

ANHANGUERA EDUCACIONAL

Curso: Engenharia de Software

Relatório de Atividades Práticas

Disciplina: Sistemas Operacionais

Aluno: Wesley Ferreira Barbosa Lima

Unidade/Polo: Anhanguera Limeira/SP

Limeira/SP
2026

ANHANGUERA EDUCACIONAL

Curso: Engenharia de Software

Disciplina: Sistemas Operacionais

Relatório de Atividades Práticas – Sistemas Operacionais

Aluno: Wesley Ferreira Barbosa Lima

RA: 202600961

Unidade/Polo: Anhanguera Limeira/SP

Semestre/Ano: 1º semestre / 2026

Limeira/SP
2026

Este relatório apresenta os resultados das atividades práticas da disciplina de Sistemas Operacionais. O objetivo é desenvolver a compreensão dos principais componentes dos sistemas operacionais Linux e Windows, com foco no kernel, shell e gerenciamento de arquivos.

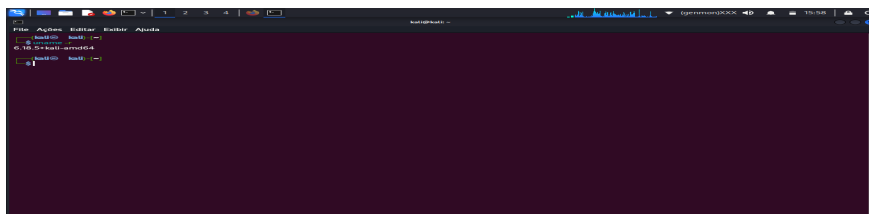
As atividades propostas permitem explorar o funcionamento do kernel e do shell, o gerenciamento de processos e permissões de arquivos, além da comparação entre as estruturas de diretórios dos sistemas Linux e Windows.

Com isso, busca-se consolidar o aprendizado sobre a organização e funcionamento dos sistemas operacionais, preparando o aluno para um gerenciamento mais eficaz desses ambientes.

Atividade 1 – Kernel e Shell (Linux)

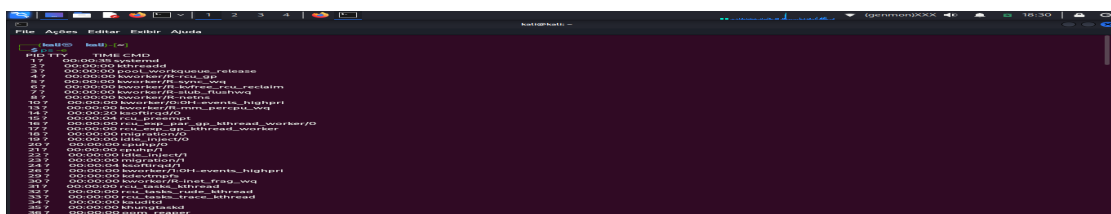
Linux

- Comando: `uname -r`
- Resultado: `6.18.5+kali-amd64`
- Explicação: Esse comando mostra a versão do kernel em execução no Linux. O kernel é responsável por gerenciar processos e comunicação com o hardware.



```
root@kali: ~  
root@kali:~# uname -r  
6.18.5+kali-amd64
```

- Comando: `ps -e`
- Explicação: O comando `ps -e` lista todos os processos ativos no sistema Linux.



```
root@kali: ~  
root@kali:~# ps -e  
PID TTY          TIME CMD  
1    ?        00:00:00 systemd  
2    ?        00:00:00 kthreadd  
3    ?        00:00:00 kworker/0/0  
4    ?        00:00:00 kworker/0/0  
5    ?        00:00:00 kworker/0/0  
6    ?        00:00:00 kworker/0/0  
7    ?        00:00:00 kworker/0/0  
8    ?        00:00:00 kworker/0/0  
9    ?        00:00:00 kworker/0/0  
10   ?        00:00:00 kworker/0/0  
11   ?        00:00:00 kworker/0/0  
12   ?        00:00:00 kworker/0/0  
13   ?        00:00:00 kworker/0/0  
14   ?        00:00:00 kworker/0/0  
15   ?        00:00:00 kworker/0/0  
16   ?        00:00:00 kworker/0/0  
17   ?        00:00:00 kworker/0/0  
18   ?        00:00:00 kworker/0/0  
19   ?        00:00:00 kworker/0/0  
20   ?        00:00:00 kworker/0/0  
21   ?        00:00:00 kworker/0/0  
22   ?        00:00:00 kworker/0/0  
23   ?        00:00:00 kworker/0/0  
24   ?        00:00:00 kworker/0/0  
25   ?        00:00:00 kworker/0/0  
26   ?        00:00:00 kworker/0/0  
27   ?        00:00:00 kworker/0/0  
28   ?        00:00:00 kworker/0/0  
29   ?        00:00:00 kworker/0/0  
30   ?        00:00:00 kworker/0/0  
31   ?        00:00:00 kworker/0/0  
32   ?        00:00:00 kworker/0/0  
33   ?        00:00:00 kworker/0/0  
34   ?        00:00:00 kworker/0/0  
35   ?        00:00:00 kworker/0/0  
36   ?        00:00:00 kworker/0/0  
37   ?        00:00:00 kworker/0/0  
38   ?        00:00:00 kworker/0/0  
39   ?        00:00:00 kworker/0/0  
40   ?        00:00:00 kworker/0/0  
41   ?        00:00:00 kworker/0/0  
42   ?        00:00:00 kworker/0/0  
43   ?        00:00:00 kworker/0/0  
44   ?        00:00:00 kworker/0/0  
45   ?        00:00:00 kworker/0/0  
46   ?        00:00:00 kworker/0/0  
47   ?        00:00:00 kworker/0/0  
48   ?        00:00:00 kworker/0/0  
49   ?        00:00:00 kworker/0/0  
50   ?        00:00:00 kworker/0/0  
51   ?        00:00:00 kworker/0/0  
52   ?        00:00:00 kworker/0/0  
53   ?        00:00:00 kworker/0/0  
54   ?        00:00:00 kworker/0/0  
55   ?        00:00:00 kworker/0/0  
56   ?        00:00:00 kworker/0/0  
57   ?        00:00:00 kworker/0/0  
58   ?        00:00:00 kworker/0/0  
59   ?        00:00:00 kworker/0/0  
60   ?        00:00:00 kworker/0/0  
61   ?        00:00:00 kworker/0/0  
62   ?        00:00:00 kworker/0/0  
63   ?        00:00:00 kworker/0/0  
64   ?        00:00:00 kworker/0/0  
65   ?        00:00:00 kworker/0/0  
66   ?        00:00:00 kworker/0/0  
67   ?        00:00:00 kworker/0/0  
68   ?        00:00:00 kworker/0/0  
69   ?        00:00:00 kworker/0/0  
70   ?        00:00:00 kworker/0/0  
71   ?        00:00:00 kworker/0/0  
72   ?        00:00:00 kworker/0/0  
73   ?        00:00:00 kworker/0/0  
74   ?        00:00:00 kworker/0/0  
75   ?        00:00:00 kworker/0/0  
76   ?        00:00:00 kworker/0/0  
77   ?        00:00:00 kworker/0/0  
78   ?        00:00:00 kworker/0/0  
79   ?        00:00:00 kworker/0/0  
80   ?        00:00:00 kworker/0/0  
81   ?        00:00:00 kworker/0/0  
82   ?        00:00:00 kworker/0/0  
83   ?        00:00:00 kworker/0/0  
84   ?        00:00:00 kworker/0/0  
85   ?        00:00:00 kworker/0/0  
86   ?        00:00:00 kworker/0/0  
87   ?        00:00:00 kworker/0/0  
88   ?        00:00:00 kworker/0/0  
89   ?        00:00:00 kworker/0/0  
90   ?        00:00:00 kworker/0/0  
91   ?        00:00:00 kworker/0/0  
92   ?        00:00:00 kworker/0/0  
93   ?        00:00:00 kworker/0/0  
94   ?        00:00:00 kworker/0/0  
95   ?        00:00:00 kworker/0/0  
96   ?        00:00:00 kworker/0/0  
97   ?        00:00:00 kworker/0/0  
98   ?        00:00:00 kworker/0/0  
99   ?        00:00:00 kworker/0/0  
100  ?        00:00:00 kworker/0/0
```

Windows

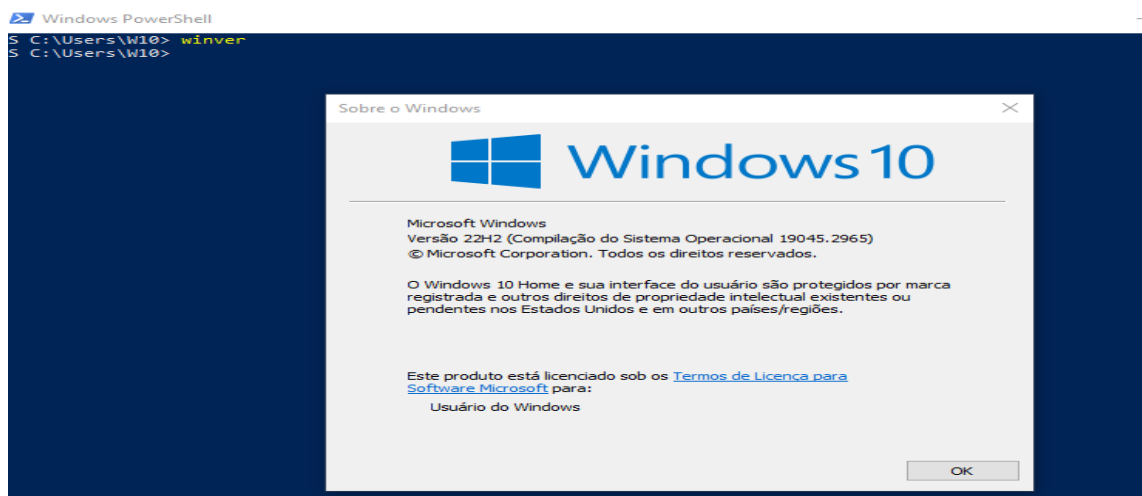
- Comando: **Get-Process**
- Explicação: O comando **Get-Process** lista todos os processos ativos no Windows. Cada processo possui um identificador (Id), consumo de memória e tempo de CPU. Isso demonstra como o **kernel do Windows** gerencia os recursos do sistema, garantindo que cada aplicação receba memória e tempo de processamento.

Experimente a nova plataforma cruzada PowerShell <https://aka.ms/pscore6>

```
PS C:\Users\W10> Get-process
```

Handles	NPM(K)	PM(K)	WS(K)	CPU(s)	Id	SI	ProcessName
378	21	9512	30800	1,11	1664	1	ApplicationFrameHost
267	14	4132	18584	0,25	1280	1	conhost
311	14	1640	4756		424	0	csrss
317	17	1772	5244		508	1	csrss
422	16	3908	19568	1,53	2584	1	ctfmon
255	22	4484	14516	0,48	1324	1	dllhost
256	27	6548	16176	0,94	3672	1	dllhost
951	48	36616	64264		920	1	dwm

- Comando: **winver**
- Explicação: O comando **winver** abre uma janela gráfica que mostra a versão do sistema operacional Windows em execução.



Atividade 2: Gerenciamento de Arquivos e Permissões

Linux

- Atividade realizada: Criar um diretório chamado `test_dir` e, dentro dele, criar um arquivo `test_file.txt`. Em seguida, atribuir permissões de leitura e escrita ao arquivo.

Pontos de atenção

- As permissões em sistemas Linux são representadas por três letras:
 - **R (read)** → leitura
 - **W (write)** → escrita
 - **X (execute)** → execução
- Essas permissões se aplicam a três grupos distintos
 - **Usuário (owner)**
 - **Grupo (group)**
 - **Outros (others)**
- Cada permissão tem um valor numérico:
 - **R = 4**
 - **W = 2**
 - **X = 1**
- Para definir as permissões, somamos os valores desejados para cada grupo.
 - Exemplo: **6 = 4 (leitura) + 2 (escrita)** → `rw-`

```
kali@kali: ~/test_dir
$ mkdir test_dir
$ cd test_dir
$ touch test_file.txt
$ ls
test_file.txt
$ chmod 600 test_file.txt
$ ls -la
total 8
drwxrwxr-x 2 kali kali 4096 fev 7 09:12 .
drwx-x-x 30 kali kali 4096 fev 7 09:11 ..
-rw----- 1 kali kali 0 fev 7 09:12 test_file.txt
$!
```

Windows

- Criei uma pasta chamada **TestFolder**, acessei seu conteúdo e dentro dela criei um arquivo chamado **TestFile.txt**. Em seguida, atribuí ao usuário **W10** a permissão **F (Full Control)** sobre esse arquivo, garantindo acesso total às suas funcionalidades.

```
Windows PowerShell (x86)
PS C:\Users\W10> New-Item -ItemType Directory -Name TestFolder

Diretório: C:\Users\W10

Mode                LastWriteTime         Length Name
----                -
d-----          07/02/2026   10:16             TestFolder

PS C:\Users\W10> Set-Location TestFolder
PS C:\Users\W10\TestFolder> New-Item -ItemType File -Name TestFile.txt

Diretório: C:\Users\W10\TestFolder

Mode                LastWriteTime         Length Name
----                -
-a-----          07/02/2026   10:17             0 TestFile.txt

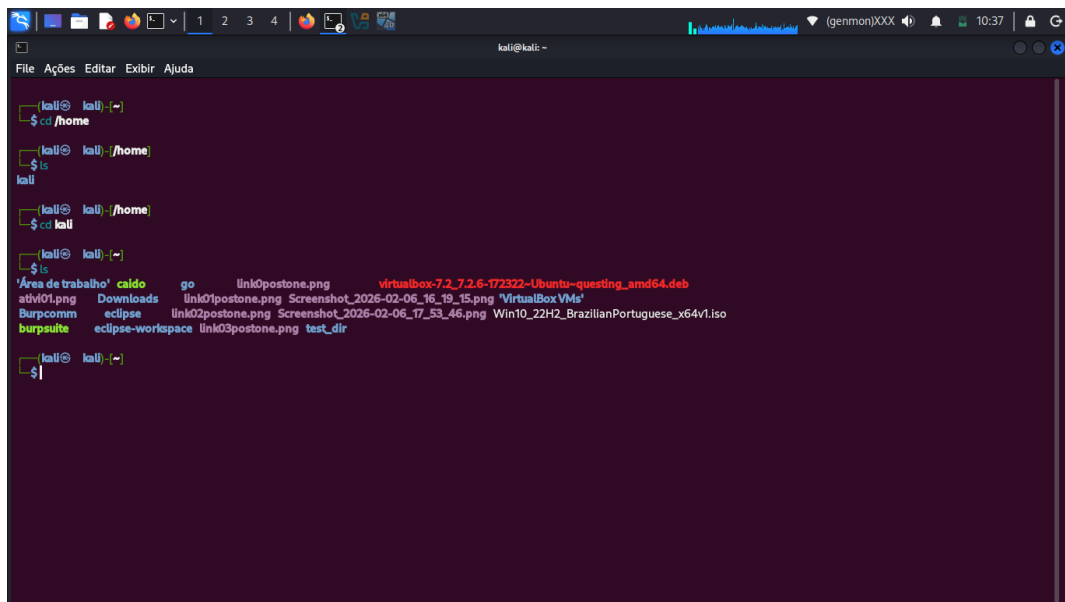
PS C:\Users\W10\TestFolder> icacls TestFile.txt /grant W10:F
arquivo processado: TestFile.txt
Processados com sucesso 1 arquivos; falha no processamento de 0 arquivos
PS C:\Users\W10\TestFolder> icacls TestFile.txt
TestFile.txt WINDOWS10\W10:(F)
               AUTORIDADE NT\SISTEMA:(I)(F)
               BUILTIN\Administradores:(I)(F)
               WINDOWS10\W10:(I)(F)

Processados com sucesso 1 arquivos; falha no processamento de 0 arquivos
PS C:\Users\W10\TestFolder>
```

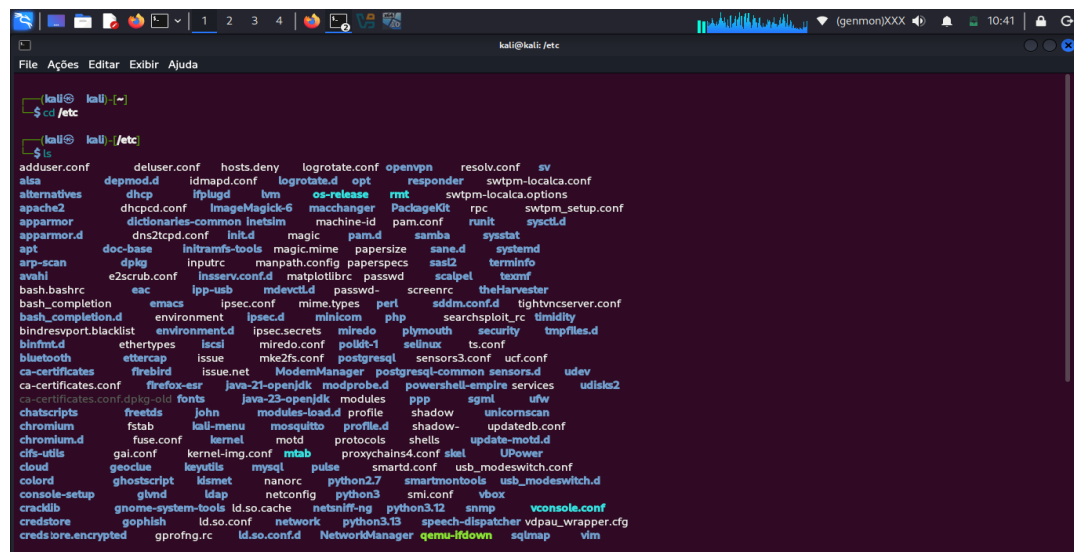
Atividade 3: Comparando Estruturas de Diretórios

Linux

- Diretórios: /home
- Função: Armazenar as informações e arquivos pessoais de cada usuário comum do sistema.



- Diretório: /etc
- Função: armazenar **arquivos de configuração do sistema.**



- Diretório: /var
- Função: serve para armazenar **informações variáveis do sistema**,

Windows

- Diretório: C:/
- Função: é o **ponto mais alto da hierarquia de arquivos** no Windows, equivalente à raiz do sistema de arquivos.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Experimente a nova plataforma cruzada PowerShell https://aka.ms/pscore6

PS C:\Users\W10> Set-Location C:/
PS C:\> ls

    Diretório: C:\

Mode                LastWriteTime         Length Name
----                -
d-----          07/12/2019    06:14          PerfLogs
d-r---          06/02/2026    18:04      Program Files
d-r---          05/05/2023    09:28  Program Files (x86)
d-r---          06/02/2026    17:08          Users
d-----          06/02/2026    18:05        Windows
-a----          06/02/2026    17:09         1592 vboxpostinstall.log

PS C:\>
```

- Diretório: C:/ Users
- Função: Guarda os perfis e dados pessoais de cada usuário .

```
Windows PowerShell
Copyright (C) Microsoft Corporation. Todos os direitos reservados.

Experimente a nova plataforma cruzada PowerShell https://aka.ms/pscore6

PS C:\Users\W10> cd C:/
PS C:\> cd C:/Users
PS C:\Users> ls

    Diretório: C:\Users

Mode                LastWriteTime         Length Name
----                -
d-r---          06/02/2026    17:09          Public
d-----          07/02/2026    10:16          W10

PS C:\Users>
```


- Diretório: C:/Windows
- Função: Onde ficam os **arquivos essenciais para o funcionamento do sistema**.

```

Windows PowerShell
PS C:\> cd C:/Windows
PS C:\Windows> ls

Diretório: C:\Windows

Mode                LastWriteTime         Length Name
----                -
d-----          07/12/2019         11:55      addins
d-----          07/12/2019          06:31    appcompat
d-----          05/05/2023          09:27    apppatch
d-----          06/02/2026         17:31  AppReadiness
d-r---          06/02/2026         17:44    assembly
d-----          05/05/2023          09:27    bcastdvr
d-----          07/12/2019          06:31      Boot
d-----          07/12/2019          06:14    Branding
d-----          07/02/2026          09:57    CbsTemp
d-----          07/12/2019          06:14    Containers
d-----          07/12/2019          06:14    Cursors
d-----          06/02/2026         17:02    debug
d-----          07/12/2019          06:31    diagnostics
d-----          05/05/2023          09:27    DiagTrack
d-----          07/12/2019         11:54    DigitalLocker

```

- Diretório: "C:/Program Files"
- Função: Função **organizar e armazenar os programas instalados no Windows**.

Resultados de aprendizagem

As atividades realizadas permitiram compreender melhor a função do kernel e do shell, bem como a organização da estrutura de diretórios em Linux e Windows. A prática mostrou como os sistemas operacionais gerenciam processos, arquivos e permissões, consolidando a compreensão sobre segurança, eficiência e interação entre componentes fundamentais do sistema.