288 Set 10 18:07 etc
drwxr-xr-x.   3 root root  4096 Set 10 18:07 home
dr-xr-xr-x.  11 root root  4096 Set 10 18:04 lib
dr-xr-xr-x.   9 root root 12288 Set 10 18:04 lib64
drwx------.   2 root root 16384 Set 10 17:56 lost+found
drwxr-xr-x.   2 root root  4096 Nov 11  2010 media
drwxr-xr-x.   2 root root     0 Set 10 18:07 misc
drwxr-xr-x.   2 root root  4096 Nov 11  2010 mnt
drwxr-xr-x.   2 root root     0 Set 10 18:07 net
drwxr-xr-x.   2 root root  4096 Nov 11  2010 opt
dr-xr-xr-x. 143 root root     0 Set 10 18:07 proc
dr-xr-x---.   5 root root  4096 Set 10 18:34 root
dr-xr-xr-x.   2 root root 12288 Set 10 18:04 sbin
drwxr-xr-x.   7 root root     0 Set 10 18:07 selinux
drwxr-xr-x.   2 root root  4096 Nov 11  2010 srv
drwxr-xr-x.  13 root root     0 Set 10 18:07 sys
drwxrwxrwt.  11 root root  4096 Set 10 18:34 tmp
drwxr-xr-x.  13 root root  4096 Set 10 17:57 usr
drwxr-xr-x.  22 root root  4096 Set 10 18:03 var
[root@localhost /]#
```

# Redirecionamentos

- Em Linux podemos redirecionar a saída/entrada de comandos/arquivos para comandos/arquivos:
- **Comando1 | Comando2**
- A saída gerado pelo comando 1 serve de entrada para o comando2;
- **Comando > Arquivo**
- A saída gerado pelo comando é usada para criar um Arquivo;
- **Comando < Arquivo**
- O Arquivo é usado como entrada para o comando;
- **Comando >> Arquivo**
- O saída do Comando é usada para incrementar o arquivo.

# Comando Cp

- Para copiar arquivos e diretórios usamos o comando cp para esse propósito.
- Sintaxe :
- **cp –opções origem destino**
- Principais opções:
- -f, Não pergunta, substitui todos os arquivos caso já exista;
- -R, Copia toda estrutura dos diretórios da origem para o destino;
- -v, Mostra os arquivos enquanto estão sendo copiados.
- **cp teste.txt /tmp**
- Copia o arquivo teste.txt para dentro do diretório /tmp.

# Comando Cp

- **cp * /tmp**
- Copia todos os arquivos do diretório atual para /tmp.
- **cp /bin/* .**
- Copia todos os arquivos do diretório /bin para o diretório em que nos encontramos no momento.
- **cp -R /bin /tmp**
- Copia o diretório /bin e todos os arquivos/sub-diretórios existentes para o diretório /tmp.
- **cp -R /bin/* /tmp**
- Copia todos os arquivos do diretório /bin (exceto o diretório /bin) e todos os arquivos/sub-diretórios existentes dentro dele para /tmp.

# Comando Mv

- Move ou renomeia arquivos e diretórios. O processo é semelhante ao do comando cp mas o arquivo de origem é apagado após o término da cópia.
- Sintaxe:
- **mv -opções origem destino**

- Opções:
- -f, Substitui o arquivo de destino sem perguntar.
- -i, Pergunta antes de substituir. É o padrão.
- -v, Mostra os arquivos que estão sendo movidos.

# Comando Mv

- Exemplos:
- **mv teste.txt teste1.txt**
- Muda o nome do arquivo teste.txt para teste1.txt.
- **mv teste.txt /tmp**
- Move o arquivo teste.txt para /tmp. Lembre-se que o arquivo de origem é apagado após ser movido.
- **mv teste.txt teste.new (supondo que teste.new já exista)**
- Copia o arquivo teste.txt por cima de teste.new e apaga teste.txt após terminar a cópia.

# Comando Rm

- Apaga arquivos. Também pode ser usado para apagar diretórios e sub-diretórios vazios ou que contenham arquivos.
- Sintaxe:
- **rm –opções dir1/arq1 dir2/arq2 ... Dirn/arqn**
- Opções:
- -v, Exibe os arquivos que são removidos;
- -r, Usado para remover arquivos em sub-diretórios. Esta opção também pode ser usada para remover sub-diretórios.
- -f, Remove os arquivos sem perguntar.

# Comando Rm

- Exemplos:
- **rm teste.txt**
- Apaga o arquivo teste.txt no diretório atual.
- **rm *.txt**
- Apaga todos os arquivos do diretório atual que terminam com .txt.
- **rm *.txt teste.novo**
- Apaga todos os arquivos do diretório atual que terminam com .txt e também o arquivo teste.novo.
- **rm -rf /tmp/teste/***
- Apaga todos os arquivos e sub-diretórios do diretório /tmp/teste mas mantém o sub-diretório /tmp/teste.

# Comando Mkdir

- Cria um diretório no sistema. Sintaxe:
- **mkdir –opções cam1/dir1 cam2/dir2 ... camn/dirn**
- Opções:
- -p: cria toda a raiz até a ultima subpasta.
- -v, Mostra uma mensagem para cada diretório criado. As mensagens de erro serão mostradas mesmo que esta opção não seja usada.
- Exemplo:
- **mkdir Documentos**
- Cria o diretório “Documentos”

# Comandos Rmdir e Touch

- Rmdir remove um diretório do sistema. O diretório a ser removido deve estar vazio e você deve ter permissão de gravação para remove-lo.
- **rmdir –opções dir1 dir2 ... dirn**
- **rmdir Documentos**

- Touch cria arquivos vazios.
- **touch arquivo**
- **touch novo_arquivo.txt**
- Cria um arquivo vazio de nome novo_arquivo.txt

# Comandos Clear e Date

- O comando clear limpa a tela e posiciona o cursor no canto superior esquerdo do vídeo.
- Exemplo:
- **clear**
- O comando date permite ver/modificar a Data e Hora do Sistema. Você precisa estar como usuário root para modificar a data e hora.
- **date**
- Ajustar a data para o dia 24 de março de 2021 e a hora para 18:30:
- **date 240318302021**
- **Date (mes)(dia)(hora)(minutos)(ano)**

# Comando Echo

- O comando echo serve para visualizar quaisquer das variáveis de ambiente, e ecoar/exibir textos.

- exemplo:
- **echo $RANDOM**
- A variável de ambiente $RANDOM exibe um número aleatório

- **echo “Alô Mundo!”**
- Exibe a mensagem “Alô mundo!” na tela do interpretador de comandos.

# Comando Cat

- O comando cat imprime na tela o conteúdo do arquivo.
- Sintaxe:
- **cat [opções] arquivo**
- **cat arquivo**
- **cat > arquivo**
- Este comando cria um arquivo recebendo o texto digitado logo após o comando. Para sair do arquivo criado utilize Ctrl + D.
- **cat arquivo1 >> arquivo2**
- Este comando faz com que o arquivo2 receba o conteúdo do arquivo1.
- **cat –n arquivo1**
- Este comando faz com que o arquivo1 seja listado com numeração de linhas.

# Comando Cut

- Ao pé da letra significa “cortar”. Ele lê o conteúdo de um ou mais arquivos e tem como saída uma coluna vertical.
- Sintaxe:
- **cut opções arquivo**
- Opções:
- **-d delimitador;**
- **-f número: Imprime a coluna número.**
- **cat etc/passwd |cut -d: -f1**
- Lista o arquivo passwd e cortar como delimitador “:” e lista a primeira coluna.

# Comando Head

- O comando head mostra as primeiras 10 linhas de um arquivo, como por exemplo:
- **head a.txt**
- Exibe as 10 primeiras linhas do arquivo a.txt;

- Quando usado com opção “-n” pode mostrar apenas as primeiros n linhas de um arquivo.
- **head –n 20 a.txt**
- Exibe as 20 primeiras linhas do arquivo a.txt.

# Comandos nl e od

- O comando nl, é utilizado para numerar as linhas de um arquivo.
- **cat /etc/passwd | nl**
- Lista o arquivo /etc/passwd e mostra a saída enumerada.

- O comando od é utilizado para visualizar o conteúdo de um arquivo nos formatos hexadecimal, octal e ASCII;
- As opções mais utilizadas são:
- -t tipo Os tipos disponíveis são(c : ASCII, o : Octal e x : Hexadecimal)
- **od -t x arquivo.txt**
- Exibe o conteúdo do arquivo “arquivo.txt” em hexadecimal.

# Comandos Split e Tac

- O comando split é usado para dividir grandes arquivos em n-arquivos menores. Os arquivos de saída são gerados de acordo com tamanho do arquivo de entrada.
- **split –20 arquivo_entrada.txt arquivo_saida.txt**
- Divide o arquivo arquivo_entrada.txt a cada 20 linhas e cria um novo arquivo_saida.txt.

- O comando tac é o oposto do comando cat que ja vimos. O comando tac exibe o conteúdo de um arquivo de trás para frente.
- Exemplo:
- **tac arquivo.txt**

# Comando Tail

- O comando tail é oposto ao comando head que ja vimos. Este comando exibe as 5 últimas linhas de um arquivo, diferente do comando head que exibe as 10 primeiras.
- As opções mais usadas são:
- -n número : Específica o número de linhas finais que o tail deverá exibir.
- -f : Mostra as últimas linhas finais de um arquivo continuamente, enquanto outro processo grava mais linhas.
- Exemplo:
- **tail -n 50 arquivo.txt**
- **tail -f /var/log/messages**

# Comando WC

- O comando wc faz a contagem em unidade do conteúdo de um arquivo, linhas, caracteres e palavras. Digitando o comando wc sem opção, o usuário terá como resultado o número de caracteres, palavras e linhas.
- As opções mais usadas são:
- -l : Exibe o número de linhas.
- -w : Exibe o número de palavras
- -c : Conta o número de caracteres de um ou mais arquivos.
- Exemplo:
- **wc arquivo.txt**
- **wc –w arquivo.txt**

# Comando Grep

- O comando grep, procura em um arquivo ou na entrada padrão linhas que coincidam com a string listada como argumento.
- Exemplo:
- **ls | grep log**
- Lista todos os arquivos do diretorio mas só exibe aqueles que contem em alguma parte do nome do arquivo a string log.
- **grep 192.168.0.1 /var/log/messages**
- Lista todas as linhas que fazem referencia no arquivo messages ao ip 192.168.0.1.

# Comando Grep

- Argumentos:
- -m(x)  Limita número de x resultados;
- -i  Ignora maiúsculas e minúsculas;
- -v  Lista tudo menos o argumento;
- -n  Exibe o numero do registro, da linha;
- -c  Exibe a contagem de resultados;
- **cat /etc/passwd | grep –v nasserala**
- Lista todas as linhas do arquivo /etc/passwd que não tenham a string “nasserala”.