oom-killer and the linux memory

Relatore: Giuseppe Sucameli

oom-killer

OOM (Out-of-memor) killer is a process the Linux kernel runs when the system has low memory.

oom-killer reviews all running processes and **kills one or more of them** in order to free up system memory and keep the system running.

```
kernel: [35010811.456576] rasterisk invoked oom-killer: gfp_mask=0x2040d0, order=3,
oom_score_adj=0
…
kernel: [35010811.569082] Out of memory: Kill process 9160 (php-fpm) score 5 or
sacrifice child
kernel: [35010811.569122] Killed process 9160 (php-fpm) total-vm:492392kB, anon-
rss:257496kB, file-rss:68416kB
```

Almost all the times the oom-killer is invoked when the available memory is not enough.

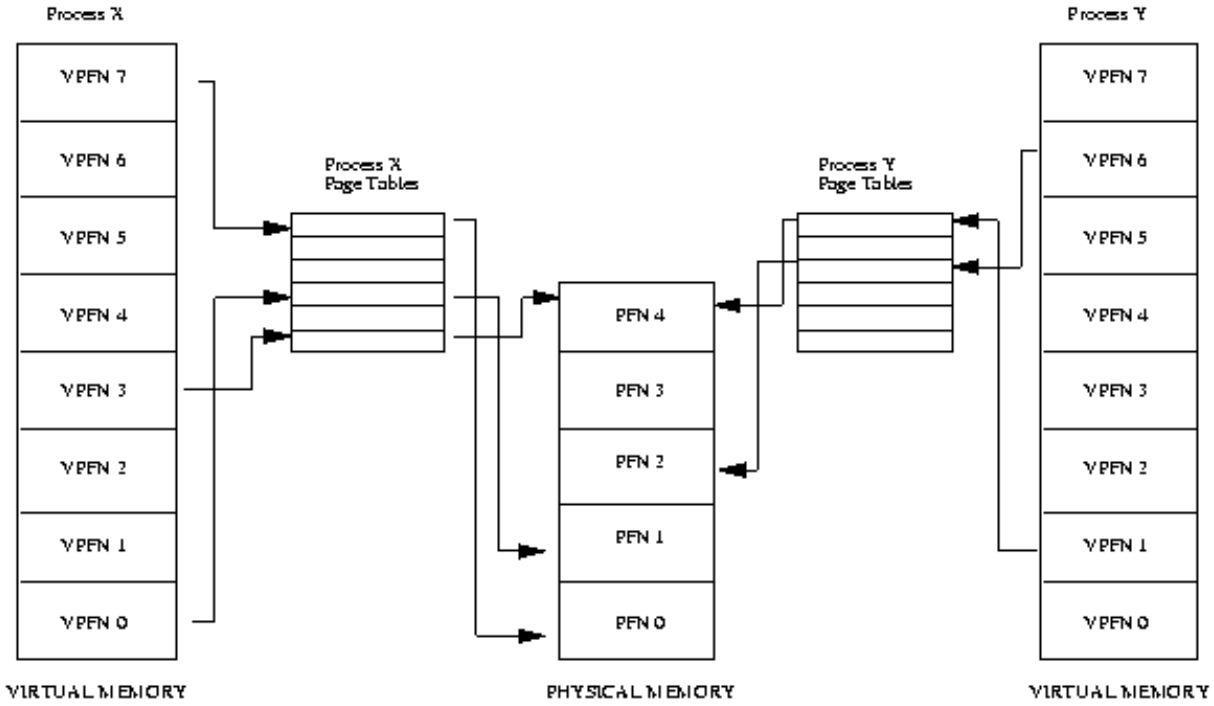
But it may eventually run even when there's lot of free memory.

Virtual memory

Virtual memory makes the system appear to have more memory than it actually has.

It provides:

- Large Address Spaces
- Protection
- Memory Mapping
- Fair Physical Memory Allocation
- Shared Virtual Memory

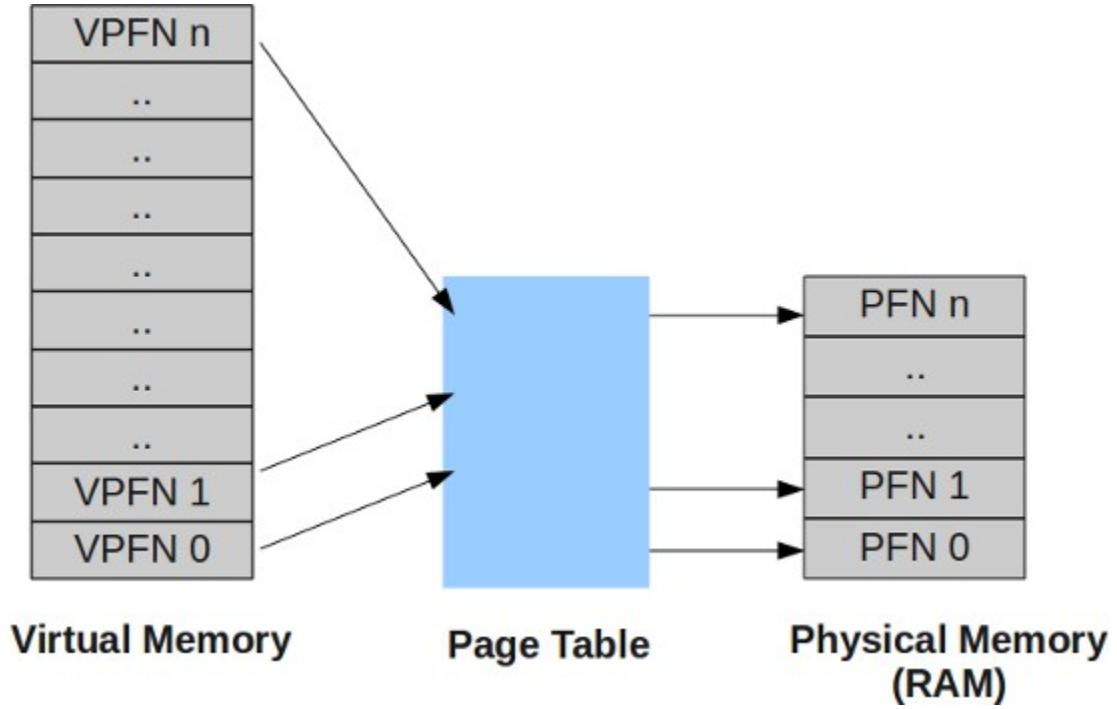


Virtual memory

The virtual memory is divided into fixed length chunks called **pages**.

A page is the basic unit of allocatable memory.

A tipical page size is **4KB**.



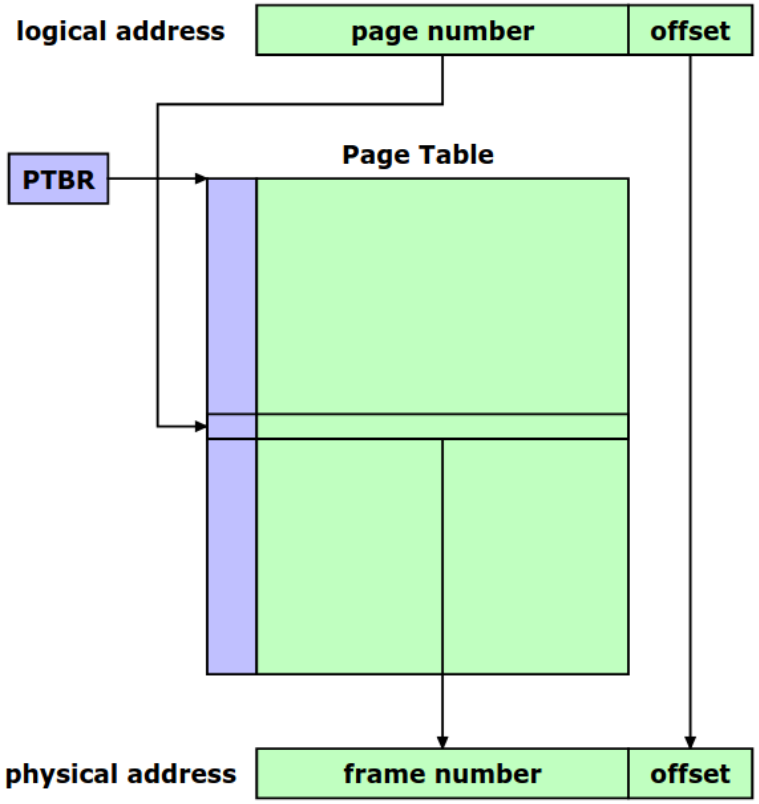
Translation between virtual and physical pages is done by using a **page table**.

Important to note that the page table always resides in physical memory.

Virtual memory address

A **virtual address** can be divided into two parts:

- an **offset**, the lowest N bits of the virtual address
- a virtual page frame number (**VPFN**), the rest of the address bits.



Physical memory

`free`

```
giuseppe@giuseppe-K55VD:~$ free -w
               total        used        free      shared     buffers       cache   available
Mem:         8050636     6233872      538748      160908       40488     1237528     1381856
Swap:        8271868     1921860     6350008
giuseppe@giuseppe-K55VD:~$ free -w -l
               total        used        free      shared     buffers       cache   available
Mem:         8050636     6233680      538856      160920       40496     1237604     1382036
Low:         8050636     7511780      538856
High:              0           0           0
Swap:        8271868     1921860     6350008
```

total: total installed memory

used: used memory (calculated as total - free - buffers - cache)

free: unused memory

shared: memory used (mostly) by tmpfs

buffers: memory used by kernel buffers

cache: memory used by the page cache and slabs

available: estimation of how much memory is available for starting new applications, without swapping.

Physical memory zones

Non-Uniform Memory Access (**NUMA**): memory may be arranged into banks having different cost to access depending on the “distance” from the processor.

Each bank is called **node**.

Each node is divided up into a number of blocks called **zones** which represent ranges within memory.

Each zone type suitable a different type of usage

32-bit Zones

- **ZONE_DMA** (<16MB):

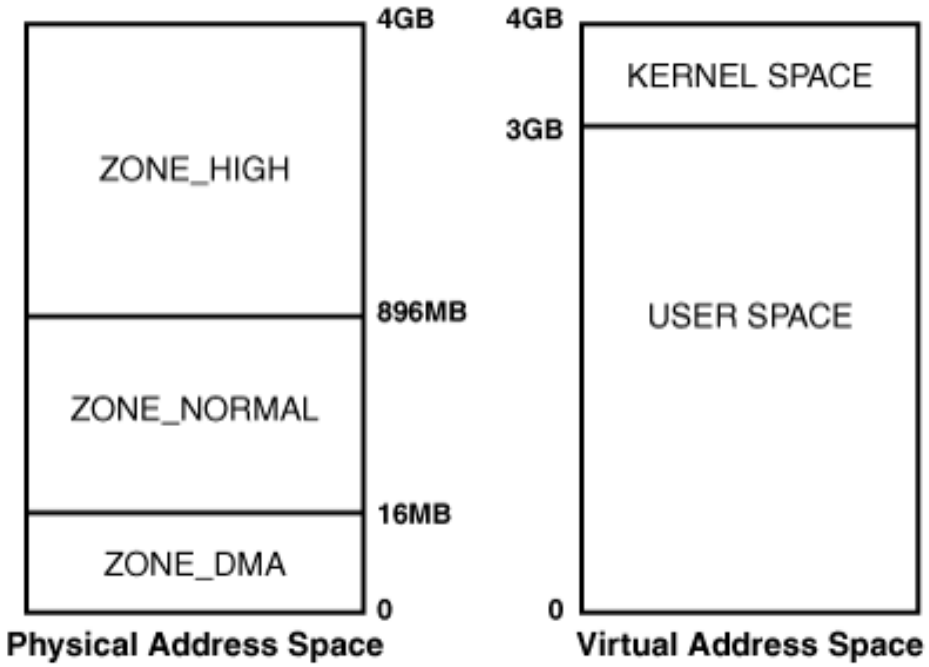
  the zone used for Direct Memory Access (DMA).

- **ZONE_NORMAL** (16MB to 896MB):

  also called low, memory normally addressable region

- **ZONE_HIGH** (>896MB):

  space that the kernel can access only after mapping resident pages to regions in ZONE_NORMAL



It is important to note that many kernel operations can only take place using ZONE_NORMAL

32-bit Zones

`cat /proc/pagetypeinfo`

```
Page block order: 10
Pages per block:  1024

Free pages count per migrate type at order       0      1      2      3      4      5      6      7      8      9     10
Node    0, zone      DMA, type    Unmovable     12     14      8      7      4      7      4      0      0      0      0
Node    0, zone      DMA, type  Reclaimable      1      3      3      1      0      1      0      0      0      0      0
Node    0, zone      DMA, type      Movable      4      2      1      1      0      5      1      0      0      0      0
Node    0, zone      DMA, type      Reserve      0      0      0      0      0      0      0      0      0      0      1
Node    0, zone   Normal, type    Unmovable    145    301    125     56     42     19      4      7      0      0      0
Node    0, zone   Normal, type  Reclaimable   1569    375    131     67     29     14      4      0      1      0      0
Node    0, zone   Normal, type      Movable    914    838    229     36      5      4      0      0      0      0      0
Node    0, zone   Normal, type      Reserve      0      0      0      0      0      0      0      0      0      0      1
Node    0, zone  HighMem, type    Unmovable      1      8     19     12      5      4      2      2      0      0      0
Node    0, zone  HighMem, type  Reclaimable      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone  HighMem, type      Movable   1626   2523   1298    799    270     46      6      4      0      0      0
Node    0, zone  HighMem, type      Reserve     18     13     11      6      8      4      1      1      1      0      0

Number of blocks type     Unmovable  Reclaimable      Movable      Reserve
Node 0, zone      DMA             1            0            2            1
Node 0, zone   Normal            18           20          179            1
Node 0, zone  HighMem             1            0          281            1
```

64-bit Zones

- **ZONE_DMA** (<16MB):

  the zone used for DMA, kept for historical reason

- **ZONE_DMA32** (16MB to 4GB):

  used for DMA, it exists because of the transition to 64Bit (some class of harware that can only do DMA to the low 4GB of memory).

- **ZONE_NORMAL** (>4GB):

  the remaining memory.

Note: a 2 GB machine running a 64-bit kernel will have no Normal memory at all while a 4 GB machine will have only a tiny amount of it.

64-bit Zones

`cat /proc/pagetypeinfo`

```
Page block order: 9
Pages per block:  512

Free pages count per migrate type at order       0      1      2      3      4      5      6      7      8      9     10
Node    0, zone      DMA, type    Unmovable      1      0      0      0      2      1      1      0      1      0      0
Node    0, zone      DMA, type      Movable      0      0      0      0      0      0      0      0      0      1      3
Node    0, zone      DMA, type  Reclaimable      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone      DMA, type   HighAtomic      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone      DMA, type          CMA      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone      DMA, type      Isolate      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone    DMA32, type    Unmovable    548    906    376    177     61     20      6      0      1      0      0
Node    0, zone    DMA32, type      Movable  26266  20339   2975    995    279     34      8      4      0      0      0
Node    0, zone    DMA32, type  Reclaimable    565    367    247    193     81     35     12      5      0      0      0
Node    0, zone    DMA32, type   HighAtomic      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone    DMA32, type          CMA      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone    DMA32, type      Isolate      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone   Normal, type    Unmovable    113     58    105    202     31      2      0      0      0      0      0
Node    0, zone   Normal, type      Movable    958   1212    617    316     44     12      0      0      0      0      0
Node    0, zone   Normal, type  Reclaimable    429    373      2      3      0      1      0      0      0      0      0
Node    0, zone   Normal, type   HighAtomic     17     15     14      6      1      0      0      0      0      0      0
Node    0, zone   Normal, type          CMA      0      0      0      0      0      0      0      0      0      0      0
Node    0, zone   Normal, type      Isolate      0      0      0      0      0      0      0      0      0      0      0

Number of blocks type     Unmovable      Movable  Reclaimable   HighAtomic          CMA      Isolate
Node 0, zone      DMA             1            7            0            0            0            0
Node 0, zone    DMA32            39         1594           23            0            0            0
Node 0, zone   Normal           208         2181           35            1            0            0
```

Memory allocation

Let’s consider the tipical linux 4KB page size.

A process can ask to allocate a block of memory consisting of one or more consecutive pages.

The block size is identified by its **order**:

order=0 means **2^0** consecutive pages = **1** page => **4KB**

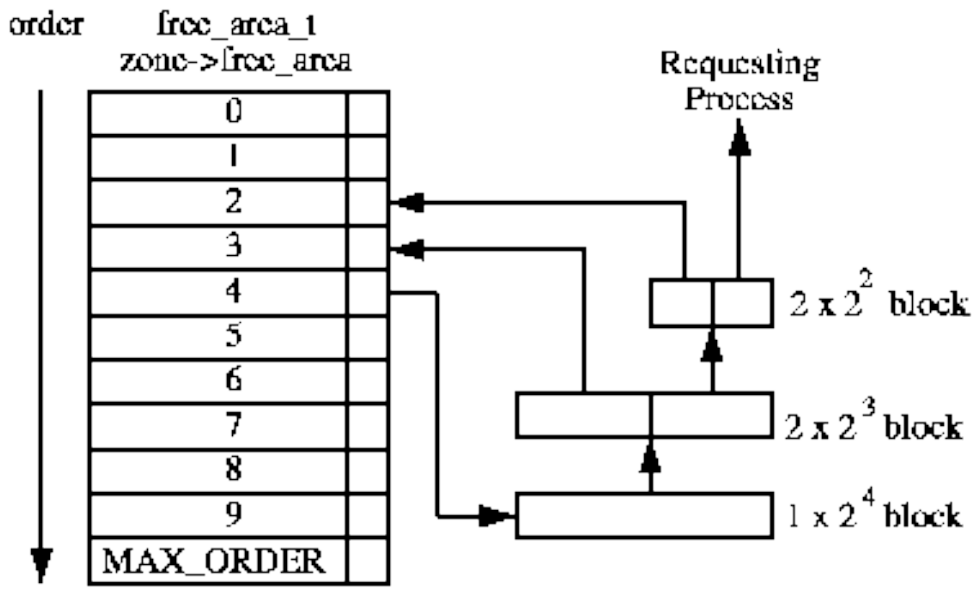
order=1 means **2^1** consecutive pages = **2** pages => **8KB**

...

order=3 means **2^3** consecutive pages = **8** pages => **32KB**

...

If a process asks for an order 3 block and there are no free blocks of that order, the allocator may split a higher-order block (e.g. order 4).



oom-killer

Almost all the times the oom-killer is invoked when the available memory is not enough.

But it may eventually run even when there's lot of free memory.

```
kernel: [35010811.456576] rasterisk invoked oom-killer: gfp_mask=0x2040d0, order=3,
oom_score_adj=0
…
kernel: [35010811.466169] Node 0 DMA free:1904kB min:100kB low:124kB high:148kB ...
kernel: [35010811.466348] Node 0 Normal free:126120kB min:2348kB low:2932kB high:3520kB …
kernel: [35010811.466540] Node 0 HighMem free:57282368kB min:512kB low:104364kB
high:208220kB ...
```

In the previous figure the oom-killer has been invoked for a **order=3** block allocation which means the process has requested 32KB of consecutive memory.

As you can see, there is enough memory in Normal zone, so it may be probably due to **memory fragmentation**, i.e. there are no free blocks for the requested order.

Links

https://www.kernel.org/doc/gorman/html/understand/understand005.html

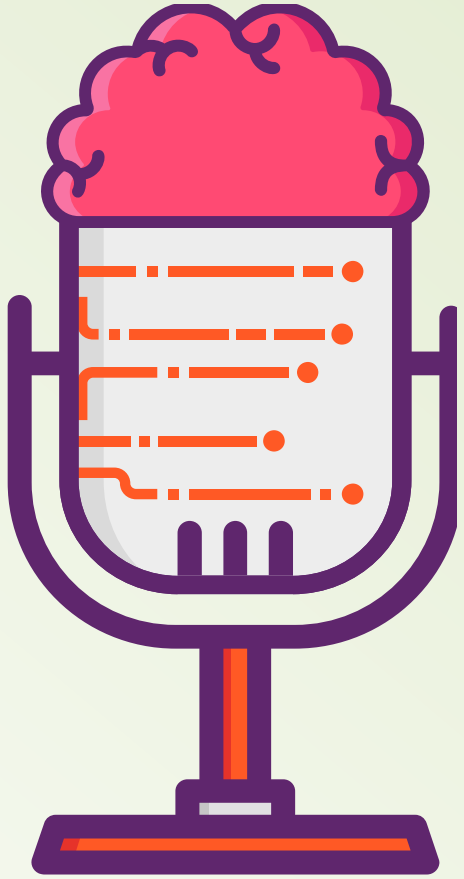
https://tldp.org/LDP/tlk/mm/memory.html

https://www.thegeekstuff.com/2012/02/linux-memory-management/

https://utcc.utoronto.ca/~cks/space/blog/linux/KernelMemoryZones

https://www.kernel.org/doc/gorman/html/understand/understand016.html

https://utcc.utoronto.ca/~cks/space/blog/linux/DecodingPageAllocFailures



oom-killer and linux memory

That’s all folks!

Questions?