목차

THP(Transparent Huge Pages) 기능과 설정 방법

원문링크 : http://www.cubrid.com/zbxe/bbs_developer_faq/3550332

THP(Transparent Huge Pages) 란?

리눅스 THP 기능에 관련하여 공식 문서들에는 기능에 대한 정의만 설명되어 있고 정확히 어떤 방식으로 어떤 기능을 하는 것인지는 설명되어 있지 않아 자세히 설명드리겠습니다.

페이지는 메모리를 일정한 크기로 나누어 관리하는 단위입니다.

CPU는 메모리에 데이터를 요청할때 Virtual address를 통해 요청하고 MMU가 그것을 Physical address로 변환시켜 실제 메모리에 엑세스 하게 됩니다. 여기서 `Virtual address`와 `Physical address`를 연결해주는 테이블이 `Page table`이고 이 테이블이 존재하는 위치를 담은 것이 TTB (Translation Table Base Address) 입니다.

CPU가 매번 TTB를 통해 물리 메모리에 접근해서 데이터를 가져오게 되면 속도가 느리므로 Virtual address 로부터 Physical address로의 매핑 정보를 가진 Entry를 가진 캐시가 있다면 메모리를 거치지 않고도 바로 Physical address를 얻을 수 있게 되어 속도가 빨라질 것입니다. 이 캐시 역할을 하는 것이 TLB (Translation Lookaside Buffer) 입니다.

CPU가 Virtual Address로 메모리를 엑세스 하려 하면 캐시 역할을 하는 TLB 내 Virtual Address Entry들에 매핑된 정보가 있는지 확인하고 만약 있다면(hit) 곧바로 메모리에 엑세스해서 데이터를 가져오지만 없다면(miss) 메모리의 TTB(Translation Table Base Address)에 1차적으로 접근하여 Physical address를 구해온 뒤 구해온 Physical address를 이용해 메모리에 또 한번 접근하여 데이터를 가져옵니다.

여기서 hit ratio가 떨어진다고 하면 miss 되어 메모리에 2번 접근하는 횟수가 늘어나므로 성능 저하가 일어납니다. (즉, hit ratio가 낮다면 TLB 존재의 의미가 없어지게 됩니다.)

따라서 이 hit ratio가 높아지려면 TLB 내의 Entry에 들어갈 내용이 작아야 하는데, 대용량 메모리를 사용할수록 page와 page table 개수가 많아지므로 TLB 내의 Entry가 커지게 됩니다. Entry를 작게 하려면 page 자체의 크기를 늘려서 개수를 줄이는 방법을 써야 하는데 그것이 THP입니다.

THP는 기존 4KB 크기의 페이지 대신 2MB 혹은 1GB 크기의 페이지를 동적으로 할당하여 사용하는 기능이며 RHEL 6 이후 기본으로 활성화 되어있습니다.

하지만 원래 THP 기능의 의도와는 달리 오히려 시스템 성능이 저하되는 경우가 많아 이 옵션을 disable 시키는 것을 권장합니다.

THP 옵션이 활성화 되어있는지 확인하는 방법

- 설정 여부 확인

```
[root@host]# cat /sys/kernel/mm/transparent_hugepage/enabled
```

위 명령 실행 후

```
[always] madvise never  -> 출력된 결과에 [always] 에 대괄호가 되어있으면 THP가 활성화 된 상태입니다.
always madvise [never]  -> 출력된 결과에 [never] 에 대괄호가 되어있으면 THP가 비활성화 된 상태입니다.
```

- 메모리 확인

```
[root@host]# cat /proc/meminfo
```

위의 명령을 실행하면 아래와 같은 결과가 나옵니다.

```
MemTotal:        1003184 kB
MemFree:          604844 kB
MemAvailable:     750616 kB
Buffers:             884 kB
Cached:           143904 kB
SwapCached:            0 kB
Active:           160624 kB
Inactive:         112368 kB
Active(anon):     128912 kB
Inactive(anon):     6356 kB
Active(file):      31712 kB
Inactive(file):   106012 kB
Unevictable:           0 kB
Mlocked:               0 kB
SwapTotal:       2097148 kB
SwapFree:        2097148 kB
Dirty:                92 kB
Writeback:             0 kB
AnonPages:        128300 kB
Mapped:            37492 kB
Shmem:              7064 kB
Slab:              60876 kB
SReclaimable:      23024 kB
SUnreclaim:        37852 kB
KernelStack:        8480 kB
PageTables:         6220 kB
NFS_Unstable:          0 kB
Bounce:                0 kB
WritebackTmp:          0 kB
CommitLimit:     2598740 kB
Committed_AS:     494544 kB
VmallocTotal:   34359738367 kB
VmallocUsed:      185924 kB
VmallocChunk:   34359535100 kB
HardwareCorrupted:     0 kB
```

```
AnonHugePages:         0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
Hugepagesize:       2048 kB
DirectMap4k:       61312 kB
DirectMap2M:      987136 kB
DirectMap1G:           0 kB
```

여기서 아래의 모든 값이 0 이어야 THP가 완전히 비활성화 된 상태입니다.

```
AnonHugePages:         0 kB
HugePages_Total:       0
HugePages_Free:        0
HugePages_Rsvd:        0
HugePages_Surp:        0
```

애초에 Linux 부팅 시점 부터 관련된 설정들을 모두 막아버리지 않으면 THP 기능 활성화 상태는 [never]로 나올지라도 AnonHugePages 는 여전히 할당되어 있는 문제가 생겨버립니다. 따라서 AnonHugePages가 완전히 0이 되도록 설정되어야 하지만 불완전한 방법으로 THP를 disable 시키는 경우가 많아 실제 유효한 방법을 기술하겠습니다.

THP 옵션 비활성화 (in RHEL7)

RHEL 7 버전에서의 THP 비활성화 방법을 기술하겠습니다.

THP는 커널 설정과 tuned 설정에 의해 활성화되므로 이 2가지를 모두 비활성화 시킬 필요가 있습니다.

Step 1: Initially, we should create a customized version of the currently running profile. The customized version will disable THP. Find out which profile is active, create a copy. In the following example we currently use the throughput-performance profile:

새로운 tuned profile을 만든 뒤 active 상태로 설정

아래와 같이 명령어를 입력하여 현재 active 상태인 tuned profile을 확인합니다.

```
[root@host]# tuned-adm active
```

위 명령어 입력 후 만약 현재 OS가 가상머신을 통해 운영중이라면 virtual-guest, 실제 서버의 콘솔에서는 throughput-performance로 나오게 됩니다.

```
Current active profile: throughput-performance   // in Server default
configuration
```

```
Current active profile: virtual-guest   // in Virtual OS
```

새로 생성하길 원하는 프로필 명으로 /etc/tuned 아래 경로에 디렉토리를 생성합니다. (여기서는 프로필 명이 cubrid라고 가정)

```
[root@host]# mkdir /etc/tuned/cubrid
```

새로 생성한 디렉토리 안에 tuned.conf 파일을 다음과 같이 생성 후 저장합니다.

```
[root@host]# vi /etc/tuned/cubrid/tuned.conf
```

```
[main]
include= throughput-performance
[vm]
transparent_hugepages=never
```

위에 생성한 파일에 실행 권한을 부여합니다.

```
[root@host]# chmod +x /etc/tuned/cubrid/tuned.conf
```

새로 만든 프로필을 활성화합니다.

```
[root@host]# tuned-adm profile cubrid
```

커널 파라미터 추가

/etc/sysconfig/grub 파일에 다음과 같이 transparent_hugepage=never 커널 파라미터를 추가합니다.

```
GRUB_TIMEOUT=5
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL_OUTPUT="console"
GRUB_CMDLINE_LINUX="rd.lvm.lv=centos/root rd.lvm.lv=centos/swap
crashkernel=auto rhgb quiet transparent_hugepage=never"
GRUB_DISABLE_RECOVERY="true"

/* GRUB_CMDLINE_LINUX 의 맨 뒤에 transparent_hugepage=never 추가 */
```

grub2-mkconfig를 통해 grub.cfg 파일을 재생성합니다.

```
[root@host]# grub2-mkconfig -o /boot/grub2/grub.cfg
```

재부팅 후 아래 명령어를 통해 설정 내용을 확인합니다. 출력 결과에 위에서 추가한 사항이 포함되어 있으면 성공입니다.

```
[root@host]# reboot
[root@host]# cat /proc/cmdline
BOOT_IMAGE=/vmlinuz-3.10.0-229.20.1.el7.x86_64 root=/dev/mapper/centos-root
ro rd.lvm.lv=centos/root rd.lvm.lv=centos/swap crashkernel=auto rhgb quiet
transparent_hugepage=never
```

참조링크

- https://access.redhat.com/solutions/1578873
- http://lunatine.net/thp-and-page-allocation-error/
- http://bloodguy.tistory.com/entry/Linux-%EC%8B%9C%EC%8A%A4%ED%85%9C%ED%94%84%EB%A1%9C%EC%84%B8%EC%8A%A4-%EB%A9%94%EB%AA%A8%EB%A6%AC-%EC%82%AC%EC%9A%A9%EB%9F%89-%ED%99%95%EC%9D%B8-check-systemprocess-memory-usage