

## **RELATÓRIO DO PROJETO**

Disciplina: Sistemas Operacionais  
Professor: Clóvis Ferraro  
Grupo: 06

## Sumário

<b>1. Introdução .....</b>	<b>3</b>
<b>2. Metodologia .....</b>	<b>3</b>
<b>3. Comparação entre os Sistemas Operacionais .....</b>	<b>6</b>
3.1 Windows .....	6
3.1.1 Instalação e Configuração .....	6
3.1.2 Testes e Comandos .....	8
3.2 Linux .....	10
3.2.1 Instalação e Configuração .....	10
3.2.2 Testes e Comandos .....	11
3.3 Android .....	13
3.3.1 Instalação e Configuração .....	13
3.3.2 Testes e Comandos .....	14
3.4 Comparação Crítica .....	17
<b>4. Análise Crítica .....</b>	<b>17</b>
<b>5. Conclusão .....</b>	<b>19</b>
<b>6. Autoavaliação .....</b>	<b>20</b>
<b>7. Referências .....</b>	<b>20</b>

## 1. Introdução

O presente módulo tem como objetivo realizar a configuração e a comparação de máquinas virtuais em diferentes sistemas operacionais, especificamente Windows, Linux e Android. Busca-se compreender as etapas necessárias para a criação, instalação e personalização desses ambientes, bem como analisar diferenças de desempenho, compatibilidade e recursos entre cada sistema. A utilização de máquinas virtuais apresenta grande relevância no atual contexto tecnológico, possibilitando a execução de diversos sistemas operacionais em um mesmo equipamento físico, sem a necessidade de particionar ou instalar diretamente no hardware. Essa prática é fundamental em ambientes educacionais e corporativos, permitindo testes, simulações e experimentações em condições seguras, além disso, reduz custos e otimiza o uso dos recursos computacionais.

## 2. Metodologia

Os testes foram realizados em três sistemas operacionais distintos (Windows, Linux e Android), cada um configurado em máquinas virtuais para garantir condições controladas e fáceis de reproduzir.

Para desenvolvimento do projeto nos três sistemas operacionais, foi realizado processo de Criação das Máquinas Virtuais, configuração das mesmas, e depois de instaladas na ferramenta de virtualização, foram realizados testes com comandos específicos para cada Sistema, afim de entender e desenvolver o funcionamento.

**Ferramentas de virtualização utilizadas:** Oracle VirtualBox

## Configuração das máquinas virtuais:

### LINUX

Memória de Vídeo: 2048 MB

CPU: 1 núcleos

Disco rígido virtual: 30 GB

**Versão:** Ubuntu-24.04.3

Os comandos utilizados no Linux para testes foram:

**ip addr show** - Serve para descobrir a interface de rede conectada, no caso descobrimos que era a **enp0s3**, que será muito importante nos próximos comandos.

**sudo ip addr add 192.168.100.10/24 dev enp0s3** – Esse comando atribui o IP (192.168.100.10) à interface, definindo o IP da máquina

**sudo ip link set enp0s3 up** – Já esse comando liga a interface que descobrimos no primeiro comando, para que o endereço IP funcione de fato

**ping 192.168.100.20** – Comando utilizado para testar a conectividade entre duas máquinas.

Realizamos as mesmas configurações e comandos em outra máquina virtual para testar a conectividade entre as duas, para isso utilizamos o comando ping, que envia um pacote de teste para o IP especificado na outra máquina, em nosso caso **192.168.100.20**, caso o destinatário estiver ativo e conectado, responde ao pacote.

## ANDROID

Memória: 2048 MB

CPU: 2 núcleos

Disco rígido virtual: 30 GB

**Versão:** Android 7.1

Os comandos utilizados no Android para testes foram:

**cat /proc/cpuinfo** – Mostra informações detalhadas sobre o processador, como modelo, frequência, número de núcleos e caches.

**df -h** – Esse comando é utilizado para mostrar o espaço disponível em disco.

**free -m** – Mostra a memória usada em livre, mas mostra em MB.

**ps** – Usado para listar os processos em execução no sistema.

**top** – Esse comando mostra CPU, memória e processos em tempo real, assim permitindo verificar continuamente o uso do sistema.

## WINDOWS

Memória: 2048 MB

CPU: 2 núcleos

Controladora Gráfica: 30 GB

**Versão:** Windows 10

Os comandos utilizados no Windows para testes foram:

**ipconfig** – Este é um dos comandos mais utilizados em Windows, serve para informar as configurações de rede da máquina, como endereço IP, máscara de sub-rede e gateway.

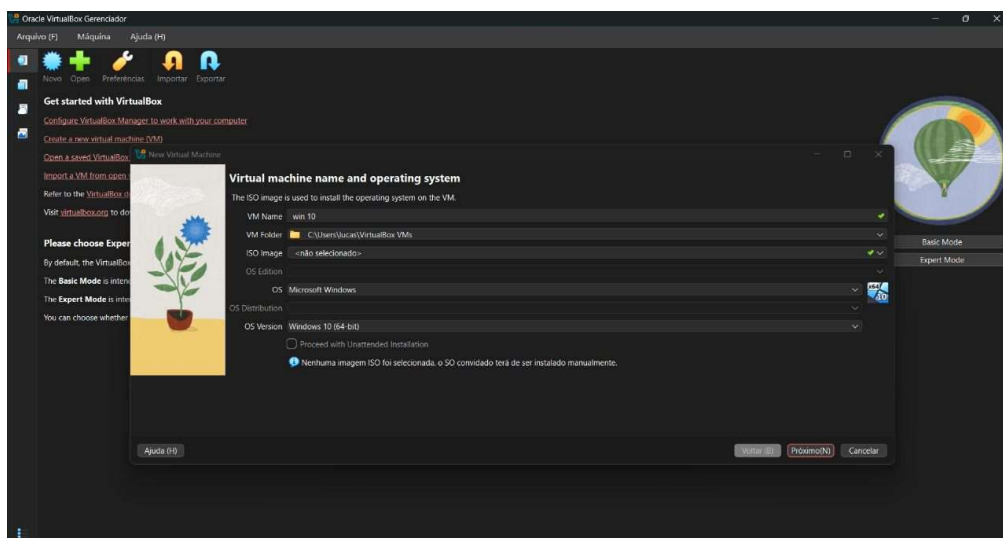
**tasklist** – Lista todos os processos em execução no Windows, mostrando o nome do processo, o PID (Identificador do Processo) e uso de memória.

**ping** – Da mesma forma como o Linux, este comando é utilizado para testar a conectividade com outras máquinas que estejam na mesma rede.

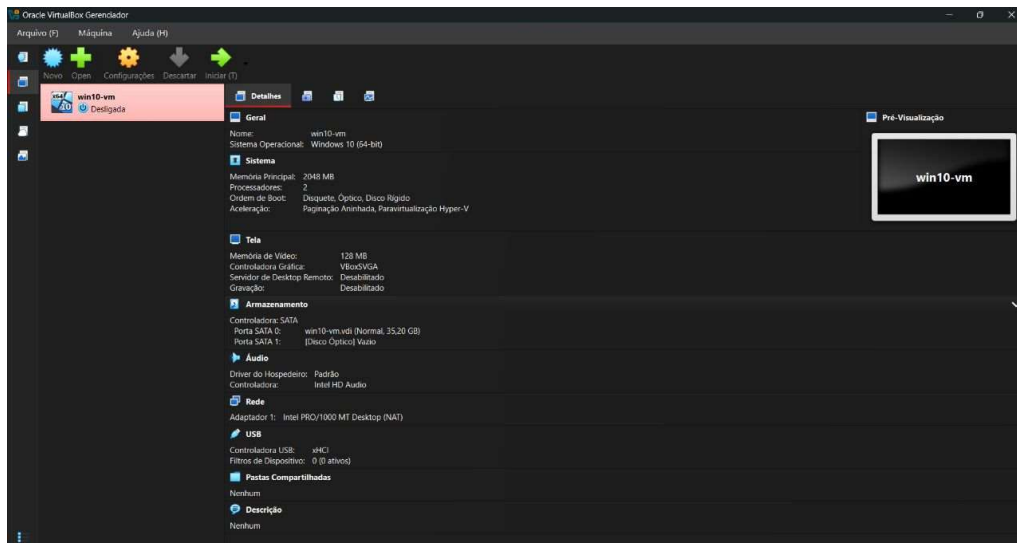
### 3. Comparação entre os Sistemas Operacionais

#### 3.1 Windows

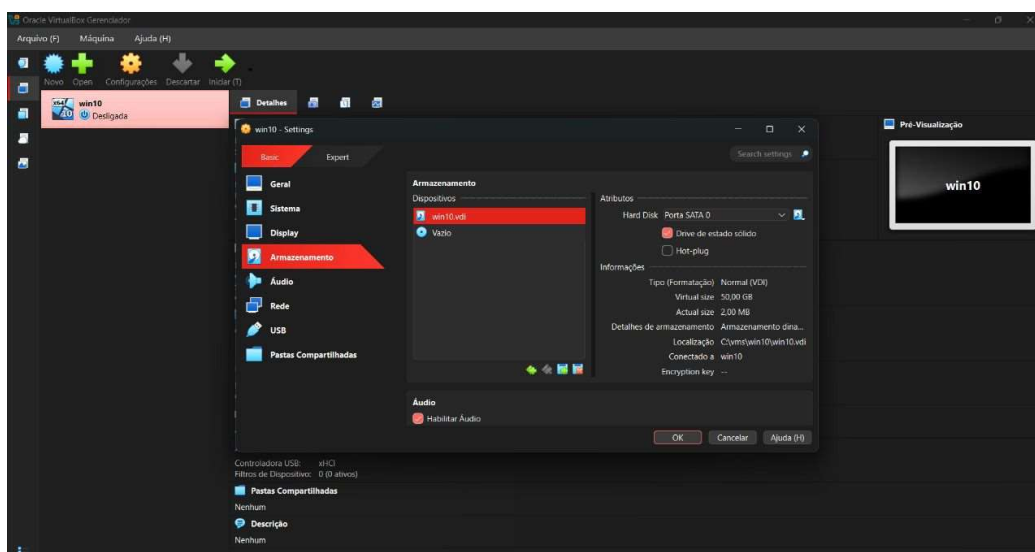
##### 3.1.1 Instalação e Configuração



Tela de definição de Sistema operacional e nome da Máquina Virtual

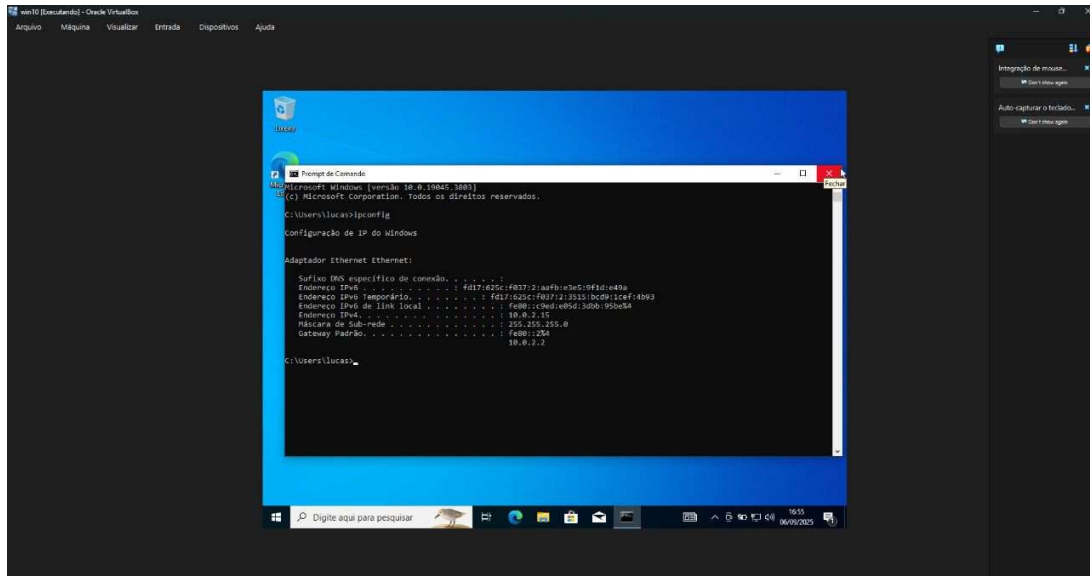


**Máquina Virtual Windows criada**

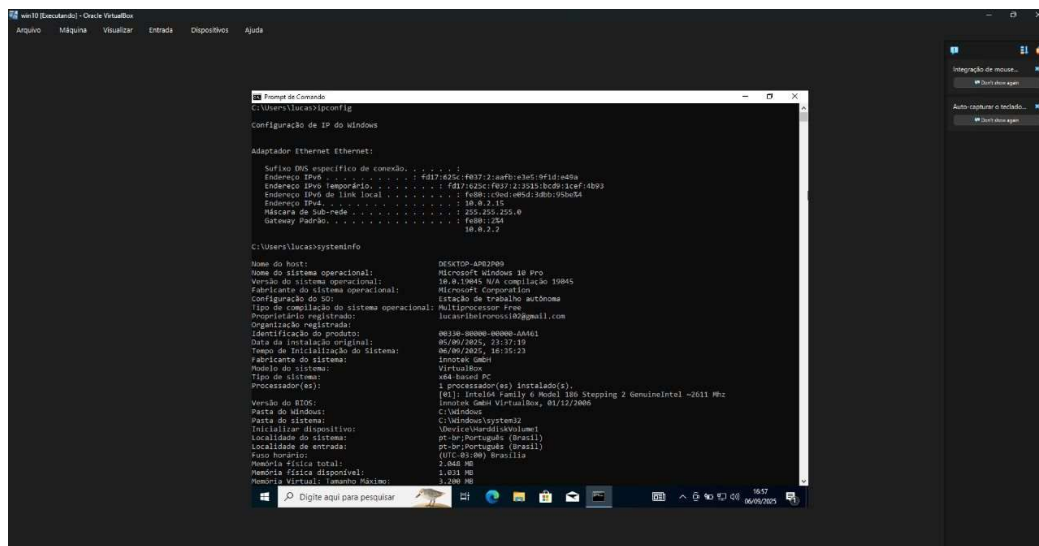


**Tela de detalhamento da configuração da máquina Virtual**

### 3.1.2 Testes e Comandos

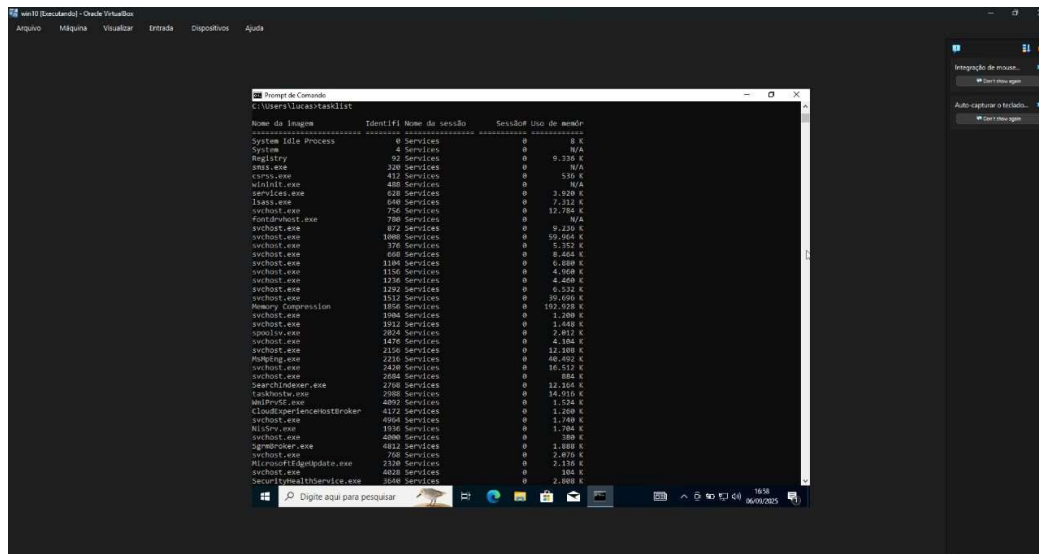


Windows e Prompt de Comandos abertos

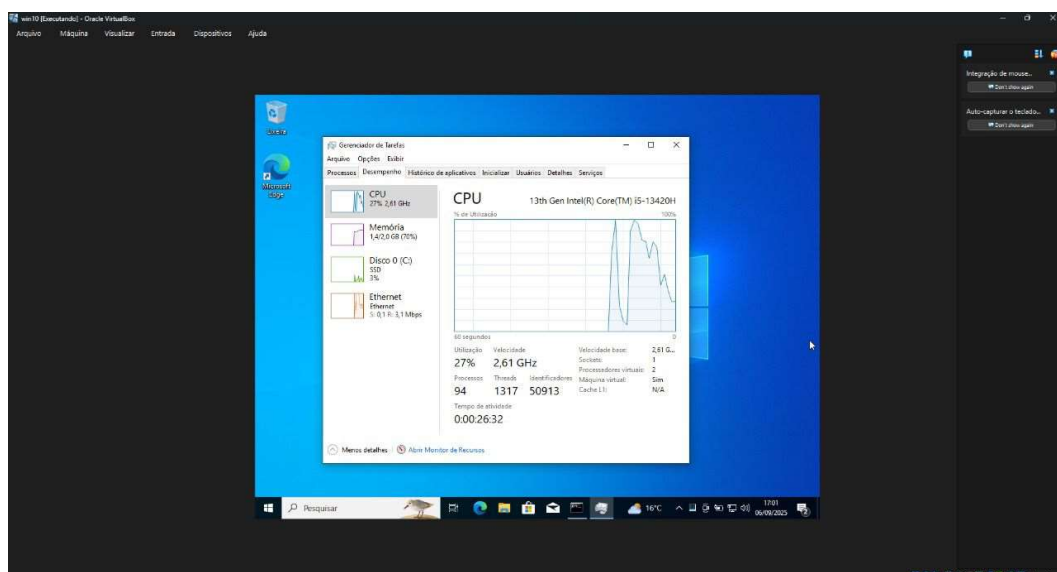


Comando utilizado: ipconfig – Informações de rede da máquina





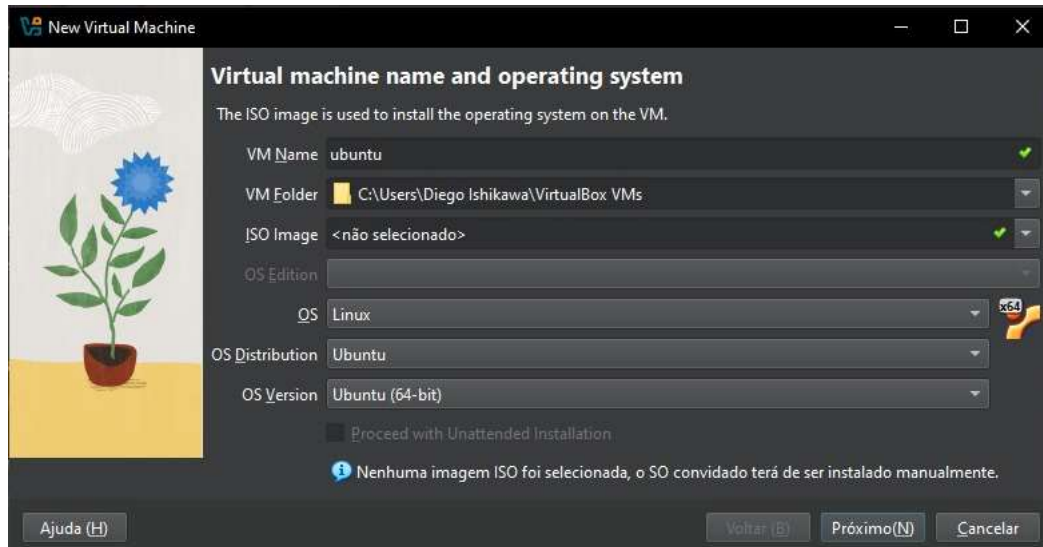
Comando utilizado: tasklist – Lista de processos em execução no Windows



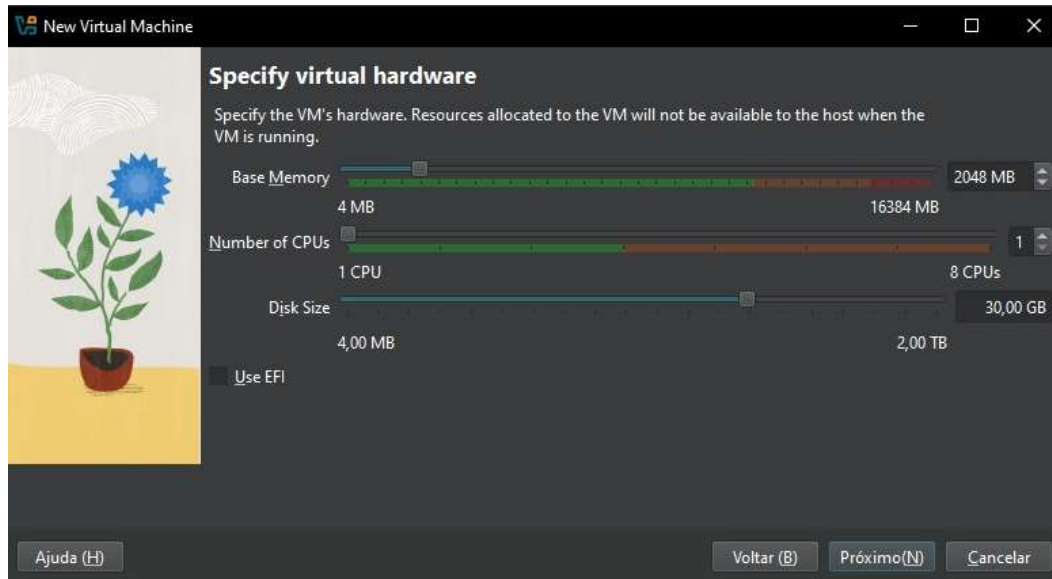
Tela do Gerenciador de Tarefas – Mostra em tempo real informações detalhadas

## 3.2 Linux

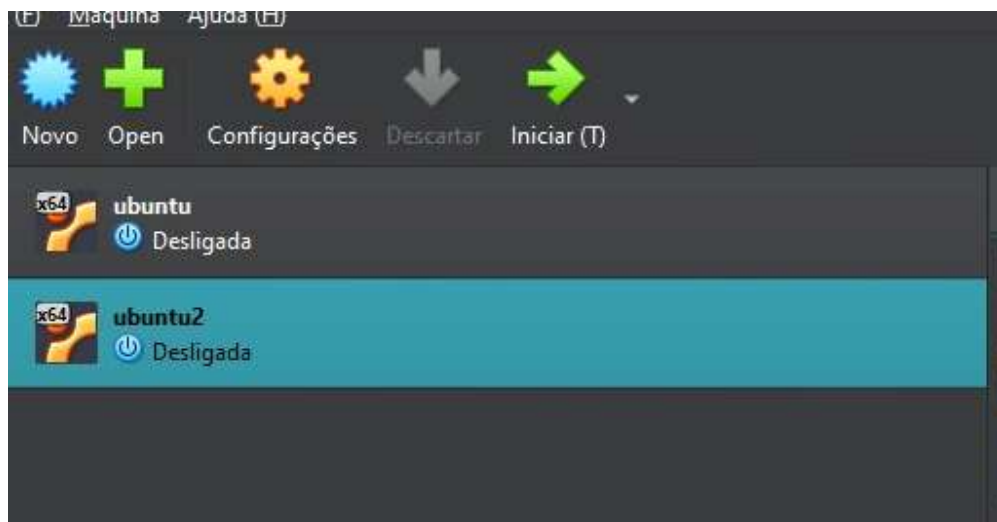
### 3.2.1 Instalação e Configuração



Tela de definição de Sistema operacional e nome da Máquina Virtual



Tela de especificação de Memória, processador e tamanho do Disco da Máquina

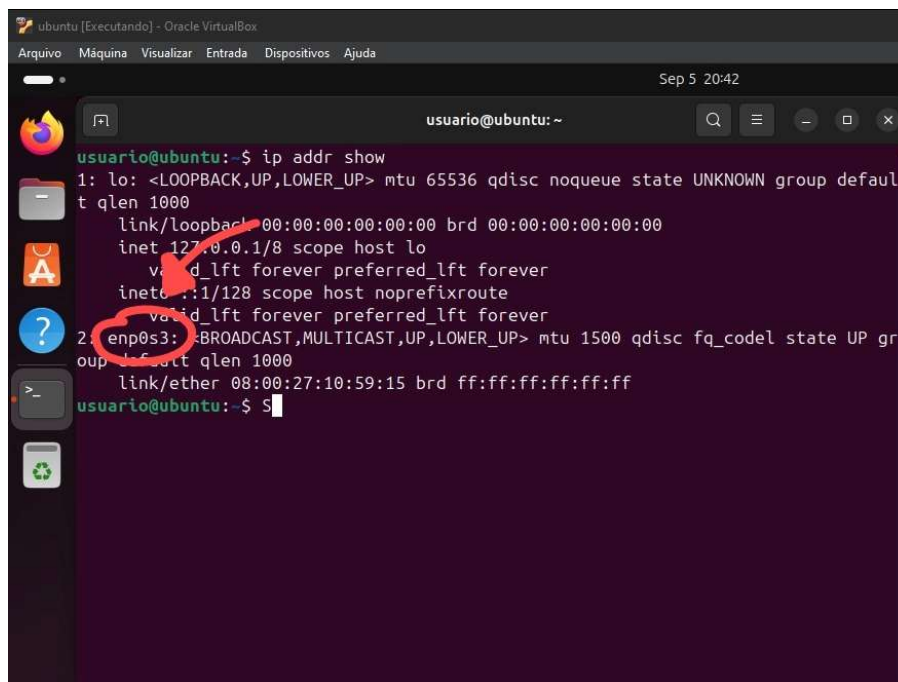


**Foram criadas duas máquinas para realização de testes de conectividade**



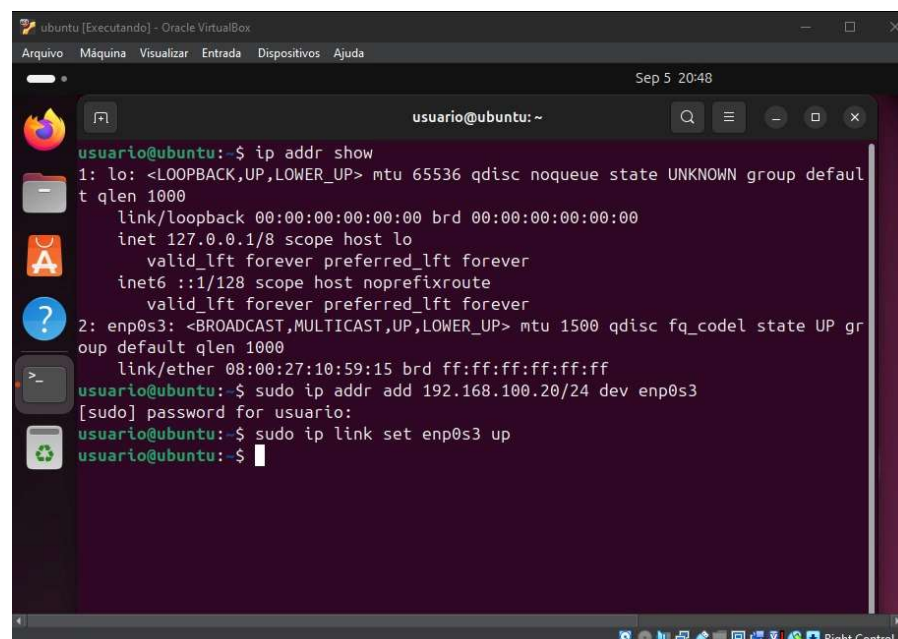
**Tela de definição da Rede a qual a máquina permanecerá ligada – No caso escolhemos Rede Interna para os mesmos testes de conectividade**

### 3.2.2 Testes e Comandos



```
usuario@ubuntu:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
    link/ether 08:00:27:10:59:15 brd ff:ff:ff:ff:ff:ff
usuario@ubuntu:~$
```

**Comando utilizado: ip addr show – Descobrir a interface de rede conectada**

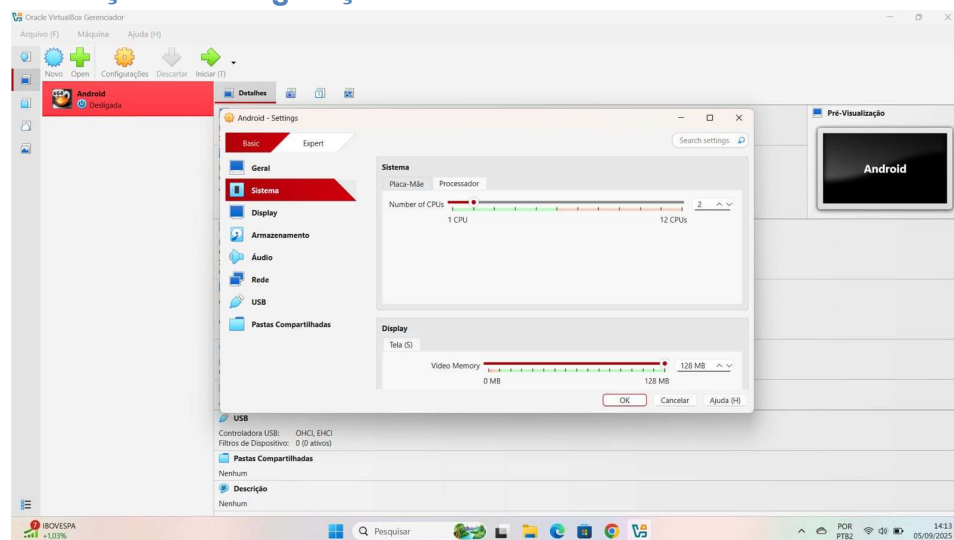


```
usuario@ubuntu:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default
    link/ether 08:00:27:10:59:15 brd ff:ff:ff:ff:ff:ff
usuario@ubuntu:~$ sudo ip addr add 192.168.100.20/24 dev enp0s3
[sudo] password for usuario:
usuario@ubuntu:~$ sudo ip link set enp0s3 up
usuario@ubuntu:~$
```

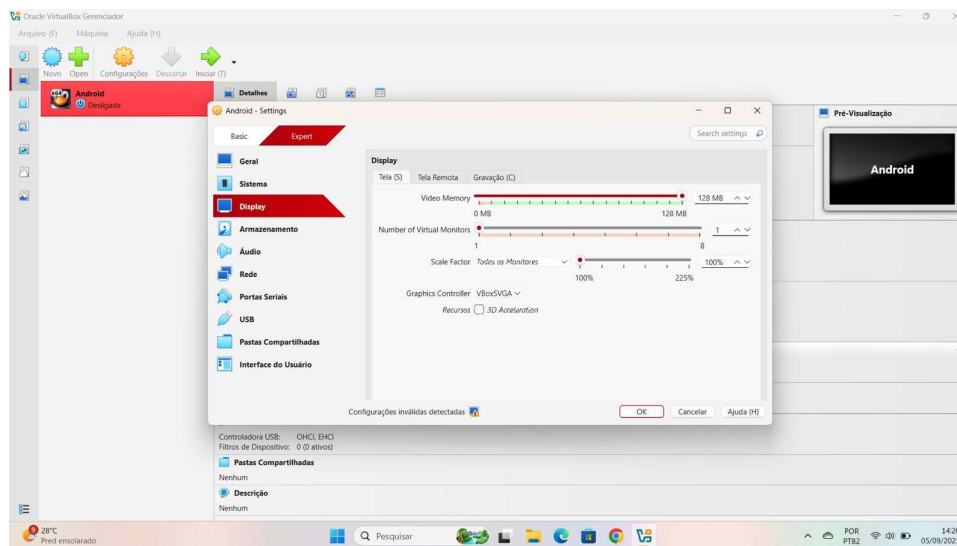
**Comandos utilizados: sudo ip addr add 192.168.100.20/24 dev enp0s3 – Definição de endereço IP para a máquina**  
**sudo ip link set enp0s3 up – Liga a Interface para funcionamento do IP definido**

## 3.3 Android

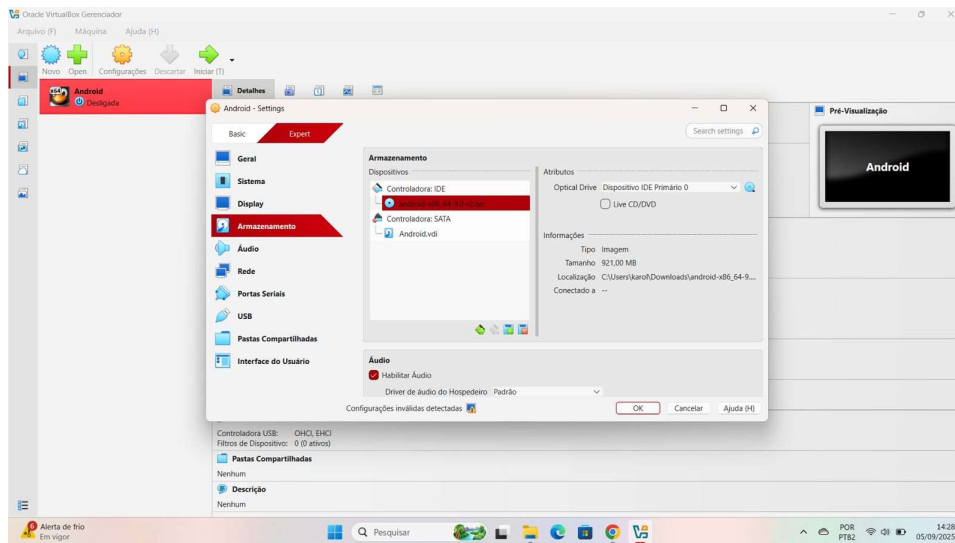
### 3.3.1 Instalação e Configuração



Tela de definição de quantidade de processadores

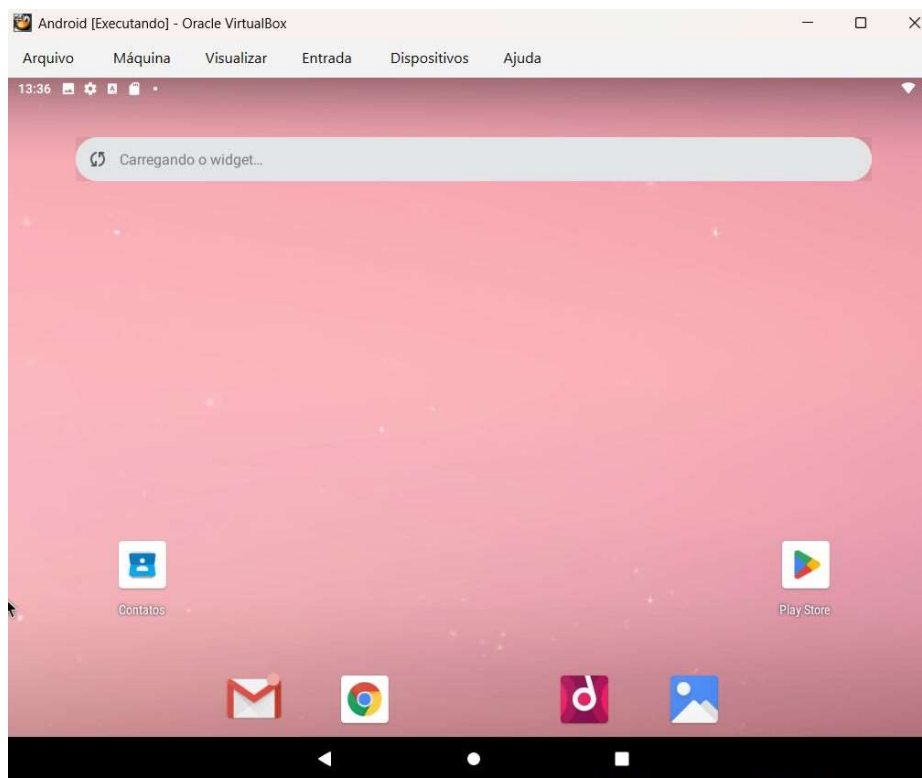


Tela de definição de Memória de Vídeo e Controladora Gráfica



**Tela de inclusão da ISO do Sistema Operacional**

### 3.3.2 Testes e Comandos



**Tela inicial do Android aberto e instalado**

```

cat /proc/cpuinfo
cpu MHz      : 2496.000
cache size   : 10240 KB
physical id  : 0
siblings     : 2
core id      : 0
cpu cores    : 2
apicid       : 0
initial apicid : 0
fpu          : yes
fpu_exception : yes
cpuid level  : 22
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xto
pology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 movbe popcnt aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid
d_single_ibrs_enhanced fsgsbase bmi1 avx2 bmi2 invpcid rdseed adx clflushopt sha_ni arat md_clear flush_l1d arch_capabilities
bugs         : spectre_v1 spectre_v2 spec_store_bypass swapgs
bogomips     : 4992.00
clflush size : 64
cache_alignm : 64
address sizes : 39 bits physical, 48 bits virtual
power management:

processor     : 1
vendor_id     : GenuineIntel
cpu family    : 6
model         : 186
model name    : 13th Gen Intel(R) Core(TM) i3-1315U
stepping      : 3
microcode     : 0xffffffff
cpu MHz      : 2496.000
cache size   : 10240 KB
physical id  : 0
siblings     : 2
core id      : 1
cpu cores    : 2
apicid       : 1
initial apicid : 1
fpu          : yes
fpu_exception : yes
cpuid level  : 22
wp           : yes
flags        : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xto
pology nonstop_tsc cpuid tsc_known_freq pni pclmulqdq ssse3 fma cx16 pcid sse4_1 sse4_2 movbe popcnt aes xsave avx f16c rdrand hypervisor lahf_lm abm 3dnowprefetch invpcid
d_single_ibrs_enhanced fsgsbase bmi1 avx2 bmi2 invpcid rdseed adx clflushopt sha_ni arat md_clear flush_l1d arch_capabilities
bugs         : spectre_v1 spectre_v2 spec_store_bypass swapgs
bogomips     : 4992.00
clflush size : 64
cache_alignm : 64
address sizes : 39 bits physical, 48 bits virtual
power management:

./ $

```

**Comando utilizado: cat /proc/cpuinfo – Mostra detalhes sobre o processador**

```

./ $ df -h
Filesystem      Size  Used Avail Use% Mounted on
tmpfs           996M   4.4M  992M   1% /
/dev/loop1      2.3G   2.1G  176M  93% /system
tmpfs           996M   740M  256M  75% /data
tmpfs           996M   316K  996M   1% /dev
tmpfs           996M     0  996M   0% /mnt
none            996M     0  996M   0% /cache
/dev/fuse       996M   740M  256M  75% /storage/emulated

./ $ free -m
              total        used        free      shared    buffers
Mem:           1992          1910           81         798          55
-/+ buffers/cache:          1855          137
Swap:            0             0             0

./ $ ps
USER      PID  PPID    VSZ   RSS  WCHAN          ADDR  S  NAME
u0_a69    19805 19783   9000   3040 sigsuspend      0     S  sh
u0_a69    20108 19805  11068   3072 0              0     R  ps

./ $

```

**Comandos utilizados: df -h: Mostra o espaço disponível**

**free -m: Mostra a memória usada e livre em MB**

**ps: Exibe a lista de processos em execução**

```
Android [Executando] - Oracle VirtualBox
Arquivo  Máquina  Visualizar  Entrada  Dispositivos  Ajuda

13:39
Janela 1

Tasks: 3 total, 1 running, 2 sleeping, 0 stopped, 0 zombie
Mem: 2040392k total, 1931564k used, 108768k free, 55284k buffers
Swap: 0k total, 0k used, 0k free, 1176132k cached
00%cpu 0%user 0%nice 0%sys 19%idle 0%iow 0%irq 0%host
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ ARGS
15783 u0_a69 10 -10 3.1G 148M 101M S 1.0 7.4 0:03.84 com.termoneplus
20118 u0_a69 10 -10 11M 3.1M 2.6M R 0.0 0.1 0:00.00 top
19805 u0_a69 10 -10 8.7M 2.9M 2.4M S 0.0 0.1 0:00.01 sh -
```

**Comando utilizado: top - Mostra CPU, memória e processos em tempo real**



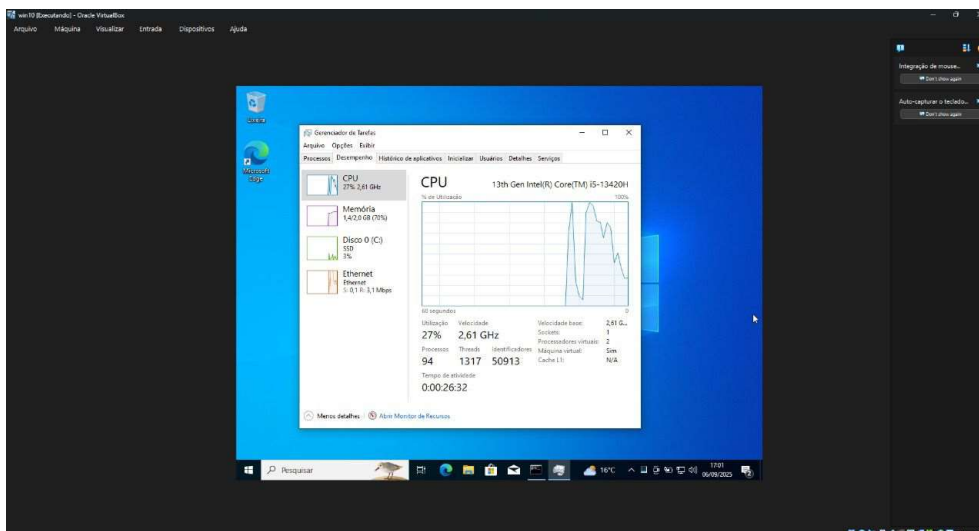
### 3.4 Comparação Crítica

Nos testes realizados, foi possível observar que os sistemas operacionais Windows, Linux e Android permitem verificar a rede e monitorar os processos, CPU e memória. O Windows se destaca pela interface gráfica e gráficos de desempenho, enquanto o Linux oferece um mais detalhes e maior controle pelo terminal. O Android, adaptado a dispositivos móveis, apresenta algumas limitações de acesso a processos. Assim, todos permitem monitoramento e configuração, variando apenas de interface para interface, na profundidade das informações apresentadas e no grau de controle oferecido ao usuário.

## 4. Análise Crítica

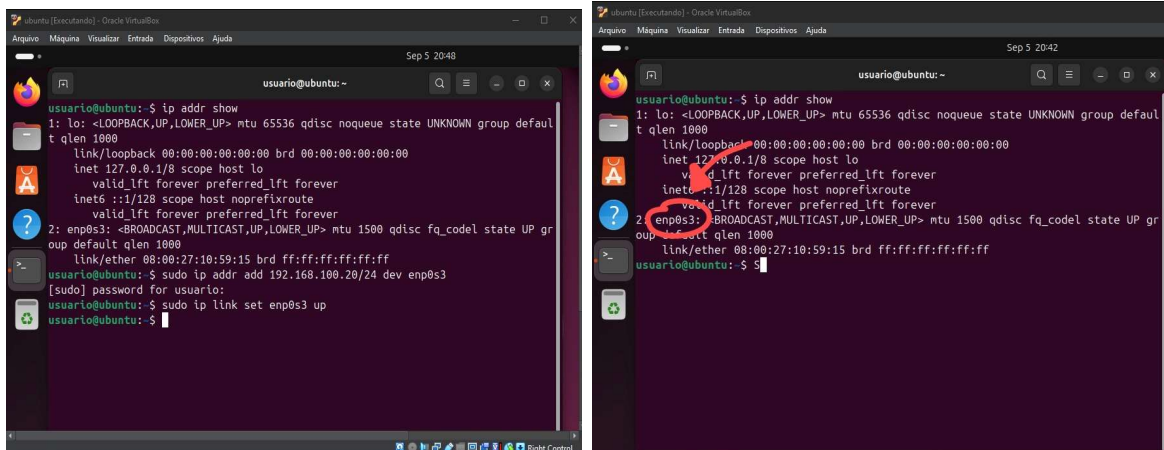
Cada sistema operacional possui uma filosofia de design diferente.

O Windows prioriza a facilidade de uso e acessibilidade, oferecendo interfaces gráficas intuitivas e ferramentas como o Gerenciador de Tarefas, que permitem monitorar CPU, memória e rede de forma visual.



Como podemos ver, através das interfaces gráficas do Windows, a visualização e monitoramento se tornam tarefas muito mais ágeis e facilitadas.

O Linux foca em flexibilidade, controle e eficiência, permitindo configurações detalhadas e monitoramento avançado via terminal com comandos como `top`, `free -m` e `ip addr`, além de possibilitar automação por scripts.



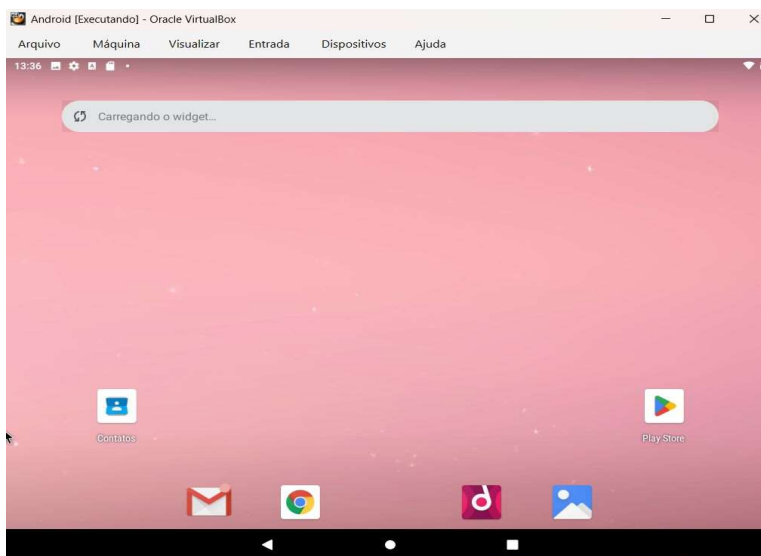
The image shows two side-by-side screenshots of a Linux terminal window. The left screenshot shows the output of the `ip addr show` command, displaying details for the loopback interface `lo` and the ethernet interface `enp0s3`. The right screenshot shows the same terminal window after running `sudo ip addr add 192.168.100.20/24 dev enp0s3` and `sudo ip link set enp0s3 up`, with the output of `ip addr show` now including the newly added IP address on `enp0s3`. A red circle highlights the `enp0s3` interface in the output of the second screenshot.

```
usuario@ubuntu:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:10:59:15 brd ff:ff:ff:ff:ff:ff
usuario@ubuntu:~$ sudo ip addr add 192.168.100.20/24 dev enp0s3
[sudo] password for usuario:
usuario@ubuntu:~$ sudo ip link set enp0s3 up
usuario@ubuntu:~$
```

```
usuario@ubuntu:~$ ip addr show
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host noprefixroute
        valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:10:59:15 brd ff:ff:ff:ff:ff:ff
    inet 192.168.100.20/24 scope host enp0s3
        valid_lft forever preferred_lft forever
usuario@ubuntu:~$
```

Como podemos ver, o Linux através do seu terminal permite uma eficiência maior e mais detalhada compara aos demais sistemas operacionais testados.

O Android, por ser uma interface voltada a uma diversidade muito maior de usuários, possui um design e informações mais limitadas, priorizando a simplicidade e a facilidade de uso em dispositivos móveis. Apesar dessas limitações, ainda permite monitoramento de recursos como CPU, memória e rede por meio de aplicativos ou comandos avançados via terminal, equilibrando controle técnico com acessibilidade para usuários comuns.



Como podemos ver, a interface do Android é intuitiva, facilitando o uso, especialmente para usuários sem experiência, permitindo acesso rápido a funções essenciais e monitoramento de recursos básicos sem necessidade de configurações muito complexas.

## 5. Conclusão

Nos testes realizados nos sistemas operacionais Windows, Linux e Android, foi possível verificar conectividade de rede, monitoramento de processos, CPU e memória. O Windows se destacou pela interface gráfica intuitiva e gráficos de desempenho em tempo real, que facilitam o acompanhamento de recursos. O Linux apresentou maior detalhamento e controle, permitindo monitoramento avançado e automação via terminal. O Android, voltado a dispositivos móveis e a uma diversidade maior de usuários, ofereceu interface simples e intuitiva, com acesso a informações essenciais de desempenho por aplicativos ou comandos via terminal.

Em relação ao objetivo, os testes deram a confirmação que cada sistema operacional atende a diferentes necessidades: Windows prioriza facilidade de uso, Linux prioriza controle e detalhamento técnico, e Android equilibra

simplicidade com acesso a recursos de monitoramento em dispositivos móveis. As descobertas mostram que a escolha do sistema depende do contexto de uso e do nível de controle desejado pelo usuário.

## 6. Autoavaliação

Durante o projeto, tivemos algumas dificuldades, por exemplo entender as interfaces diferentes dos sistemas operacionais, aprender os comandos do Linux e do Android, e entender os resultados dos testes de rede, CPU e memória. Cada um do grupo contribuiu de um jeito, fazendo pesquisas, realizando os testes, anotando os resultados e desenvolvendo o presente relatório, o que ajudou bastante na divisão das tarefas. Além disso, conversar entre nós mesmos trouxe diferentes pontos de vistas sobre o funcionamento e a forma como apresentar os testes. No geral, aprendemos muito sobre como cada sistema funciona, sobre os comandos e ferramentas de monitoramento, e também melhoramos nossas habilidades de analisar e comparar dados.

## 7. Referências

MICROSOFT. *Windows 10: Guia de referência rápida*. São Paulo: Microsoft, 2020.

MICROSOFT DOCS. *Comandos do Prompt de Comando do Windows*.

Disponível em: <https://docs.microsoft.com/pt-br/windows-server/administration/windows-commands/windows-commands>. Acesso em: 5 set. 2025.

MICROSOFT. *Guia de instalação e configuração do Windows 10*. Disponível em: <https://docs.microsoft.com/pt-br/windows/deployment/>. Acesso em: 4 set. 2025.

STALLMAN, Richard; et al. *Linux: Guia do usuário*. 2. ed. Rio de Janeiro: Novatec, 2019.

LINUX FOUNDATION. *The Linux Command Line*. Disponível em: <https://linuxjourney.com/>. Acesso em: 4 set. 2025.

UBUNTU. *Documentação oficial de instalação e configuração*. Disponível em: <https://ubuntu.com/tutorials/install-ubuntu-desktop>. Acesso em: 5 set. 2025.

ANDROID DEVELOPERS. *Android Developer Documentation*. Disponível em: <https://developer.android.com/docs>. Acesso em: 4 set. 2025.

ANDROID DEVELOPERS. *Ferramentas de linha de comando do Android*. Disponível em: <https://developer.android.com/studio/command-line/adb>. Acesso em: 4 set. 2025.

ANDROID DEVELOPERS. *Configuração de ambiente e SDK do Android*. Disponível em: <https://developer.android.com/studio/install>. Acesso em: 4 set. 2025.

MÉTODO DE COMANDOS LINUX. *Manual do Linux: Comandos essenciais*. Disponível em: <https://www.kernel.org/doc/>. Acesso em: 6 set. 2025.