

# Sistemas Operativos

2º ano – 1º sem.  
2022 – 2023

**Conceitos sobre inicialização da máquina e preparação do sistema operativo**

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Tópicos

---

Bootstrapping  
BIOS e firmware  
Partições, sector boot, MBR  
Dualboot  
UEFI e GPT

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## O que distingue o SO dos restantes programas?

---

### Processo de arranque do sistema (“bootstrap”)

- Trata-se de uma questão de importância central

Exemplo de questão relacionada com o processo de arranque

O sistema operativo age como gestor da máquina e tem mais “poderes” do que qualquer aplicação regular (ditas aplicações utilizador).

- *Porquê? O que lhe dá esses “poderes” privilegiados?*

## O que distingue o SO dos restantes programas?

---

*O que lhe dá “poderes” especiais ao sistema operativo?*

- O **processador** assume características e capacidades diferentes **consoante está a executar código do SO ou de aplicação**.
- O processador faz essa distinção com base nas **zonas de memória** em que as instruções dos programas se encontram
  - As instruções do sistema operativo encontram-se numa zona de memória que o processador automaticamente reconhece como tendo mais privilégios – aceita executar determinadas instruções necessárias ao controlo da máquina
  - As instruções das meras aplicações estão em zonas de memórias nas quais o processador se recusa a executar certas instruções (as que são necessárias para controlar a máquina)
  - (Isto vai ser visto mais adiante no semestre)
- Inicialmente a máquina encontra-se num estado “configurável”. O software que executa em primeiro lugar pode configurar a máquina, limitando todo o software que se executa depois

## O que distingue o SO dos restantes programas?

---

É importante que o sistema operativo esteja entre os primeiros programas a executar para poder assumir o controlo configurando esta questão das zonas de memória

- Excluindo o *firmware*, o SO é o **primeiro programa completo** a ser carregado para a máquina e **configura o hardware de forma a atribuir a si mesmo as capacidades totais do hardware**.
  - Todos os restantes programas (aplicações) são executados pelo sistema operativo que os coloca num **ambiente de execução controlado com menos poderes sobre a máquina**.
  - Basicamente, o sistema chega à máquina primeiro e “arranja” o cenário (configura a máquina) de forma a quem os programas tenham apenas os privilégios que o sistema entende dar.
  - É então importante perceber como tudo começa: como é que o SO é carregado

-> Assunto deste conjunto de slides: **como se processa a inicialização da máquina e o carregamento do sistema operativo?**

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Carga (“boot”) do sistema operativo

---

Conceitos necessários

- *Firmware*, BIOS, POST
- Discos, Partições, MBR, Sector boot
- Bootstrapping, Boot loader, chainloading
- UEFI, GPT, Secure boot

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Firmware

---

### Firmware

- Software armazenado de forma permanente (não precisa de alimentação permanente), inicialmente ROM, agora memória flash
- Contém rotinas utilitárias para controlar aspectos do equipamento e rotinas para inicializar esse equipamento e colocá-lo num estado inicial coerente: *Basic Input Output System* (BIOS)
- Efectua um teste simplificado ao equipamento: *Power On Self Test* (POST)
- Em equipamentos simples, pode conter a totalidade do software necessário à operação do dispositivo.

### Nos computadores habituais

- Contém rotinas de arranque inicial da máquina, rotinas para interacção com dispositivos standard (ex.: discos, teclado, etc.)
- Corre normalmente em modo simplificado (não *privilegiado*) do processador e as suas capacidades de gestão são limitadas.
- Inclui normalmente o software habitualmente designado por BIOS

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## BIOS

---

### BIOS (Basic Input Output System)

- Parte habitual do software habitualmente designado de *firmware*.
  - Chamar “BIOS” ao firmware é errado (mas é um erro muito comum)
- Parte da norma estabelecida para os IBM-PC e compatíveis.
- Apresenta algumas limitações face ao equipamento moderno e está em processo de substituição (foi substituída) pela norma UEFI
- Contém software para
  - ***Interação com dispositivos standard normalmente presentes em qualquer máquina.***
  - Carregar ou iniciar a carga do sistema operativo

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Tarefas da BIOS

---

### Inicialização da máquina

- **Identificação e teste** dos componentes principais da máquina (ex., memória, teclado, etc.). Tarefa normalmente designada por POST (Power On Self Test)
- Enumeração dos dispositivos presentes e configuração de cada um de acordo com parâmetros standard
- Passagem do controlo (da execução) ao software que inicia a carga do sistema operativo (início do processo de *bootstrap*) e que normalmente se encontra em disco

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Tarefas da BIOS

---

### Rotina de interação com dispositivos standard

- Materializado sob a forma de um conjunto de rotinas acessíveis ao sistema operativo e a qualquer outro programa
- Podem agir como substituto de funções sistema em cenários em que o sistema operativo é muito simples  
Exemplo: IBM PC/MSDOS.
  - Rotinas da BIOS (exemplos): acessíveis via int 10H, int 09H, etc.
- Limitadas na sua ação por:
  - Estarem preparadas para dispositivos standard, e portanto *não aproveitam capacidades específicas ou otimizadas de hardware melhor.*
  - Correrem habitualmente com o processador em modo não privilegiado (implica menor capacidade de ação)
- Estas rotinas são essenciais ao processo de arranque e configuração da máquina, sendo apenas dispensáveis se o sistema operativo tiver software que as substitua e apenas depois deste estar presente em memória

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Tarefas da BIOS - Bootstrapping

---

### Início do processo de bootstrap

- Leitura dos parâmetros de configuração da máquina em memória não volátil para determinação de qual o dispositivo por onde se inicia a carga do sistema (ex.: disco, CD, USB, etc.)
- Análise do dispositivo em questão (o disco, CD, *pen*, o que for) e leitura do **boot loader** para memória
- Passagem do controlo da execução para o **boot loader** que foi carregado. Deste ponto em diante a BIOS age essencialmente como repositório de rotinas utilitárias. O processo de **bootstrap** prossegue com a execução do **bootloader**.

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Bootstrapping

---

### **Bootloader**

- Pequeno programa armazenado em memória secundária que começa o processo de carga do sistema
- Razoavelmente *standard* quanto à localização e formato mas ainda assim algo dependente do sistema operativo e bastante dependente da arquitectura
  - Tamanho standard típico (norma MBR): menos que 512 bytes.  
Sendo um tamanho bastante pequeno, o **bootloader** tipicamente carrega outros programas (cada vez mais complexos e dependentes do sistema operativo) e transfere o controlo a esses programas. O efeito é o de uma sequência de peças de dominó a em que cada uma empurra a seguinte.
  - Locais típicos: **início do disco** , **início da partição activa do disco**
- A sua localização depende do tipo de dispositivo
  - Os exemplos e descrição seguintes assumem que o dispositivo é um disco rígido (cenário mais habitual)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Discos rígidos

### Organização dos discos rígidos

- Os discos rígidos estão normalmente organizados em **partições**.
  - Cada partição pode ter um sistema de ficheiros diferente
  - Cada partição pode hospedar sistema operativo independente
  - Alguns sistemas de ficheiros conseguem abarcar várias partições dando a ideia da existência de um só sistema de ficheiros

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Discos rígidos

### Organização dos discos rígidos

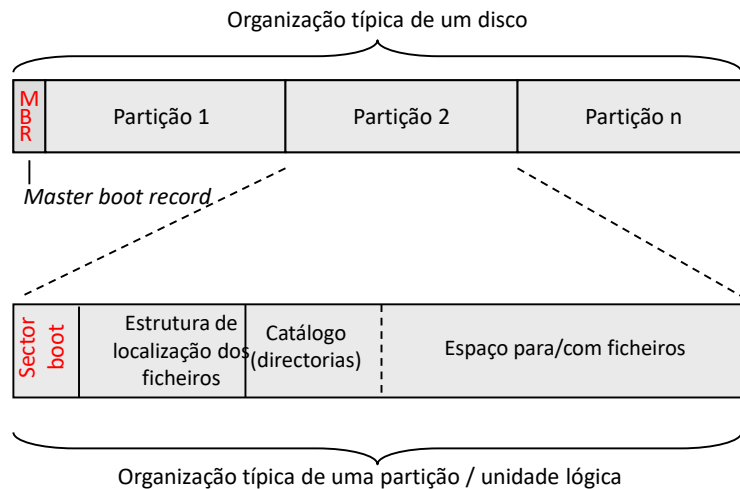
- O particionamento dos discos (e do processo de bootstrap) é dependente de normas
- Existem duas normas principais
  - **BIOS/MBR** – original do IBM PC mas limitada e a cair em desuso
    - O nome é artificial (justaposição de partes) e foi dado retroativamente
    - Ainda bastante usada em sistemas pequenos ou de 32 bits
  - **UEFI** – Mais recente e poderosa, na prática a mais usada hoje em dia
    - Nome: *Unified Extensible Firmware Interface*

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Discos rígidos – Lógica Bios/MBR



DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Discos rígidos - Partições

### MBR – Master boot record

- Normalmente um sector (pode ser mais), sendo o primeiro sector do disco
- Contém um pequeno programa que é um *boot loader*.
  - Este programa pode, em teoria, carregar o sistema directamente, mas normalmente passa o controlo para um outro *boot loader* que se encontra no sector boot da partição activa
- Mantém informação acerca da geometria do disco:
  - Quantas partições existem
  - Onde começa e acaba cada partição
  - Qual a partição que está activa (qual a que contém o S.O. de arranque)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães



## Sistemas de ficheiros – Partições: constituintes

### **Sector *boot***

- Normalmente 1 sector (pode ser mais), sendo o primeiro sector de uma partição
- Contém um pequeno programa que é um boot loader e dá início ao arranque do sistema (se a partição for a activa)
  - Normalmente o sistema que está nessa partição, mas pode ser outro
  - Este boot loader é, normalmente, específico ao sistema operativo, enquanto que o boot loader no MBR é, normalmente, mais genérico.
  - A instalação de boot loader deve ser feita, sempre que possível, na partição do sistema operativo em questão, deixando o MBR intacto tanto quanto possível uma vez que esse diz respeito ao disco todo.
- Contém informação variada acerca da partição
  - Localização acerca das outras componentes
  - Tamanho dos blocos lógicos (*clusters*)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Bootstrapping

### **Processo de carga do sistema**

1. A BIOS identifica o dispositivo onde deve procurar o sistema operativo (informação armazenada na configuração em memória não volátil)  
Assume-se nestes exemplos de que se trata de um disco rígido
2. A BIOS lê o MBR para memória e transfere a execução para o programa que se encontra dentro do MBR.
3. O programa no MBR determina qual a partição activa (tabela existente no MBR), lê o sector boot dessa partição para memória e transfere-lhe a execução.
4. O programa no sector boot prossegue a carga do sistema carregando os ficheiros do sistema presente na sua partição, ou apresentando um menu possibilitando passar para outro sector boot de outra partição (dual boot)  
O processo de um sector boot carregar o sector de outra partição e passar-lhe o controlo é designado de *chainloading*

Tanto o programa do MBR como o do sector boot da partição podem apresentar opções de escolha para dual boot. Normalmente esta escolha é feita a nível do sector boot pois as alterações no MBR afectam o disco todo (por oposição de afectar apenas uma partição) e portanto são de evitar tanto quanto possível

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Dual boot

Consiste em ter vários sistemas operativos na máquina (normalmente em partições diferentes) e ter a possibilidade de arrancar a máquina com qualquer um desses sistemas operativos (em alternativa)

### Método de *dual boot* em BIOS/MBR

Configuração do *boot loader* no sector *boot* para apresentar um menú que possibilite

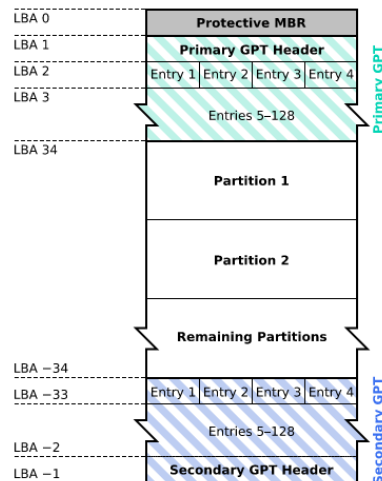
- Carregar os ficheiros do sistema presentes na partição em questão
  - Ou então
  - Duplicar o procedimento efectuado pelo programa no MBR e
    1. Carregar um outro sector *boot* para memória
    2. Passar a execução para o código desse outro sector *boot*
- Este procedimento é conhecido com *chainloading*

## Norma UEFI

### UEFI – Unified Extensible Firmware Interface

- Norma que substitui a anterior (BIOS/MBR)
- Sendo mais moderna, permite um conjunto alargado de funcionalidades mais em linha de conta com o hardware recente, por exemplo:
  - Discos GPT (GPT = **G**UID **P**artition **T**able, GUID = **G**lobal **U**nique **I**Dentfier)
    - São discos normais particionados segundo um esquema diferente daquele possível com o MBR.
  - Secure boot
    - Utilização de certificados digitais nos *loaders* para controlar o acesso de um sistema à máquina. Um sistema que não tenha um certificado válido quando comparado com os que estão presentes na memória da máquina, será impedido de se carregar e executar.
    - Os certificados podem ser adicionados e a opção de secure boot pode (eventualmente) ser desligada (acções do administrador)
  - Rotinas de interface com hardware mais evoluídas de carácter semelhante a *device drivers* em linguagem independente do processador
  - Suporte melhorado para placas gráficas recentes logo ao início

## Organização de discos rígidos GPT



### Discos GPT

- Possíveis em *chipsets* e sistemas compatíveis com UEFI
- GPT: substitui a lógica antes atribuída ao MBR, mas essencialmente é a mesma coisa com mais capacidades e flexibilidade
- Permite 128 partições
- O arranque do sistema pode ser feito de forma mais flexível e através de pequenos programas (loaders) existentes na primeira partição (designada de "partição de sistema")
- Suportam características mais evoluídas tais como certificados de **secure-boot**

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Exemplo Dual Boot Linux + Windows em UEFI

Neste exemplo

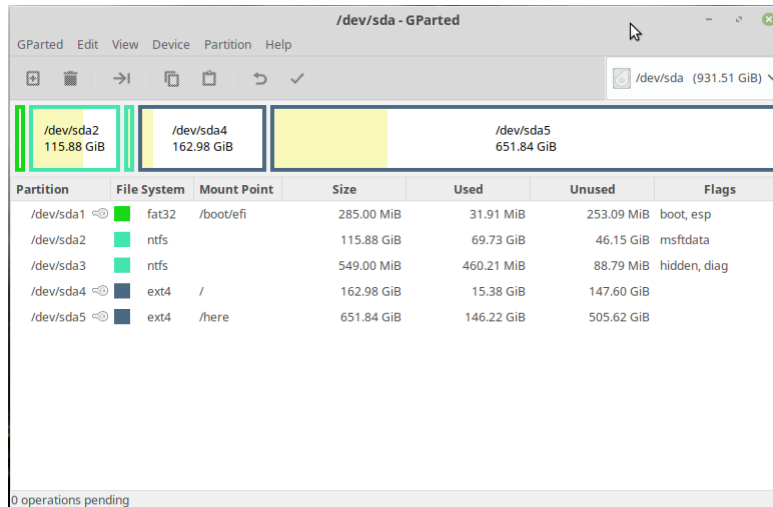
- A máquina está configurada com UEFI
- Existem dois sistemas instalados: Ubuntu Mint e Windows 10.
  - Existia um outro (Debian) que foi removido mas cujo *bootloader* não foi totalmente apagado e esse "resto" é visível na partição EFI ("de sistema")
- Existem 5 partições GPT
  - Partição EFI – tem os bootloaders do disco, formatada com FAT32 para compatibilidade entre todos os sistemas
  - Partição Windows formatadas com NTFS
  - Partição de "Recuperação do Windows" formatada com NTFS
  - Partição Linux (para o sistema) formatada com EXT4
  - Partição Linux (para dados) formatada com EXT4 (para Linux)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

### Exemplo Dual Boot Linux + Windows em UEFI



Partition	File System	Mount Point	Size	Used	Unused	Flags
/dev/sda1	fat32	/boot/efi	285.00 MiB	31.91 MiB	253.09 MiB	boot, esp
/dev/sda2	ntfs		115.88 GiB	69.73 GiB	46.15 GiB	msftdata
/dev/sda3	ntfs		549.00 MiB	460.21 MiB	88.79 MiB	hidden, diag
/dev/sda4	ext4	/	162.98 GiB	15.38 GiB	147.60 GiB	
/dev/sda5	ext4	/here	651.84 GiB	146.22 GiB	505.62 GiB	

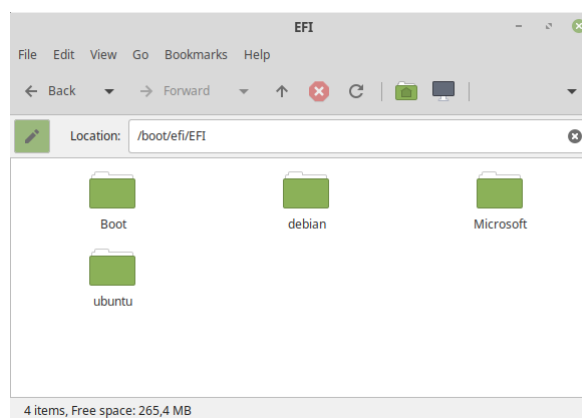
Partições existentes (vistas com o programa *gparted* em Linux)

DEIS/SEC

Sistemas Operativos – 2022/23

João Durães

### Exemplo Dual Boot Linux + Windows em UEFI



Conteúdo da partição EFI: contém os *bootloaders* dos vários sistemas presentes

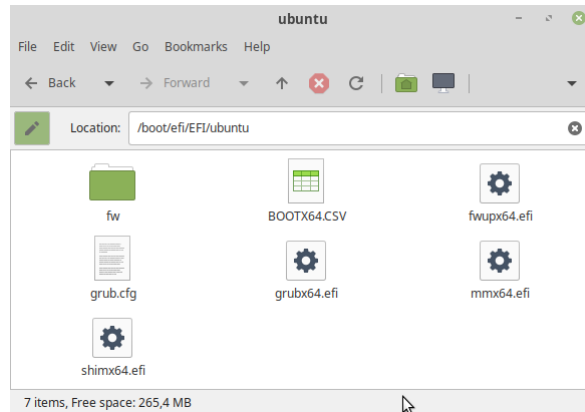
(O sistema debian já não está instalado mas permaneceu a diretoria referente ao seu bootloader)

DEIS/SEC

Sistemas Operativos – 2022/23

João Durães

### Exemplo Dual Boot Linux + Windows em UEFI



Ficheiros de arranque do Linux “Ubuntu Mint”

-> Reparar *onde* a partição EFI “de sistema” foi montada: /boot/efi

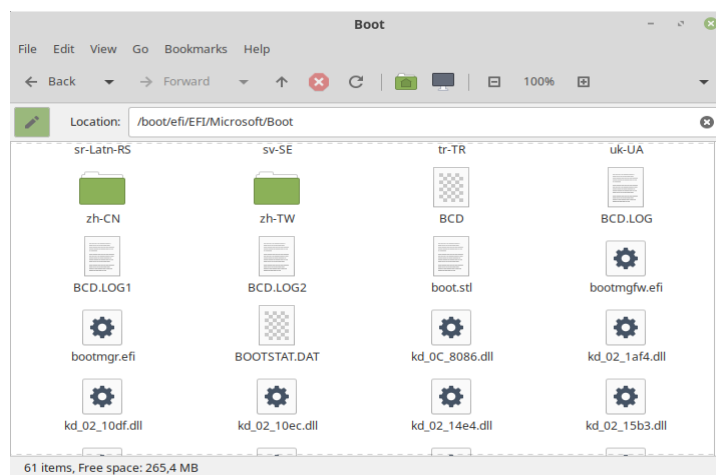
Tratam-se de ficheiros específicos a este sistema, mas com alguns ficheiros que a norma UEFI espera encontrar (ficheiro “.efi”)

DEIS/SEC

Sistemas Operativos – 2022/23

João Durães

### Exemplo Dual Boot Linux + Windows em UEFI



Ficheiros de arranque do Windows (observados a partir de um Linux)

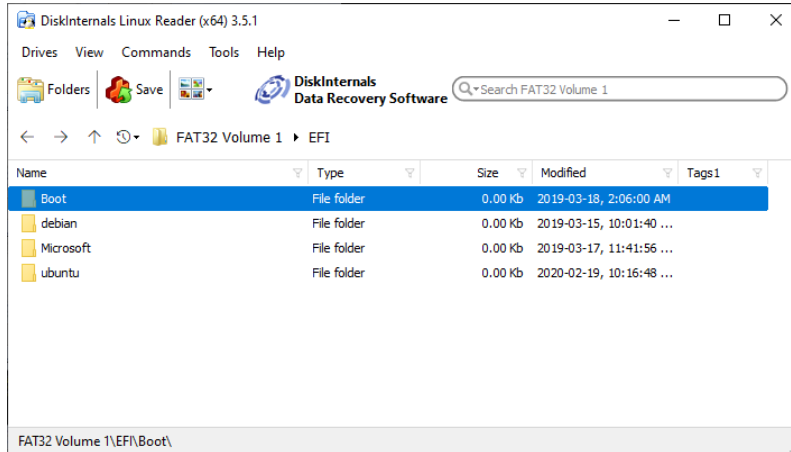
Ficheiros específicos a este sistema (“.dll”), mas com alguns ficheiros que cumprem a norma geral UEFI (notar os ficheiros com extensão “.efi”)

DEIS/SEC

Sistemas Operativos – 2022/23

João Durães

## Exemplo Dual Boot Linux + Windows em UEFI



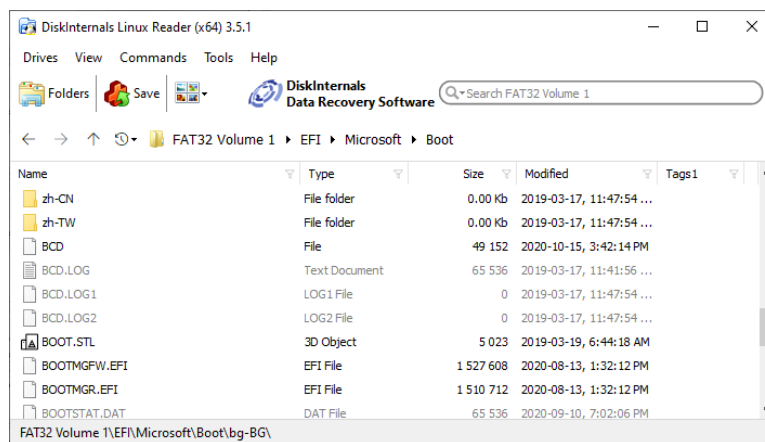
A mesma informação observada a partir de um sistema Windows com recurso a ferramentas *third party* ("DiskInternals Linux Reader")

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

## Exemplo Dual Boot Linux + Windows em UEFI



Ficheiros específicos da Microsoft relativos ao arranque do Windows  
( Partição UEFI de sistema, directoria \EFI\Microoft\Boot )

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

### Exemplo Interface UEFI no firmware (sem qualquer SO envolvido)

```
UEFI Interactive Shell v2.0. UEFI v2.31 (EDK II, 0x00010000) . Revision 1.02
Mapping table
  BLK0: Alias(s):
        PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
  BLK1: Alias(s):
        PciRoot(0x0)/Pci(0x1,0x1)/Ata(0x0)
Press ESC in 0 seconds to skip startup.nsh or any other key to continue.
2.0 Shell> _
```

Aspecto do ecrã de uma *shell* de arranque num sistema com UEFI.  
Note-se a existência de um *prompt* para a execução de comandos

Em alguns casos pode estar disponível um browser para a web

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

### Exemplo Interface UEFI no firmware (sem qualquer SO envolvido)

dh	- Displays the device handles in the UEFI environment.
disconnect	- Disconnects one or more drivers from the specified devices.
dmem	- Displays the contents of system or device memory.
dmpstore	- Manages all UEFI variables.
drivers	- Displays the UEFI driver list.
drvcfg	- Invokes the driver configuration.
drdiag	- Invokes the Driver Diagnostics Protocol.
echo	- Controls script file command echoing or displays a message.
edit	- Full screen editor for ASCII or UCS-2 files.
eficompress	- Compress a file using UEFI Compression Algorithm.
efidecompress	- Decompress a file using UEFI Decompression Algorithm.
else	- Identifies the code executed when 'if' is FALSE.
endfor	- Ends a 'for' loop.
endif	- Ends the block of a script controlled by an 'if' statement.
exit	- Exits the UEFI Shell or the current script.
for	- Starts a loop based on 'for' syntax.
getmtc	- Gets the MTC from BootServices and displays it.
goto	- Moves around the point of execution in a script.
help	- Displays the UEFI Shell command list or verbose command help.
hexedit	- Full screen hex editor for files, block devices, or memory.
if	- Executes commands in specified conditions.
ifconfig	- Modifies the default IP address of the UEFI IP4 Network Stack.
load	- Loads a UEFI driver into memory.
loadpcirom	- Loads a PCI Option ROM.

Exemplo 2 – lista parcial dos muitos comandos disponíveis (lista obtida com comando 'help') – Não há aqui nenhum SO envolvido

Notar funcionalidade complexa: comandos para *scripting* (*echo*, *for*, *if*)

DEIS/ISEC

Sistemas Operativos – 2022/23

João Durães

### Exemplo Interface UEFI no firmware (sem qualquer SO envolvido)

```
parse      - Retrieves a value from a record output in a standard format.
pause     - Pauses a script and waits for an operator to press a key.
pci       - Displays PCI device list or PCI function configuration space.
ping      - Pings the target host with an IPv4 stack.
reconnect - Reconnects drivers to the specific device.
reset     - Resets the system.
rm        - Deletes one or more files or directories.
sermode   - Sets serial port attributes.
set       - Displays or modifies UEFI Shell environment variables.
setsize   - Adjusts the size of a file.
setvar    - Changes the value of a UEFI variable.
shift     - Shifts in-script parameter positions.
smbiosview - Displays SMBIOS information.
stall     - Stalls the operation for a specified number of microseconds.
time      - Displays or sets the current time for the system.
timezone  - Displays or sets time zone information.
touch     - Updates the filename timestamp with the current system date and
time.
type      - Sends the contents of a file to the standard output device.
unload    - Unloads a driver image that was already loaded.
ver       - Displays UEFI Firmware version information.
vol       - Displays or changes information about a disk volume.

Help usage:help [cmd|pattern|special] [-usage] [-verbose] [-section name] [-b]
Shell> _
```

Exemplo 3 – lista de comandos disponíveis no *firmware* (sem qualquer sistema operativo a correr ou sequer existente)

Notar funcionalidade: **ping** (redes), **setsize** (trabalha com ficheiros)