RELATÓRIO 2 DO PROJETO

MÓDULO PROCESSOS DE INICIALIZAÇÃO

Disciplina: Sistemas Operacionais
Professor: Clóvis Ferraro
Grupo 15

Sumário

1. Introdução

A tecnologia vem se tornando cada vez mais avançada a necessidade de compreender os mecanismos de inicialização de sistemas operacionais torna-se indispensável.

Processos de inicialização juntamente com a instalação de sistemas operacionais é um dos fundamentos principais da computação moderna, mostrando a transição e dinâmica dos hardwares com os softwares. Este modulo tem como principal objetivo fazer a comparação e análise entre os processos de boot de sistemas operacionais distintos como Windows, Linus e Android com foco especial na identificação das camadas de firmware (BIOS/UEFI), estruturas de particionamento (MBR/GPT) e gerenciadores de boot.

A importância do conceito desse modulo é de entender como diferentes sistemas atuam em seus processos de inicialização, permitindo que tenhamos uma maior clareza e visão dessa etapa essencial para qualquer dispositivo. Através de comandos nativos dos sistemas este trabalho busca evidenciar na prática as diferenças arquiteturais entre os sistemas.

2. Metodologias

- Windows

Para analisar o processo de boot e estrutura de disco neste equipamento, foram utilizados comandos nativos do Windows para identificar o tipo de firmware, tabela de partição, layout do disco e configuração do bootloader.

- Linux

Foram realizados os testes em modelo no Linux Lite 7.6. Foram executados comandos para identificar o firmware, tabela de partição, estrutura de arquivos e configuração do gerenciador de boot GRUB.

- Android

Para investigar o processo de inicialização (boot) e a estrutura de disco do emulador Android, foram utilizados comandos nativos do shell do sistema através do Terminal do Android Studio.

Os principais pontos analisados foram:

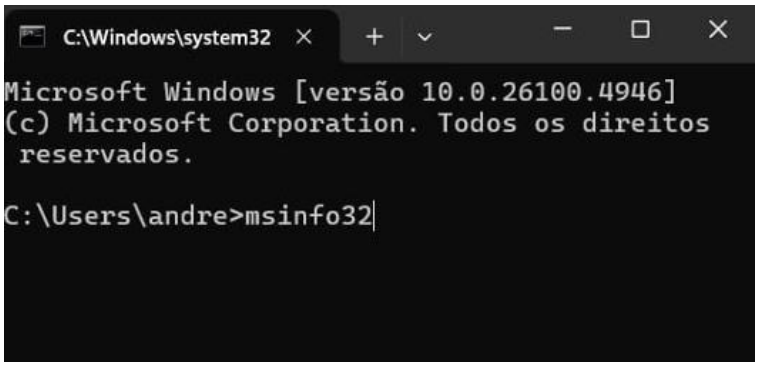
- Sistema de arquivos e espaço em disco (df -h);
- Listagem de dispositivos de bloco (ls -l /dev/block);
- Informações de propriedades de boot (getprop | grep boot);
- Versão do kernel (uname -a);
- Pontos de montagem e tipos de partição (mount).

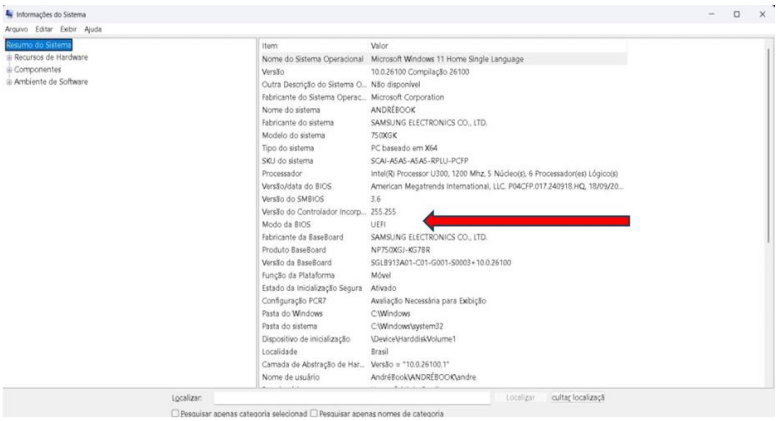
3. Comparação entre os Sistemas Operacionais

3.1 Windows

- Comando usado para acessar o terminal: “ Win + R”
- 

Análise Firmware (BIOS/UEFI)
• Comando usado: **msinfo32**





Através desse comando pode ser verificado o “Modo de BIOS” do computador, podemos notar que o computador segue um modelo UEFI, sendo um firmware moderno que suporta Secure Boot e discos de grande capacidade.

Tabela de partição (MBR/GPT)

• Comando usado: **Get-Disk**

```
PS C:\Users\andre> Get-Disk

Number Friendly Name                                        Serial Numbe
r                    HealthStatus         OperationalStatus      Tota
l Size Partition


       Style
------ -------------                                        ------------
-                    ------------         -----------------      ----
------ ----------
0      SAMSUNG MZVL4256HBJD-00B                             0025_3881_F1
19_6EA8.             Healthy              Online                    238
.47 GB GPT
```

Utilizando o comando **Get-Disk**, identificou-se que o disco principal utiliza o esquema **GPT**, compatível com o firmware UEFI e que não possui as limitações do MBR.

Análise do Layout de Partições

• Comando usado: **Get-Partition + Get-Volume**

```
PS C:\Users\andre> Get-Partition


   DiskPath: \\?\scsi#disk&ven_nvme&prod_samsung_mzvl4256#5&e359669&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                            1048576                                    260 MB System
2                            273678336                                   16 MB Reserved
3                C           290455552                                 220.14 GB Basic
4                            236664651776                               850 MB Recovery
5                            237555941376                             16.23 GB Recovery
6                            254986420224                                1 GB Recovery
```

```
PS C:\Users\andre> Get-Volume

DriveLetter FriendlyName     FileSystemType DriveType HealthStatus OperationalStatus SizeRemaining      Size
----------- ------------     -------------- --------- ------------ ----------------- -------------      ----
C                            NTFS           Fixed     Healthy      OK                    94.14 GB 220.14 GB
            SAMSUNG_REC      FAT32          Fixed     Healthy      OK                   374.16 MB   1020 MB
            Windows RE tools NTFS           Fixed     Healthy      OK                   213.3 MB    850 MB
            SYSTEM           FAT32          Fixed     Healthy      OK                   200.39 MB    256 MB
            SAMSUNG_REC2     NTFS           Fixed     Healthy      OK                     1.7 GB  16.23 GB
```

Com os comandos Get-Partition e Volume, nos é mostrado a estrutura do Windows, uma partição **EFI** para o bootloader, partições de **Recuperação** e a partição principal **Basic** formatada como **NTFS** (Figura 3), onde o sistema operacional está instalado.

Análise de gerenciador Boot
• Comando usado: **bcdedit**

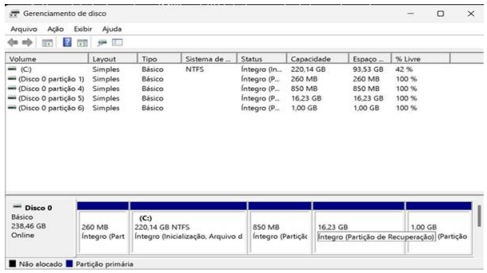
```
Gerenciador de Inicialização do Windows
---------------------------------------
identificador           {bootmgr}
device                  partition=\Device\HarddiskVolume1
path                    \EFI\Microsoft\Boot\bootmgfw.efi
description             Windows Boot Manager
locale                  pt-BR
inherit                 {globalsettings}
default                 {current}
resumeobject            {1ec44efa-91d9-11ef-8161-8cd459daea15}
displayorder            {current}
toolsdisplayorder       {memdiag}
timeout                 30

Carregador de Inicialização do Windows
--------------------------------------
identificador           {current}
device                  partition=C:
path                    \Windows\system32\winload.efi
description             Windows 11
locale                  pt-BR
inherit                 {bootloadersettings}
recoverysequence        {1633787b-91bc-11ef-b795-0072eec6948f}
displaymessageoverride  Recovery
recoveryenabled         Yes
isolatedcontext         Yes
allowedinmemorysettings 0x15000075
osdevice                partition=C:
systemroot              \Windows
resumeobject            {1ec44efa-91d9-11ef-8161-8cd459daea15}
nx                      OptIn
bootmenupolicy          Standard
PS C:\Users\andre>
```

O comando **bcdedit** mostra detalhadamente a configuração Windows Boot Manager mostrando o caminho para o carregador de boot e a partição de boot, finalizando o ciclo de inicialização do sistema.

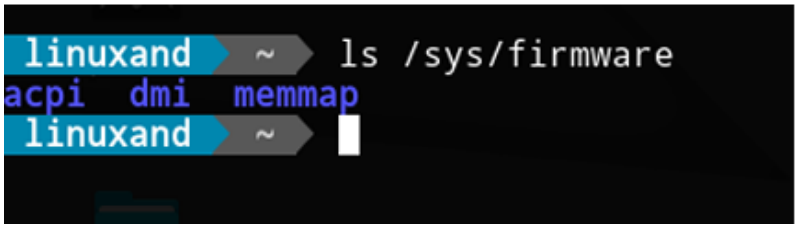
Gerenciamento de Disco Gráfico
• Ferramenta: **diskmgmt msc**.

• A ferramenta de visualização gráfica **Gerenciamento de Disco** forneceu uma visualização intuitiva da mesma estrutura de partições, validando as informações obtidas anteriormente com maior clareza.



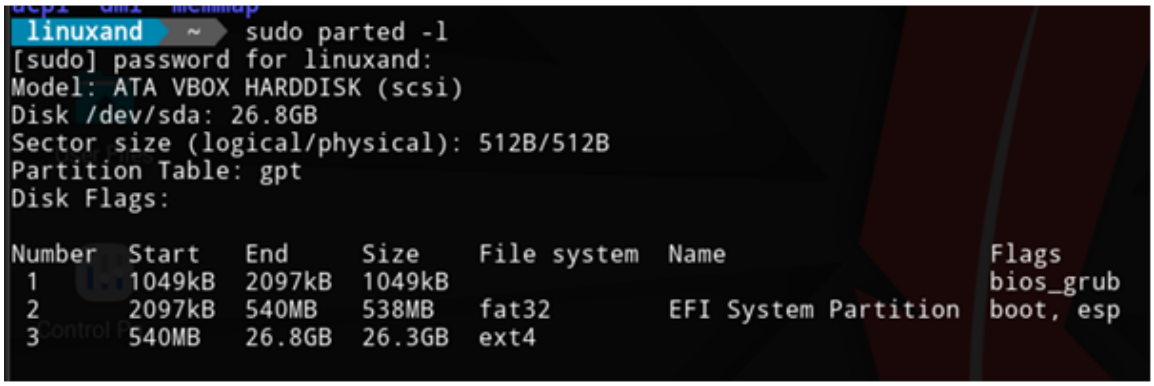
3.2 Linux

Análise do Firmware (BIOS/UEFI)
• Comando usado: **ls /sys/firmware**



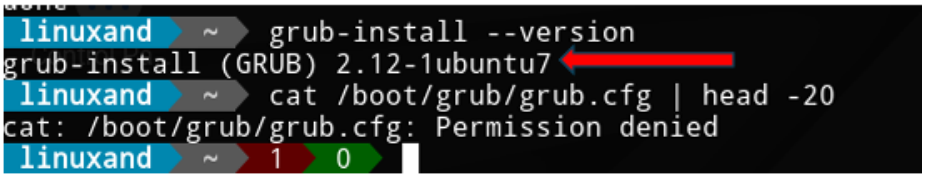
A presença apenas dos diretórios acpi, dmi e memmap confirmam que o sistema opera em modo BIOS, característico de hardware mais antigo.

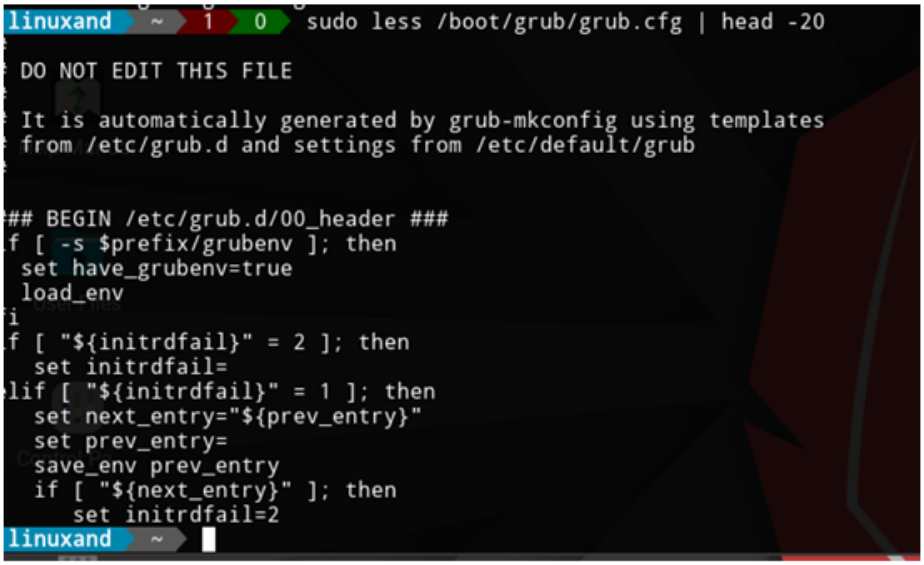
Análise da Tabela de Partição e Discos
• Comando usado: **sudo parted -l**



O comando revela uma situação interessante: mesmo operando em modo BIOS, o sistema utiliza o esquema moderno GPT como tabela de partição. Isso é possível graças ao suporte ao GPT por parte da BIOS. Mostrando uma configuração híbrida que aproveita vantagens do modelo GPT.

Análise do Gerenciador de Boot GRUB
• Comandos usados: **grub-install --version e sudo less /boot/grub/grub.cfg**



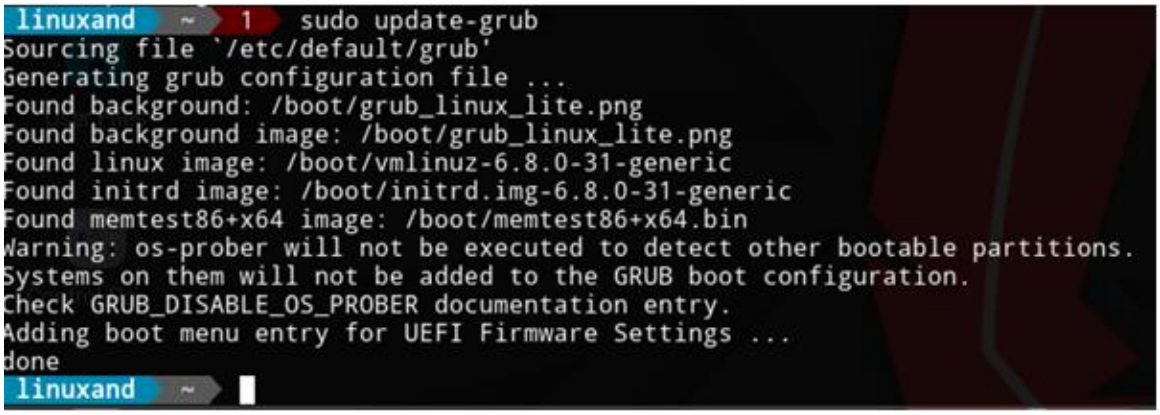


Ao utilizar o comando: **grub-install –version**, é possível notar que o sistema utiliza uma versão **Grub 2.12** como gerenciador boot principal, mostrando a compatibilidade do gerenciador boot com a distribuição do Linux Lite.

A análise do arquivo **/boot/grub/grub.cfg** confirmou sua natureza automatizada, gerado a partir de templates em **/etc/grub.d/** como o aviso **“DO NOT EDIT THIS FILE”**.

Operação com o GRUB

• Comando usado: **sudo update-grub**



A execução do comando **sudo update-grub** demonstrou o processo de varredura do sistema em busca de kernels e sistemas operacionais, atualizando dinamicamente o menu de boot com as entradas encontradas.

3.3 Android

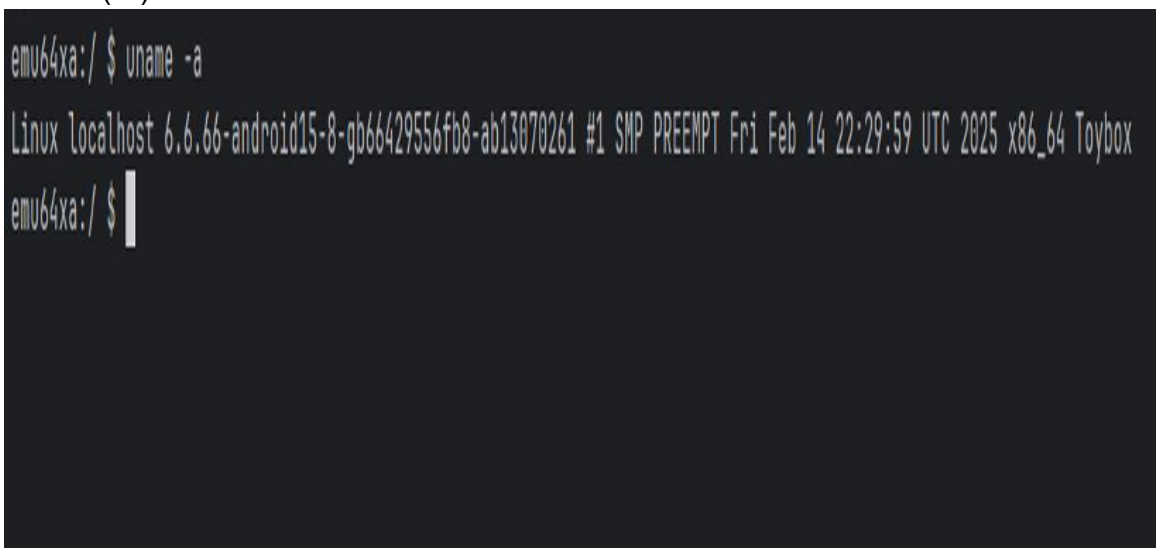
Acesso ao Terminal

O terminal foi aberto diretamente dentro do Android Studio, conectado ao emulador Android Virtual Device (AVD). O prompt exibido foi: emu64xa:/ $

Análise do Espaço em Disco

Comando usado: df –h

Mostrou todas as partições montadas, seus tamanhos e percentuais de uso. Identificou-se que a partição /data possui cerca de 5.8 GB totais, com 1.7 GB usados (32%). Também foi possível ver /system montado em modo somente leitura (ro).

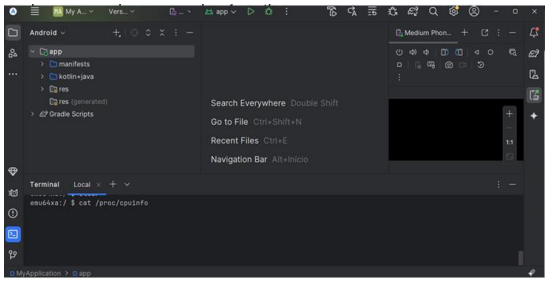


Listagem dos Dispositivos de Bloco

Comando usado: ls -l /dev/block

Mostrou os dispositivos virtuais de armazenamento (dm-0, dm-1, …) usados pelo sistema Android.

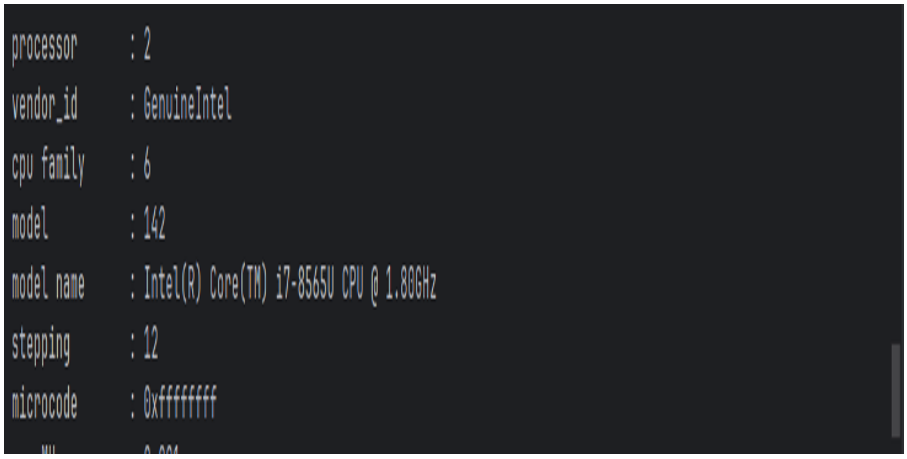
Cada dispositivo corresponde a uma partição lógica do sistema.



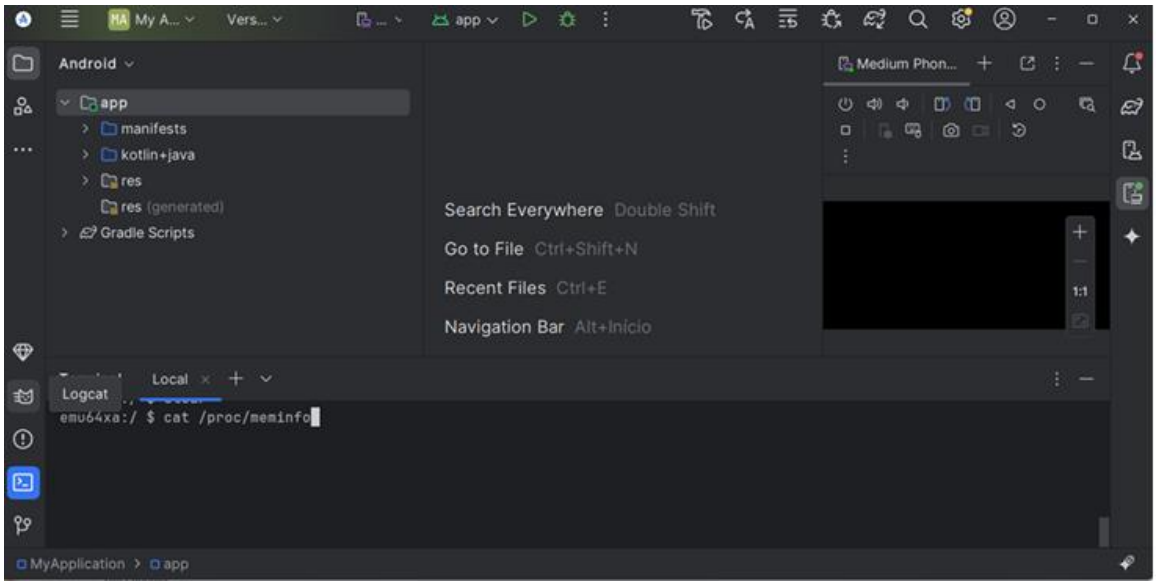
Propriedades do Boot

Comando usado: getprop | grep boot

Exibiu propriedades de inicialização do sistema. Mostrou o número de série do emulador (ro.boot.serialno) e o hash da imagem de boot (ro.boot.vbmeta.digest). Indicou que o sistema usa qemu (emulador) e Secure Boot ativo.



Informações do Kernel
Comando usado: uname -a
Mostrou detalhes sobre a versão do kernel Linux usado pelo Android no emulador.



Informações de Memória e CPU
Comando usado: cat /proc/meminfo e cat /proc/cpuinfo
Esses comandos exibiram informações sobre a memória disponível no

dispositivo e sobre o processador em uso pelo emulador Android.

Pontos de Montagem
Comando usado: mount
Exibiu todos os sistemas de arquivos montados e seus modos (ro/rw). Confirmou que a raiz (/) está montada em erofs (somente leitura) e /data em ext4 com acesso de leitura e escrita.

3.4 Comparação Crítica

Windows: Utiliza um fluxo moderno de inicialização UEFI → GPT → Windows Boot, facilmente identificado via terminal e comandos nativos do sistema.

Linux: Adota um fluxo mais tradicional baseado em BIOS, mas com elementos modernos como partições GPT, gerenciadas pelo GRUB 2.12, integrando características híbridas de inicialização.

Android: No emulador, segue o fluxo Bootloader → Kernel Linux → Partições (erofs/ext4) → Sistema Android, permitindo observar claramente a organização de partições e a inicialização do SO via terminal do Android Studio.

4. Conclusão

Os testes realizados em Windows, Linux e Android permitiram analisar seus fluxos de inicialização e organização de partições.
-> O Windows apresenta fluxo moderno: UEFI → GPT → Windows Boot, eficiente e de fácil acesso via terminal.
-> O Linux mantém fluxo tradicional baseado em BIOS, mas incorpora partições

GPT gerenciadas pelo GRUB 2.12.
-> O Android adapta o kernel Linux e partições erofs/ext4, iniciando pelo Bootloader até o sistema Android.

Cada sistema reflete seu ambiente de uso: desktops, servidores/PCs flexíveis e dispositivos móveis. A análise confirma que terminais e linhas de comando são eficazes para estudar inicialização e estrutura de partições.

5. Autoavaliação e Dificuldades

5.1 Principais Dificuldades

- **Complexidade de comandos:** A dificuldade inicial enfrentada pelo grupo foi de interpretar e se familiarizar com os comandos principais dos tetes, devido sua alta variedade entre os três ambientes, exigindo uma curva de aprendizado para compreender os parâmetros e saídas de cada um.
- **Interpretação das saídas:** Ao realizar os comandos e visualizarmos as saídas que ocorreram, especialmente em comandos como: **sudo less /boot/grub/grub.cfg,** foi possível notar uma maior dificuldade de interpretação por parte do grupo necessitando de pesquisa adicional para decodificar informações técnicas.
- **Configuração Híbrida:** Durante os testes com o sistema Linux, foi identificado um firmware BIOS utilizando tabela de partição GPT, o que inicialmente foi visto como algo contraintuitivo, sendo necessário maiores pesquisas em volta de conceitos como o GRUB.

5.2 Autoavaliação

O grupo demonstrou capacidade de aprender e aplicar novos comandos e conceitos técnicos mais complexos com uma maior facilidade, além de uma maior familiaridade com o uso das linhas de comando terminal dos sistemas testados o que aumenta a eficiência dos testes.

Entretanto é necessário o aprimoramento no conhecimento em comandos essenciais para o recolhimento de dados como os coletados no módulo. O principal aprendizado foi entender que o processo de boot é uma troca complexa entre o firmware, tabela de partições e gerenciador de boot, onde cada componente deve ser compatível com o outro, além de aprendermos que os conceitos não são totalmente sólidos, mostrado quando um sistema pode usar BIOS com GPT, quebrando a lógica simplista de que BIOS sempre equivale a MBR.

Sendo assim conhecimentos postos em prática e visualizados diretamente pelo grupo e consolidados em nosso pacote de habilidades essenciais que nos ajudara em projetos e módulos futuros.

6. Refêrencias

BIOS e UEFI: https://youtu.be/dspL2kP1XyY?si=dvEmgHqaaSq8S8sf

UEFI: https://youtu.be/PI1atKjmDsc?si=Jm56bPpnDPDiUh1M

Partições: https://youtu.be/UNVRKBF4Iz4?si=zySMaxOXV0WNFfJ_

Boot Linux: https://youtu.be/Ok-L4dfkvsk?si=23cNLIxKttW7Wv78