



APRESENTAÇÃO

Clínica de Tecnologia da Informação e Comunicação

Sistemas Operacionais

Prof. MSc. Jhonatan Geremias
jhonatan.geremias@pucpr.br



Sistema de arquivo – Linux

1/2

- `/` - diretório raiz;
- `/root` - diretório local do superusuário (usuário root) ;
- `/bin` – diretório que armazena os arquivos binários de comandos essenciais do sistema e mais frequentemente utilizados;
- `/sbin` – arquivos essenciais, executáveis de administração do sistema;
- `/lib` – arquivos de bibliotecas compartilhados para o kernel e aplicativos;
- `/boot` – arquivos estáticos de boot/bootloader usados na inicialização;
- `/dev` – arquivos usados para acessar os dispositivos de entrada/saída;
- `/etc` – configuração do sistema da máquina local, como rede, som e vídeo.



Sistema de arquivo – Linux

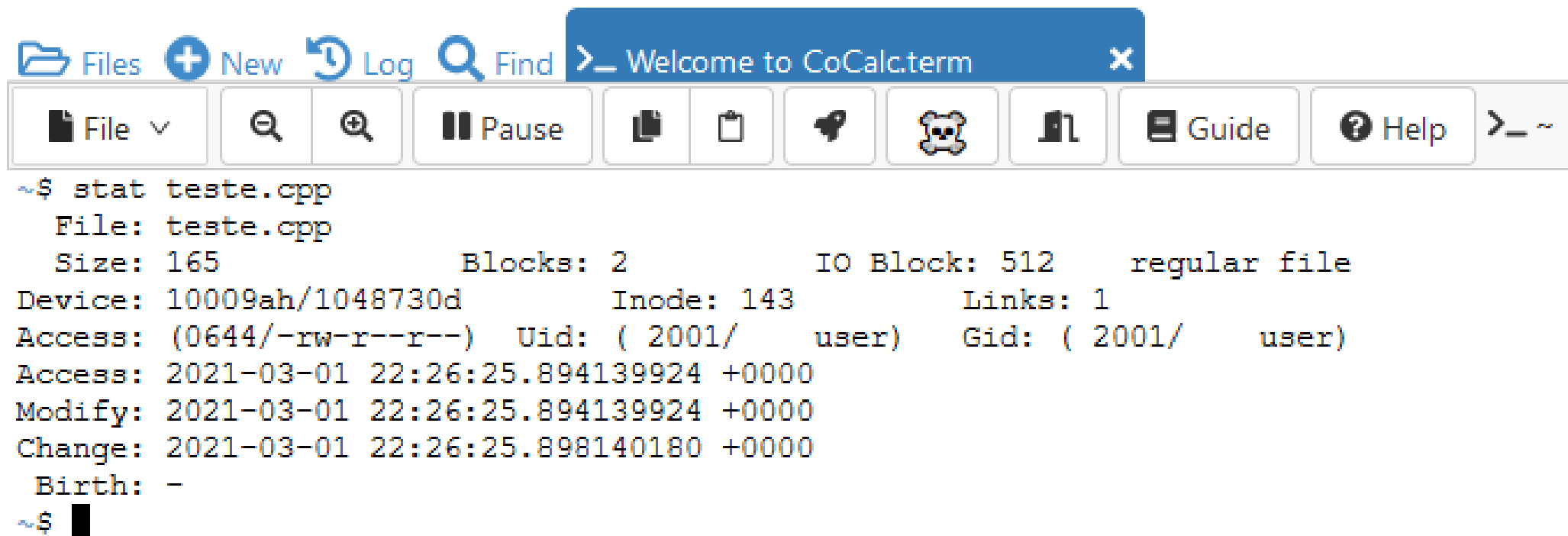
2/2

- `/home` – diretório local com os arquivos dos usuários;
- `/mnt` – ponto de montagem de partição que receberá o conteúdo da mídia que for montada;
- `/tmp` – arquivos temporários gerados por alguns utilitários;
- `/usr` – diretório de instalação de programas e aplicativos;
- `/var` – informações variáveis, contém a maior parte dos arquivos que são gravados com frequência pelos programas dos sistemas
 - Ex.: caches, spools, logs, e-mails;
- `/proc` – armazena e atualiza informações do sistema em tempo real;
 - Ex.: configuração de hardware, programas em execução, recurso de memória, dispositivos PCI e muito outros.



Comando para gerenciamento de arquivos

- `stat <filename>`: exibe informações gerais a respeito de um arquivo.



```
~$ stat teste.cpp
  File: teste.cpp
  Size: 165          Blocks: 2          IO Block: 512   regular file
Device: 10009ah/1048730d    Inode: 143          Links: 1
Access: (0644/-rw-r--r--)  Uid: ( 2001/      user)   Gid: ( 2001/      user)
Access: 2021-03-01 22:26:25.894139924 +0000
Modify: 2021-03-01 22:26:25.894139924 +0000
Change: 2021-03-01 22:26:25.898140180 +0000
  Birth: -
~$
```



Atividades Comando Stat

- Criar um arquivo “.cpp” com o comando `touch`;
- Editar o arquivo utilizando o `vim`;
- Criar um programa em C para imprimir um “Hello World!!”;
- Compilar o arquivo utilizando o `gcc`;
 - Definir a saída do comando utilizando a opção “-o” no `gcc`;
- Comparar o arquivo binário gerado com o arquivo “.cpp” utilizando o comando `stat`;
- Quais informações podemos obter a partir do comando `stat`.



Comandos de verificação - Autenticação

- `last`: informações sobre os últimos usuários que se logaram no sistema;
- `lastb`: apresenta a última tentativa malsucedida no sistema;
 - `cat /var/log/btmp` # informação armazenada no arquivo btmp;
- Acessar via putty colocando uma senha inválida.
 - Auditoria e Segurança;
 - Comando `w` é bom comando de segurança.



Configuração SSH

- Alterar a porta padrão do SSH (principalmente se o servidor tiver acesso externo);

Comandos:

```
sudo vim /etc/ssh/ssh_config
```

Mudar a **porta de 22** para uma outra maior, por ex.: **1022** ou outra porta diferente.

```
systemctl reload sshd
```

```
[root@localhost ~]# systemctl status sshd
● sshd.service - OpenSSH server daemon
   Loaded: loaded (/usr/lib/systemd/system/sshd.service; enabled; vendor preset: enable
   Active: active (running) since Fri 2021-08-20 10:39:42 EDT; 25min ago
     Docs: man:sshd(8)
           man:ssh_config(5)
   Main PID: 958 (sshd)
     Tasks: 1 (limit: 49468)
    Memory: 7.7M
    CGroup: /system.slice/sshd.service
            └─958 /usr/sbin/sshd -D -oCiphers=aes256-gcm@openssh.com,chacha20-poly1305@o

ago 20 10:39:42 localhost.localdomain systemd[1]: Starting OpenSSH server daemon...
ago 20 10:39:42 localhost.localdomain sshd[958]: Server listening on 0.0.0.0 port 22.
ago 20 10:39:42 localhost.localdomain sshd[958]: Server listening on :: port 22.
ago 20 10:39:42 localhost.localdomain systemd[1]: Started OpenSSH server daemon.
```



Configuração SSH – Acesso como root

- Desabilitar o login de root por SSH;

Comandos:

```
sudo vim /etc/ssh/ssh_config
```

Alterar a opção **PermitRootLogin** para **no**

```
systemctl reload sshd
```



Modo de Operação do Linux

- **runlevel**: nível do modo de operação do Linux;
 - 0 – halt;
 - 1 – monousuário;
 - 2 – modo multiusuário, sem NFS;
 - 3 – modo multiusuário completo;
 - 4 – não usado
 - 5 – X11
 - 6 – reboot



Comandos de verificação – Conta Usuário

- **chage**: mostrar informações sobre a tempo de expiração do usuário;

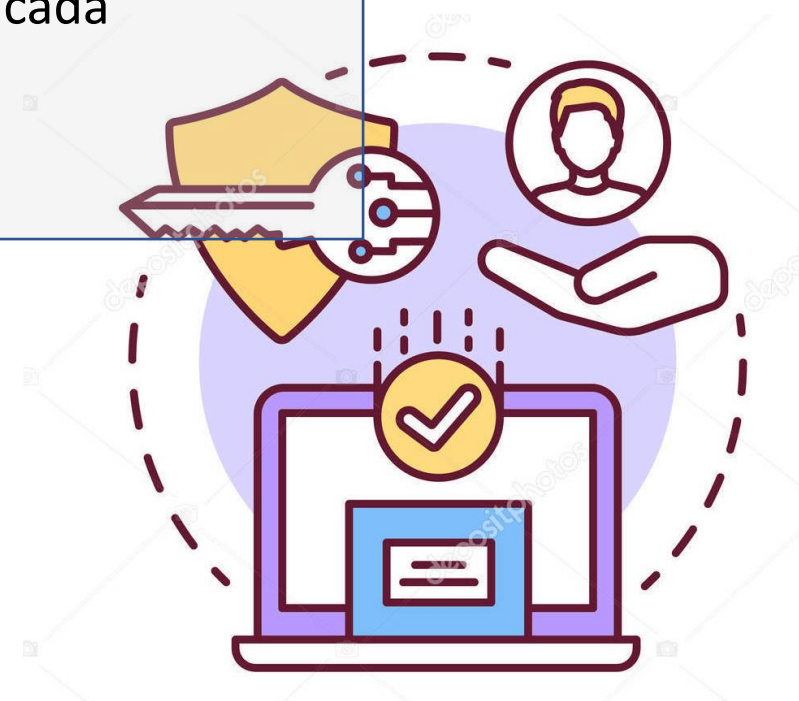
chage -l <user> # informações sobre a conta

chage -E YYYY-MM-DD <user> # expirar a conta na data especificada

chage -M 60 <user> # expirar o password em tantos dias

chage -m 0 -M 99999 -l -1 -E -1 <user> # nunca expira

```
root@ubuntu:/home/jhonatan# chage projeto
chage: user 'projeto' does not exist in /etc/passwd
root@ubuntu:/home/jhonatan# chage -l projeto
chage: user 'projeto' does not exist in /etc/passwd
root@ubuntu:/home/jhonatan# chage -l jhonatan
Last password change                : Dec 20, 2020
Password expires                    : never
Password inactive                   : never
Account expires                     : never
Minimum number of days between password change : 0
Maximum number of days between password change : 99999
Number of days of warning before password expires : 7
root@ubuntu:/home/jhonatan#
```



Administração de usuários - Linux

- **finger** – mostra a informação do sistema sobre um usuário existente;
 - Caso não esteja instalado por default: `apt-get install finger`
 - Sintaxe: `finger <usuário>`
- **id** – mostra a informação do usuário a que grupo ele está associado, inclusive se o usuário for do domínio;
 - Sintaxe: `id <usuário>`
- **useradd** – permite criar um usuário na máquina;
- **userdel** – permite excluir um usuário na máquina;
- **passwd** – possibilita alterar a senha de um usuário;
- **usermod** – permite modificar a conta de um usuário.



Administração de grupos - Linux

- `groupadd` – permite criar um novo grupo na máquina;
 - Sintaxe: `groupadd <grupo>`
- `groupdel` – permite excluir um grupo na máquina;
 - Sintaxe: `groupdel <grupo>`
- `groups` – permite listar os grupos associados a um usuário;
 - Sintaxe: `groups <usuário>`
- `groupmod` – permite modificar um grupo;
- `usermod` – permite também adicionar um usuário em um grupo;
 - Sintaxe: `usermod -aG <grupo> <usuário>`
- `deluser` – permite excluir um usuário de um grupo;
 - Sintaxe: `deluser <usuário> <grupo>`



Atividades

- Configurar o gerador repositório do Ubuntu;
- Atualizar repositório;
- Adicionar pacote com chave;
- Configurar o DNS (resolv.conf);
- Configurar o hostname (/etc/hosts);
- Configurar o IP da máquina;
- Como alterar o local do diretório home do usuário;
- Criar um novo usuário e mostrar onde ele fica;
- Alterar o interpretador de comando /bin/sh e /bin/bash;
- Montar e desmontar uma partição utilizando o comando mount e umount.



Comandos de Manutenção – Usuário e Grupo

- **usermod** – Ainda permite trancar (L) e destrancar (U) uma conta do usuário;
 - Sintaxe: **usermod -L <usuário>** e **usermod -U <usuário>**
- **chfn** – permite alterar as informações a respeito do usuário ativo (**/etc/passwd**).
- **pwck** – permite verificar a integridade do arquivo **passwd**;
- **grpck** – permite permite verificar a integridade e corrigir o arquivo **group**;
- **chage** – permite definir um prazo para que o password do usuário expire;
 - Sintaxe: **chage <parâmetros> <dias> <usuário>**
 - Ex.: **chage -M 100 projeto**
- **quota** – permite definir quotas de disco para o usuário;
- **quotaon** – comando para habilitar o sistema de quotas;
- **quotaoff** – comando para desabilitar o sistema de quotas;



Privilégios do sudo

- Permitir que um determinado usuário possa executar o comando sudo;
- Configuração do arquivo /etc/sudoers;

Comandos:

`sudo vim /etc/sudoers`

Adicionar a linha “**root ALL=(ALL) ALL**”

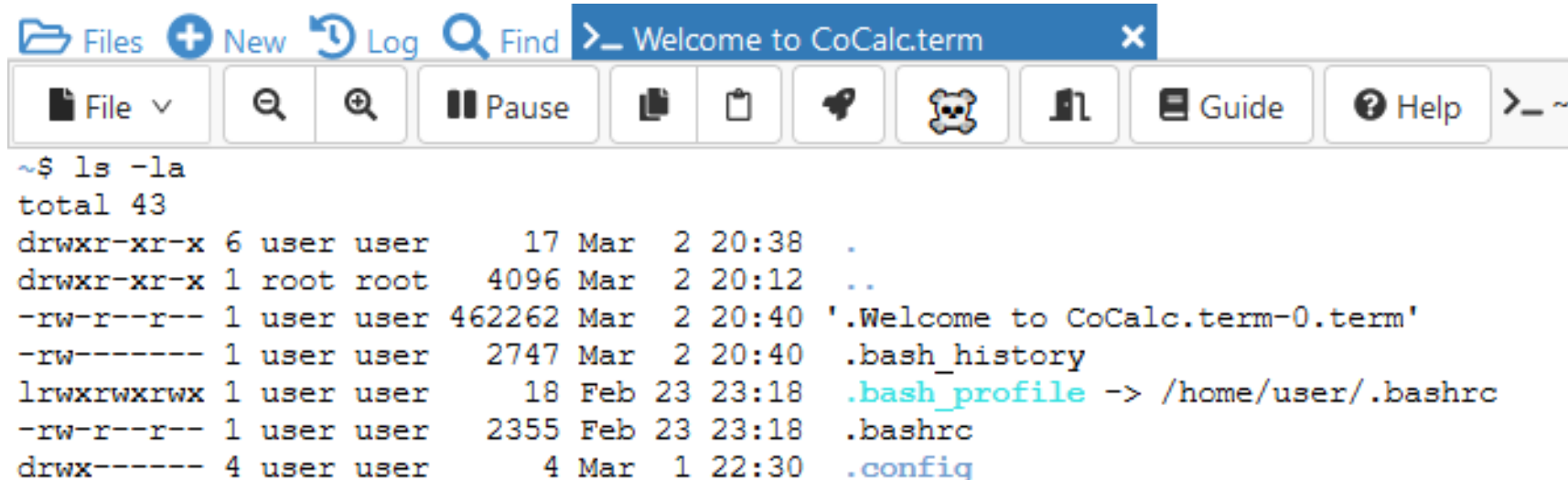
`<usuário> ALL=(ALL) ALL # <usuário> - alterar pelo usuário correspondente`

- O comando **visudo** é recomendado quando faz-se necessário uma alteração no sudoers.
 - Esse programa faz uma cópia temporária do arquivo sudoers, verificando erros de configuração e sintaxe.



Comando para gerenciamento de arquivos

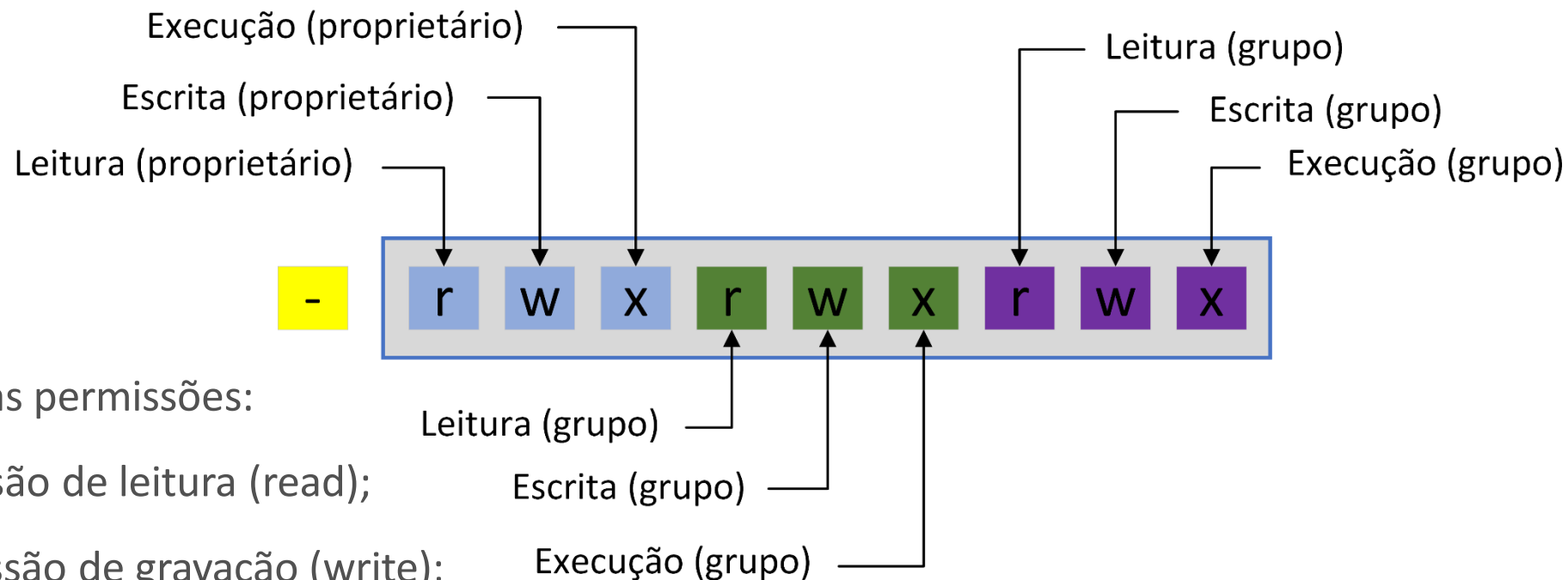
- `ls <filename>`: lista conteúdo de um diretório
 - Opções de argumentos:
 - `l` – listagem com detalhes, tipo de permissão, dono do diretório ou arquivo, grupo etc.;
 - `a` – listagem com arquivos ocultos;
 - `d` – lista somente diretórios – arquivos com `.`(ponto) no início do arquivo;
 - `F` – listagem diferenciada com caractere no nome.



```
~$ ls -la
total 43
drwxr-xr-x 6 user user    17 Mar  2 20:38 .
drwxr-xr-x 1 root root  4096 Mar  2 20:12 ..
-rw-r--r-- 1 user user 462262 Mar  2 20:40 '.Welcome to CoCalc.term-0.term'
-rw----- 1 user user   2747 Mar  2 20:40 .bash_history
lrwxrwxrwx 1 user user     18 Feb 23 23:18 .bash_profile -> /home/user/.bashrc
-rw-r--r-- 1 user user   2355 Feb 23 23:18 .bashrc
drwx----- 4 user user     4 Mar  1 22:30 .config
```



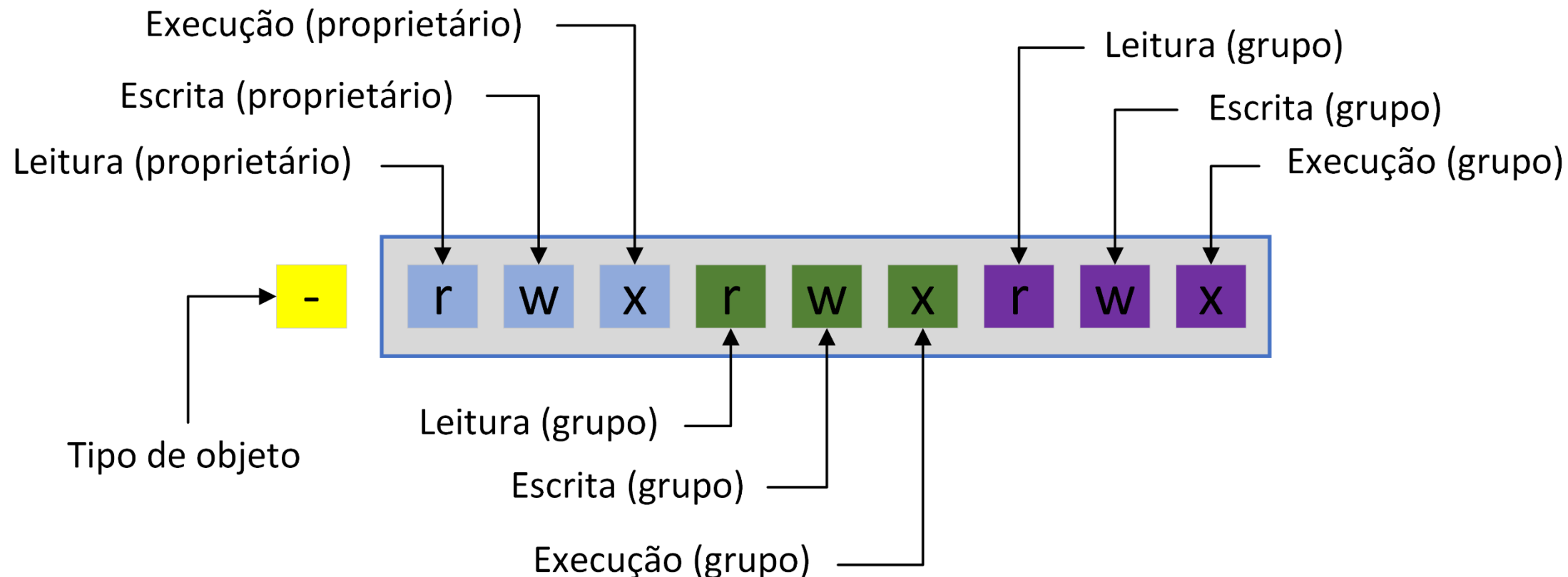
Permissões Linux



- Estrutura das permissões:
- **r** => permissão de leitura (read);
- **w** => permissão de gravação (write);
- **x** => permissão de execução (execution);
- **-** => permissão desabilitada.



Permissões Básicas - Linux



Permissões – Comando chmod

- O comando chmod permite alterar o nível de permissão de um arquivo ou diretório;
- As permissões são sempre definidas na sequência: usuário, grupo e outros;
 - A permissão pode ser definida de forma decimal ou alfabética;
- Conforme especificação:

Ex.: `chmod ugo+x`

• r – permissão de leitura	• o – permissão atribuída a outros usuários;
• w – permissão de gravação	• u – permissão atribuída ao proprietário;
• x – permissão de execução	• = – define permissão para
• a – permissão atribuída a todos (<i>all</i>)	• + – adiciona a permissão
• g – permissão atribuída ao grupo;	• - – retira permissão



Definir Permissões para o Comando chmod

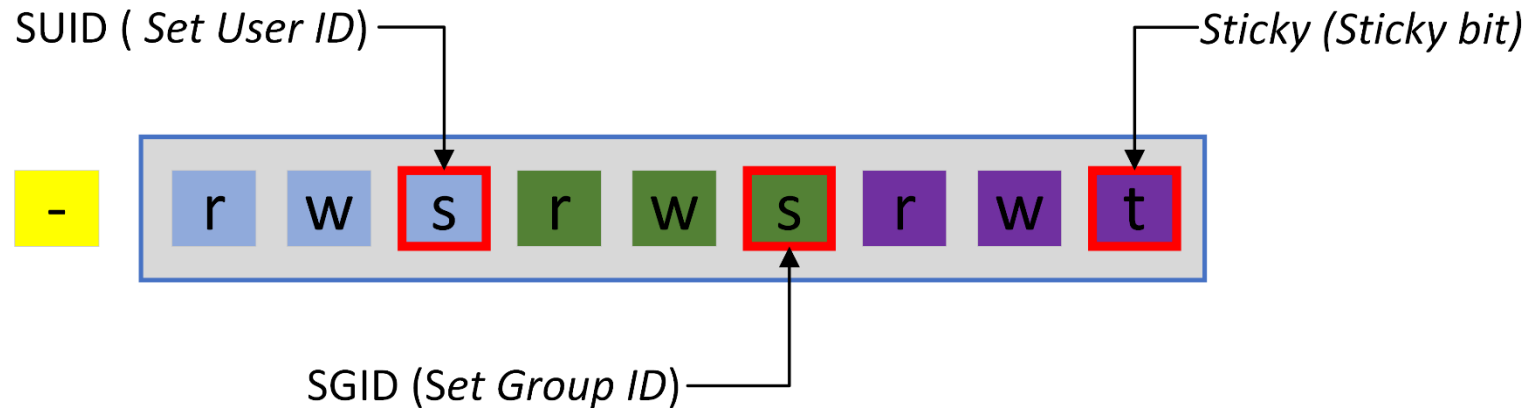
- Uma maneira para simplificar o processo é ter mente os valores de leitura, escrita e execução;
 - $r = 4$
 - $w = 2$
 - $x = 1$
- Para obter a permissão os valores decimais são somados diretamente;
 - Ex: `chmod 540 arquivo.txt`;
 - Permissão do usuário: leitura = 4, execução = 1, somando as permissões obtém o valor 5.

Permissão	Binário	Decimal
---	000	0
--X	001	1
-W-	010	2
-WX	011	3
r--	100	4
r-X	101	5
rW-	110	6
rWX	111	7

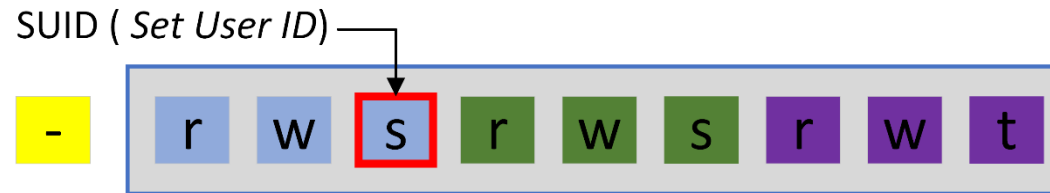


Permissões de Acesso Especiais

- Existem três modelos especiais para controle de acesso, chamados SUID (*Set User ID*), sgid (*Set Group ID*) e Sticky (*Sticky bit*);



Permissões de Acesso Especiais - SUID



- A propriedade SUID é apenas para arquivos executáveis não tendo efeito sob diretórios.
 - Um arquivo executável com a propriedade SUID aplicada, o programa rodará com o ID do dono do arquivo, não com o ID do usuário que executou o programa.
 - Normalmente o usuário dono do programa executável é também dono do processo sendo executado.
 - O processo do arquivo executável utilizando o acesso SUID é executado como se estivesse sido iniciado pelo dono do arquivo.
 - A permissão de acesso especial SUID pode aparecer somente no campo proprietário;
 - Um exemplo para arquivo executável com a propriedade SUID é o arquivo `/usr/bin/passwd`.
 - Exemplo de comando: `chmod u+s arquivo.sh`



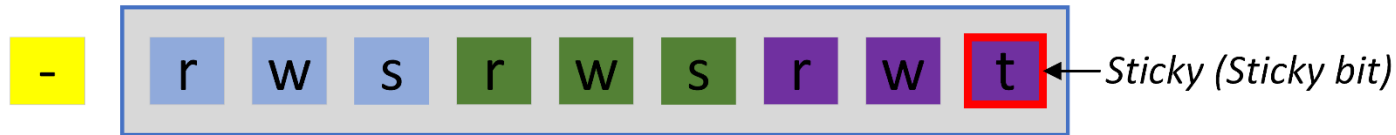
Permissões de Acesso Especiais - SGID



- A propriedade SGID tem a mesma função que o SUID para arquivos executáveis;
- Ainda o SGID tem um efeito especial aplicado em diretórios;
 - Quando SGID é aplicado em um diretório, os novos arquivos que são criados dentro do diretório assumem o mesmo ID de Grupo do diretório com a propriedade SGID aplicado;
 - A permissão de acesso especial SGID pode aparece somente no campo grupo;
 - Exemplo comando: `chmod g+s /home/equipe`



Permissões de Acesso Especiais - Sticky



- Em arquivos executáveis, a propriedade *Sticky* faz com que o sistema mantenha uma imagem do programa em memória depois que o programa finalizar.
- Esta capacidade aumenta o desempenho;
 - Será efetuado um cache do programa para a memória;
 - A próxima vez que ele for executado, será carregado mais rápido;
- Capacidade de Aumentar a segurança;
 - A propriedade *Sticky* aplicada em diretórios impede que outros usuários deletem ou renomeie arquivos que não são donos.
 - O diretório estará em modo *append-only* (somente incremento);
 - Somente o dono do arquivo, poderá deletar ou renomear os arquivos;
 - Exemplo comando: `chmod o+t programa_pesado.sh`



Permissões - Comando umask

- O comando `umask` permite criar uma máscara utilizada para definir as permissões de novos arquivos e diretórios;
 - Permissões aplicadas pela máscara em novos arquivos inicialmente não possui o privilégio de execução para nenhum dos usuários – valor máximo: `666 (rw-rw-rw-)` (`110-110-110`);
 - Para acesso aos diretórios essa permissão é aplicada – valor máximo: `777 (rwx-rwx-rwx)` (`111-111-111`);
- Para utilizar o `umask` é necessário subtrair o valor máximo pela permissão desejada;
 - Exemplo: de permissão para arquivo: `(rw-r-----)` a permissão seria `640`
 - Obtendo a permissão para o `umask`: $666 - 640 = 026$
 - Comando: `umask 0026`



Permissões - Comando umask

umask →

```
~$ touch arquivo1.txt
~$ mkdir pasta1
~$ ls -l
total 1
-rw-r--r-- 1 user user 0 Mar  3 22:32 arquivo1.txt
drwxr-xr-x 2 user user 2 Mar  3 22:32 pasta1
~$ umask 0017
~$ touch arquivo2.txt
~$ mkdir pasta2
~$ ls -l
total 2
-rw-r--r-- 1 user user 0 Mar  3 22:32 arquivo1.txt
-rw-rw---- 1 user user 0 Mar  3 22:33 arquivo2.txt
drwxr-xr-x 2 user user 2 Mar  3 22:32 pasta1
drwxrw---- 2 user user 2 Mar  3 22:33 pasta2
~$
```

permissão do arquivo criado antes do comando umask

permissão do diretório criado antes do comando umask

permissão do arquivo criado depois do comando umask

permissão do diretório criado depois do comando umask



Permissões - Comando umask

- O valor umask pode ser encontrado e configurado no /etc/profile ou /etc/bash.bashrc;
- Para definir as permissões com o umask é possível utilizar a notação alfabética;
 - Exemplo do comando: `umask u+rw.`

umask notação alfabética

```
~$ umask ugo+w
~$ touch arquivo.txt
~$ mkdir pasta
~$ ls -l
total 1
-rw-rw-rw- 1 user user 0 Mar  3 22:50 arquivo.txt
drwxrwxrwx 2 user user 2 Mar  3 22:50 pasta
~$ umask ugo-wx
~$ touch arquivo2.txt
~$ mkdir pasta2
~$ ls -l
total 2
-rw-rw-rw- 1 user user 0 Mar  3 22:50 arquivo.txt
-r--r--r-- 1 user user 0 Mar  3 22:51 arquivo2.txt
drwxrwxrwx 2 user user 2 Mar  3 22:50 pasta
dr--r--r-- 2 user user 2 Mar  3 22:52 pasta2
```



Comando de Permissão de Acesso - chown

- O comando chown permite alterar o proprietário e grupo no qual está associado um arquivo ou diretório;
- Sintaxe do comando:
 - Alterar proprietário: `chown <usuário> <arquivo>`
 - Alterar proprietário e grupo: `chown <usuário>:<grupo> <arquivo>`

```
root@socps-pc:~/aula# ls -l
total 4
-rw-r--r-- 1 root root 14 Mar  4 09:28 arquivo.txt
root@socps-pc:~/aula# chown projeto:projeto arquivo.txt
root@socps-pc:~/aula# ls -l
total 4
-rw-r--r-- 1 projeto projeto 14 Mar  4 09:28 arquivo.txt
```



Comando de Permissão de Acesso - chgrp

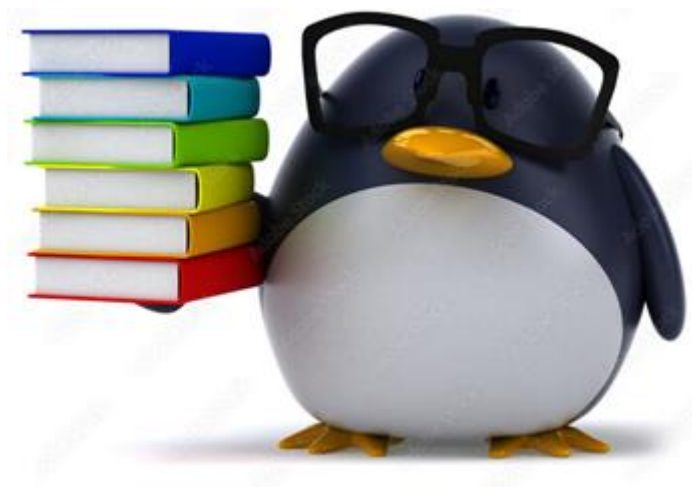
- O comando chgrp permite alterar o grupo no qual está associado um arquivo ou diretório;
- Sintaxe do comando:
 - Alterar grupo: `chown <grupo> <arquivo>`

```
root@ctic-pc:~/aula# ls -l
total 4
-rw-r--r-- 1 root root 14 Mar  4 09:28 arquivo.txt
root@ctic-pc:~/aula# chgrp projeto arquivo.txt
root@ctic-pc:~/aula# ls -l
total 4
-rw-r--r-- 1 root projeto 14 Mar  4 09:28 arquivo.txt
```



Lista de Controle de Acesso Estendida

- Uma ACL (*Access Control List* / Lista de Controle de Acesso) é uma configuração de segurança que nos fornece um controle mais refinado sobre quais usuários podem acessar diretórios e arquivos específicos do que as permissões tradicionais do Linux.



Lista de Controle de Acesso Estendida

- A permissão da lista de controle de acesso (ACL) estendida é aplicada utilizando o comando **setfact**.
- Sintaxe do comando setfact:
 - Passar o parâmetro -m para adicionar a permissão,
 - Fornecer o usuário “u:<usuário>”, um sinal de dois pontos “:” e a permissão a ser atribuída
 - Permissões: leitura (r), escrita (w) e execução (x)

setfact -m u:<usuário>:<permissão> <objeto>

```
aluno@ubuntu:~/u4$ ls -la
total 8
drwxr-xr-x  2 aluno equipe1 4096 Jul  6 06:28 .
drwxr-xr-x  3 aluno equipe1 4096 Jul  6 06:26 ..
-rw-rw----+ 1 aluno equipe1    0 Jul  6 06:28 arquivo1.txt
aluno@ubuntu:~/u4$
```

getfact arquivo1.txt

```
aluno@ubuntu:~/u4$ getfacl arquivo1.txt
# file: arquivo1.txt
# owner: aluno
# group: equipe1
user::rw-
user:aluno2:rw-
user:aluno3:rw-
group::r--
mask::rw-
other::---
aluno@ubuntu:~/u4$
```





Obrigado!

Jhonatan Geremias

Jhonatan.geremias@pucpr.br

