Substituição do MBR pela tabela de partição GUID

Dança dos discos

Discos de alta capacidade usam a tabela de partição GUID ao invés do antigo master boot record, mas os aplicativos fdisk usuais do Linux não conseguem trabalhar com o novo esquema de particionamento. Veja como contornar isso usando as ferramentas certas.

por Hans-Peter Merkel

Esbarrar em um limite. Esse é um tema constante na história dos discos rígidos. Primeiro foi a BIOS, depois as controladoras [1] e até os sistemas operacionais. Número máximo de setores e cabeças, limites no particionamento e no número de partições, áreas de boot... a lista não tem fim.

Nos anos 1980, o MS DOS 3.2 forçou os usuários a dividir seus caros discos em partições de 32 MB. Naquela época, um disco rígido de 85 MB precisaria de pelo menos três partições (**figura 1**).

Dois discos modernos Seagate ST 33000651 AS, gentilmente cedidos à **Linux Magazine** pelo fornecedor para os testes deste artigo, contam com 3 TB cada, o que ultrapassa o atual limite da tabela de partição do tipo *master boot record* (MBR) para PCs (**figura 2**) [2].

O antigo esquema usado no particionamento MBR – CHS (cilindros, cabeças e setores; na sigla em inglês) – chegou ao fim da estrada há muitas primaveras, aos 8 GB (máximo de 1.024 cilindros x 255 cabeças x 63 setores). Desde então, a BIOS e o sistema operacional têm usado o padrão de 254 cabeças, 63 setores e 1.023 cilindros, usando 4 bytes na tabela de partição para definir posição e tamanho dos setores (endereçamento de bloco lógico, ou LBA, na sigla em inglês).

O mundo dos PCs vinha seguindo tranquilo com esse truque, mas os tempos estão mudando (ou já mudaram), pois um tamanho de (232-1 setores) X 512 bytes resulta em um máximo de 2,2 TB para uma partição e 4 TB para um disco [3] [4].

Em outras palavras, se você quiser usar todo o espaço de um disco de última geração com 3 TB, precisará realizar algumas mudanças grandes. Bem-vindo ao mundo da tabela de partição GUID!

A tabela de partição GUID (GPT, na sigla em inglês) é a sucessora da tabela MBR, sendo parte da EFI (*Extensible Firmware Interface*) [5]. A especificação EFI – a versão independente de fornecedores é conhecida como Unified EFI (UEFI) – substitui e padroniza as partes da BIOS responsáveis por inicializar e processar programas pré-boot. A GPT usa tabelas MBR do tipo LBA, mas com valores de 64-bits, permitindo assim até 8 ZB (1 zettabyte = 1 bilhão de terabytes) para um disco com setores de 512 bytes. Além disso, a GPT pode gerenciar até 128 partições (**figura 3**) [6].

Quadro 1: Inicialização a partir de discos GPT

Os servidores-padrão, mainframes e computadores Apple mais recentes cumprem a especificação EFI e inicializam a partir de um disco particionado com GPT. Diferentemente disso, as placas-mãe de desktops para CPUs Intel ou AMD atualmente inicializam quase que exclusivamente através das rotinas MBR da BIOS.

Felizmente, os usuários Linux contam com uma solução no GRUB 2 para resolver esse problema. O carregador de boot fornece um ambiente EFI para o computador, e o Linux pode assim inicializar a partir de um disco GPT sem problemas. Mas tenha cuidado: versões anteriores do GRUB não contam com esse recurso.

Mão na massa

O kernel Linux é compatível com a atual geração de discos de alta capacidade e GPTs a partir da versão 2.6.25, embora Linus Torvalds tenha depreciado a EFI como sendo uma "lesão cerebral da Intel" [7]. Nosso sistema de testes endereça o disco Seagate como `/dev/sdh`; a **listagem 1** mostra a saída do comando `dmesg`.

Se tentar particionar uma GPT com as populares ferramentas `fdisk` ou `cfdisk`, você não chegará muito longe, mas o aplicativo gráfico GParted trabalha com GPTs (**figura 4**). Aqui, usarei esse aplicativo para criar uma única grande partição no primeiro disco.

No GParted, você primeiro seleciona *Dispositivo* / *Criar Tabela de Partição*. Isso leva a uma janela com um grande aviso de que todos os dados serão perdidos. Na mesma janela, clique no triângulo ao lado de *Advançado*, e no menu drop-down ao lado de *Selecione o tipo da nova tabela de partição*, e escolha *GPT*.

A opção *Partição* / *Nova* abre caminho para a criação de uma partição primária com sistema de arquivos ext4. Depois de clicar em *Adicionar*, uma mensagem avisa que isso pode levar algum tempo, dependendo do número e do tamanho das operações.

Em nosso laboratório, isso foi uma pausa para o café de 10 minutos. No final, fiquei com 3 TB de espaço, menos um overhead de gerenciamento significativo, de 44,05 GB (**listagem 2**).

A **listagem 3** mostra o MBR 512-bytes, que a GPT usa e corretamente preenche por questões de compatibilidade. O `0x55AA` na **linha 5** indica o fim do MBR. O segundo setor da GPT contém os dados de cabeçalho para o particionamento GUID (**linhas 7 a 12**). As entradas de partição aparecem na posição `0x400`, contendo:

- Tipo de partição (16 bytes)
- Partição GUID (16 bytes)
- Início da partição (8 bytes)
- Fim da partição (8 bytes)
- Atributos (8 bytes)
- Nome da partição (72 bytes)

O GUID (*Globally Unique Identifier*) cuida dos nomes na GPT, sendo uma codificação 16-bytes, de tipo único, do tipo de partição. O sistema de testes agora tem uma partição de dados graças ao GParted. O GUID correspondente é `EBD0A0A2-B9E5-4433-87C0-68B6B72699C7`. A implementação na **linha 14** alterna alguns itens no ID (isso lembra um pouco a arquitetura do tipo *Big Endian*-*Little Endian*). Consequentemente, a EFI armazena uma cópia da GPT no final da mídia de dados como backup. Assim, você pode restaurar uma GPT primária danificada a partir dessa cópia. Se você precisar converter um disco de GPT para MBR, também precisará apagar a GPT secundária, senão o sistema vai continuar a identificar o disco como uma partição GPT.

Figura 1 Velho e novo lado a lado: um disco rígido grande no tamanho mas com apenas 85 MB, e um HD SATA 3 TB de última geração da Seagate.

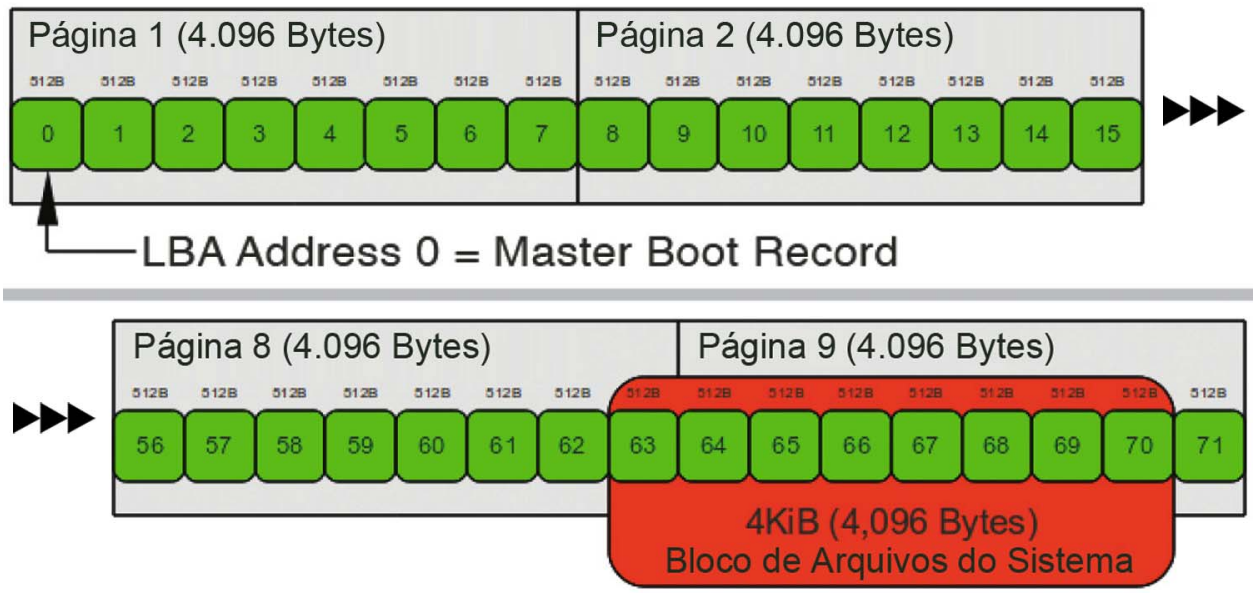

Figura 2 Estrutura do master boot record. A tabela de partição com quatro entradas tem apenas 4 bytes por número de setor.

Listagem 1: O kernel detecta discos de 3 TB

```
01 [782663.410960] ata6: SATA link up 6.0 Gbps
➥ (SStatus 133 SControl 300)
02 [782663.445923] ata6.00: ATA-8: ST33000651AS, CC44, max UDMA/133
03 [782663.445931] ata6.00: 5860533168 sectors,
➥ multi 0: LBA48 NCQ (depth 31/32)
04 [782663.446811] ata6.00: configured for UDMA/133
05 [782663.446830] ata6: EH complete
06 [782663.447076] scsi 7:0:0:0: Direct-AccessATAST33000651ASCC44
➥ PQ: 0 ANSI: 5
07 [782663.447507] sd 7:0:0:0: Attached scsi generic sg9 type 0
08 [782663.447585] sd 7:0:0:0: [sdh] 5860533168 512-byte logical
➥ blocks: (3.00 TB/2.72 TiB)
09 [782663.447780] sd 7:0:0:0: [sdh] Write Protect is off
10 [782663.447788] sd 7:0:0:0: [sdh] Mode Sense: 00 3a 00 00
11 [782663.447928] sd 7:0:0:0: [sdh] Write cache: enabled, read
➥ cache: enabled, doesn't support DPO or FUA
12 [782663.448846] sdh: unknown partition table
13 [782663.474623] sd 7:0:0:0: [sdh] Attached SCSI disk
```

Listagem 2: fdisk -lu depois do particionamento

```
01 Disk /dev/sdh: 3000 GB, 3000590369280 bytes
02 255 heads, 63 sectors/track, 364801 cylinders, total 5860528065 sectors
03 Units = sectors of 1 * 512 = 512 bytes
04
05    Device  Boot  Start   End   Blocks  Id  System
06 /dev/sdh1         2048  5860532223  2930272033  83  Linux
07 Warning: Partition 1 does not end on cylinder boundary.
```

Listagem 3: dd if=/dev/sdh | xxd (trecho)

```
01 [...]
02 00001c0: 0100 eefe ffff 0100 0000 ffff ffff 0000 ................
03 00001d0: 0000 0000 0000 0000 0000 0000 0000 0000 ................
04 00001e0: 0000 0000 0000 0000 0000 0000 0000 0000 ................
05 00001f0: 0000 0000 0000 0000 0000 0000 0000 55aa ..............U.
06 [...]
07 0000200: 4546 4920 5041 5254 0000 0100 5c00 0000 EFI PART....\...
08 0000210: 0e0f 49ee 0000 0000 0100 0000 0000 0000 ..I.............
09 0000220: afa3 505d 0100 0000 2200 0000 0000 0000 ..P]....".......
10 0000230: 8ea3 505d 0100 0000 cbb9 6e3f 0765 a746 ..P]......n?.e.F
11 0000240: a7c2 70a1 3db8 25c4 0200 0000 0000 0000 ..p.=.%.........
12 0000250: 8000 0000 8000 0000 6afb 3a17 0000 0000 ........j.:.....
13 [...]
14 0000400: a2a0 d0eb e5b9 3344 87c0 68b6 b726 99c7 ......3D..h..&..
15 0000410: 6903 b666 b91c 7646 a8dd e9d1 d7c6 bc5a i..f..vF.......Z
16 0000420: 0008 0000 0000 0000 ff9f 505d 0100 0000 ..........P]....
```

Listagem 4: Informações de partições do gdisk

```
01 Disk /dev/sdh: 5860533168 sectors, 2.7 TiB
02 Logical sector size: 512 bytes
03 Disk identifier (GUID): 0AE667A2-818F-4670-A328-500021C76A73
04 Partition table holds up to 128 entries
05 First usable sector is 34, last usable sector is 5860533134
06 Partitions will be aligned on 2048-sector boundaries
07 Use 'l' on the experts' menu to adjust alignment
08 Last sector (2048-5860533134, default =5860533134)
➥ or {+-}size{KMGTP}:
09 Current type is 'Linux/Windows data'
10 Hex code or GUID (L to show codes, Enter =0700):
```

Outras ferramentas

Os administradores de servidores que dependem da linha de comando não acharão o GParted muito útil. Atualmente há poucos substitutos do fdisk que aceitam GPT. O mais interessante é o GPT fdisk project [8], que depende dos programas `gdisk`, `sgdisk` e `fixparts`. A versão atual do Ubuntu tem apenas o `gdisk` em seu repositório, então faz mais sentido pegar todos os pacotes diretamente do openSUSE Build Service [9].

Depois de instalar esse conjunto de ferramentas, você pode verificar e modificar o disco. O comando

```
gdisk /dev/sdh
```

retorna o seguinte em relação a um novo disco:

```
Partition table scan:
    MBR: not present
    BSD: not present
    APM: not present
    GPT: not present
```

No menu de texto, a opção `p` retorna a tabela de partição (**listagem 4**), enquanto a opção `o` primeiro exibe um alerta de segurança, depois cria uma tabela GPT. A opção `n` permite que você crie uma nova partição do tamanho do disco, como fiz com o GParted antes. O Gdisk também dá acesso a algumas opções GUID que o GParted não tem.

Usei o valor-padrão de `0700` para dados Linux/Windows e teclei *w* ("*Write table to disk and exit*"). No nosso laboratório, usei o `mkfs.ext4` para formatar a partição. Na linha de comando, todo o processo levou três minutos menos do que com o GParted. A GPT criada ficou idêntica à do GParted, como revelou um teste com `dd` e `xxd`.

MBR para GPT

O Gdisk pode converter discos particionados com MBR para GPT, o que acaba com a necessidade de copiar e restaurar o conteúdo. A opção `r` leva ao menu de recuperação e conversão. Neste teste, usei um disco antigo de 200 GB com uma partição MBR.

Depois da conversão, o comando `gdisk /dev/sde` retornou o seguinte:

```
Partition table scan:
     [...]
Found valid GPT with protective
➥ MBR; using GPT.
```

A **listagem 5** mostra a saída da ferramenta forense `mmls` do pacote `sleuthkit` [10]. A **linha 2** confirma que a conversão foi bem sucedida. A entrada de partição da **listagem 6** é interessante: o tipo de dados está disponível tanto no GUID quanto em entrada de texto, entre as **linhas 4 e 6**.

GPT para MBR

O Gdisk também pode converter de GPT para MBR. Como, em alguns casos, isso não vai funcionar, é bom ser cuidadoso. Após reconverter o disco que tinha transformado em GPT, o `gdisk` retornou a saída da **listagem 7**. Neste ponto, o programa `sgdisk` do pacote GPT fdisk é útil para reparos de baixo nível.

Clusters

Os discos rígidos convencionais armazenam dados em setores de 512 bytes. Os sistemas operacionais e seus sistemas de arquivos agrupam múltipos setores para formar um cluster. Normalmente, serão agrupados oito setores, resultando em 4.096 bytes. Para um desempenho melhor, os discos de alta capacidade agora usam setores de cluster de 4.096 bytes internamente (os fornecedores geralmente se referem a eles como discos "4k"). Os discos Seagate usados em nossos testes eram desse tipo.

Para manter a compatibilidade externa com a BIOS, a lógica interna do disco converte os clusters de volta para setores de 512-bytes. Para evitar lentidão devido a essa adaptação de unidades de medida – entre o sistema operacional e o disco – os clusters de 4 KB do disco e do sistema de arquivos precisam coincidir perfeitamente, tanto para partições MBR quanto para GPT.

Se esse não for o caso, os resultados podem ser desastrosos: se você tiver uma porção de 2.048 bytes e o sistema operacional quiser gravar um cluster no disco, o disco precisa gravar dois clusters pela metade fisicamente, ocupando assim dois clusters inteiros. Os discos SSD são mais afetados por esse problema de alinhamento, pois os ciclos de gravação sobrepostos vão disparar um ciclo de leitura-modificação-gravação no drive flash.

Início da partição

Esse problema de desalinhamento de unidades de medida entre disco e sistema operacional é mais uma exceção do que a regra. A Microsoft é a responsável por essa situação, já que as ferramentas de particionamento desde o MS DOS até o Windows XP começam o particionamento imediatamente depois do MBR, no setor 63. Os discos particionados com esse método estarão sempre desalinhados. Se a primeira partição começar no setor 64, será possível obter grupos completos de 4.096 blocos. Assim, o Fdisk no Linux imita esse comportamento por questões de compatibilidade.

A Microsoft finalmente colocou um fim no problema com o Windows Vista e agora permite que a primeira partição comece no setor 2.048 em vez do 63. Dependendo da versão do fdisk que você usa no Linux, será

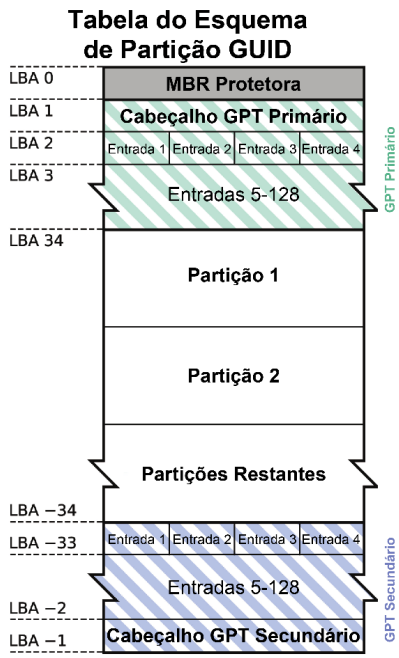

Figura 3 Estrutura GPT. Um cabeçalho de 100 bytes segue o padrão MBR para compatibilidade com sistemas operacionais anteriores. A tabela de partição usa 32 blocos para armazenar os dados de um máximo de 128 partições de 128 bytes.

preciso desabilitar a compatibilidade DOS através de uma opção de linha de comando. Felizmente, as ferramentas que podem criar uma GPT usam todas o padrão de 2.048 como setor inicial.

Esse comportamento é exatamente o que percebi durante os testes para este artigo: a **linha 6** da **listagem 2** mostra `2048` como setor inicial, o

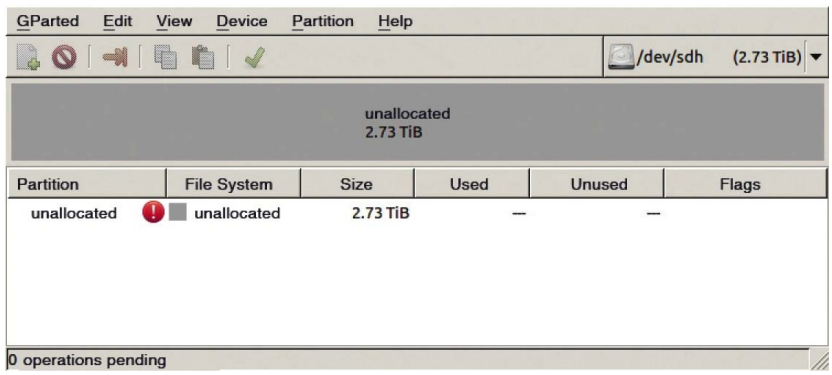

Figura 4 O GParted detecta um disco não particionado de 3 TB sem problemas.

que indica que o alinhamento está perfeito para o primeiro disco de 3 TB. Mas é preciso ter cuidado com discos que tenham sido usados antes. A **linha 10** da **listagem 5** revela que uma GPT também pode começar no setor 63 em vez de 2.048. Deve-se ter em mente que converti o segundo disco a partir de um esquema MBR para GPT; isso não alterou os limites da partição no setor 63.

Há muita discussão sobre se esse problema é tão ruim na prática quanto parece em teoria, já que os discos modernos usam técnicas de mapeamento secretas e específicas e também porque os sistemas operacionais e seus sistemas de arquivos usam diversos buffers e otimizações. Dito isso, definir setores de 4 KB na partição é uma coisa boa. Os administradores devem habilitar o recurso.

Conclusão

A era da GPT definitivamente começou. Os discos modernos precisam dessas novas tabelas para ir além dos 2,2 TB. Assim, os administradores devem manter distância do fdisk por ora, enquanto ferramentas como o GParted ajudam a tapar o buraco.

As placas-mãe de desktops não têm problema para inicializar o Linux a partir de um disco GPT, com o carregador de boot GRUB 2. Em termos de desempenho, é preciso manter os olhos abertos: a maioria dos discos de última geração usa clusters de 4 KB internamente em vez de setores de 512 bytes, sendo algo que você não vai notar com o esquema LBA; contudo, isso pode afetar a velocidade da partição no pior das hipóteses. ■

Gostou do artigo?

Queremos ouvir sua opinião. Fale conosco em cartas@linuxmagazine.com.br

Este artigo no nosso site: http://lnm.com.br/article/5992

Listagem 5: mmls /e3v/sde

```
01 GUID Partition Table (EFI)
02 Offset Sector: 0
03 Units are in 512-byte sectors
04
05     Slot    Start        End          Length       Description
06 00: Meta    0000000000   0000000000   0000000001   Safety Table
07 01: ---     0000000000   0000000062   0000000063   Unallocated
08 02: Meta    0000000001   0000000001   0000000001   GPT Header
09 03: Meta    0000000002   0000000033   0000000032   Partition Table
10 04: 00      0000000063   0390716864   0390716802   Linux/Windows data
11 05: ---     0390716865   0390721967   0000005103   Unallocated
```

Listagem 6: dd if=/dev/sde | xxd (trecho)

```
01 0000400: a2a0 d0eb e5b9 3344 87c0 68b6 b726 99c7   ......3D..h..&..
02 0000410: 42c4 6d88 c9f9 c84c a33d a0a6 ceb3 2bf3   B.m....L.=....+.
03 0000420: 3f00 0000 0000 0000 c0dd 4917 0000 0000   ?.........I.....
04 0000430: 0000 0000 0000 0000 4c00 6900 6e00 7500   ........L.i.n.u.
05 0000440: 7800 2f00 5700 6900 6e00 6400 6f00 7700   x./.W.i.n.d.o.w.
06 0000450: 7300 2000 6400 6100 7400 6100 0000 0000   s. .d.a.t.a.....
```

Listagem 7: gdisk /dev/sde

```
01 GPT fdisk (gdisk) version 0.7.1
02
03 Partition table scan:
04   MBR: MBR only
05   BSD: not present
06   APM: not present
07   GPT: not present
08
09 ***************************************************************
10 Found invalid GPT and valid MBR; converting MBR to GPT format.
11 THIS OPERATION IS POTENTIALLY DESTRUCTIVE! Exit by typing 'q' if
12 you don't want to convert your MBR partitions to GPT format!
13 ***************************************************************
14
15 Exact type match not found for type code A400; assigning type code for
16 'Linux/Windows data'
```

Mais informações:

[1] Interface ST506 de 1982: http://en.wikipedia.org/wiki/ST506

[2] MBR: http://en.wikipedia.org/wiki/Master_Boot_Record

[3] Tabela de partição MBR: http://en.wikipedia.org/wiki/Partition_table

[4] FAQ: Drive Partition Limits: http://www.uefi.org/learning_center/UEFI_MBR_Limits_v2.pdf

[5] EFI e UEFI: http://developer.intel.com/technology/efi/; http://www.uefi.org/home/

[6] GPT: http://en.wikipedia.org/wiki/GUID_Partition_Table

[7] Linus e o Extensible Firmware Interface: http://kerneltrap.org/node/6884

[8] GPT fdisk: http://www.rodsbooks.com/gdisk/

[9] GPT fdisk do openSUSE Build Service: https://build.opensuse.org/package/show?package=gptfdisk&project=home%3Asrs5694

[10] Sleuthkit: http://sleuthkit.org/