목차

Linux 메모리 효율을 위한 vfs_cache_pressure 3
유휴 메모리가 전부 어디로 간거지? (Page Cache) 3
메모리를 전부 사용하고 있는 시스템 3
Slab 3
사라진 메모리를 찾아보자 4
vfs_cache_pressure 4
메모리를 되찾자 5
추가로 알아두면 좋은 팁 6
출처 6

Linux 메모리 효율을 위한 vfs_cache_pressure

유휴 메모리가 전부 어디로 간거지? (Page Cache)

Linux는 I/O 성능을 높이기 위해서 Page Cache를 사용한다. 이 글에서는 Page Cache에 대해서는 다루지 않지만 간단히 설명하면 다음과 같다. Linux는 물리적인 저장/통신 장치와 데이터를 주고 받을 때 메모리에 먼저 적재한 후에 데이터를 주고 받는데 이는 동일한 데이터에 대한 접근을 할 경우 메모리에서 바로 가져오도록 하여 I/O 성능을 높이기 위함이다. 이를 Page라는 단위로 관리를 하며 흔히 Page Cache라고 이야기 한다.

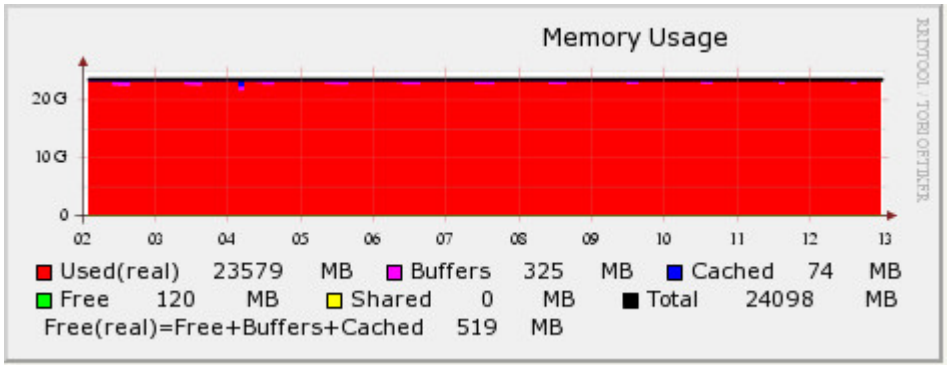
따라서, 한번이라도 데이터를 읽거나 쓴 적이 있다면 메모리는 Page Cache에 적재되고 아래의 파일에서 Cached 영역으로 표기 된다.

```
/proc/meminfo
```

Linux 커뮤니티에서 흔히 받는 질문 중 하나인 ‘ 왜? 제Linux의 Free 메모리가 이것밖에 안되나요?'의 원인이 Page Cache 매커니즘이다.

메모리를 전부 사용하고 있는 시스템

아래 그림은 특정 시스템의 메모리 사용현황을 RRDTool을 이용해서 그래프로 도식화 한 것이다. 24GB 메모리의 대부분을 사용중에 있는 것으로 나타나는데 실제 해당 서버의 프로세스가 사용하고 있는 메모리 크기를 확인해 보면 15GB 정도를 사용하고 있는 시스템이었다.



그렇다면, 남은 메모리는 어디로 간 것일까? 그래프에서 Page Cache로 잡힌 부분은 Cached (파란색)으로 표기가 된다. 따라서, 남은 메모리가 Page Cache에 의해서 사용 된 것이 아니라는 걸 알 수 있다.

그 많은 메모리는 대체 어디로 사라진 걸까?

Slab

Slab Allocator라는 것이 있다. 이는 일종의 자원 할당자 중 하나로 4KB의 크기를 가진 Page로 데이터를 저장하고 관리할 경우 발생하는 단편화를 최소화 하기 위해 고안 된 물건이다. Linux의 커널은 자료구조로 Slab을 사용하고 있으며 `/proc/meminfo`에서 아래 항목은 Linux 커널이 사용하는 캐시 크기를 의미

한다.

```
Slab:           349364 kB
```

Linux 커널에서 커널과 디바이스 드라이버, 파일시스템 등은 영구적이지 않은 데이터들을 저장하기 위한 공간이 필요한데(inode, task 구조체, 장치 구조체 등) 이것이 Slab 구조하에 관리 되고 있다. 따라서, 앞서 언급한 meminfo 파일의 Slab 항목은 이러한 데이터들의 메모리상 크기를 의미한다. 그래서 커널 캐시라고도 표현한다.

이러한 캐시 데이터 중에서 본 글의 목적과 관련성이 높은 것은 inode와 dentry에 대한 캐시이다. inode와 dentry는 파일 자료구조를 의미한다. VFS(Virtual File System)와 관련된 부분을 공부하다보면 자주 만나게 되는 dentry는 경로명 탐색을 위한 캐시 역할도 수행한다고 알려져 있다.

간단히 얘기해서 어떠한 파일을 생성할 때 파일의 정보를 담고 있는 inode와 dentry는 보다 빠른 데이터 접근을 위해서 커널의 Slab 자료구조에 추가된다고 이해하면 된다.

사라진 메모리를 찾아보자

앞서 살펴본 시스템에서 어플리케이션(프로세스)이 사용하는 메모리를 뺀 나머지 메모리는 `/proc/meminfo` 파일의 Slab 항목에서 찾을 수 있었다. Slab의 크기가 9GB 정도 되는 것으로 확인 되었다. 즉, 커널이 Slab 자료구조에 계속해서 캐시데이터로 담고 있었던 것이다.

왜 이런 일이 벌어졌을까?

이는 동작하고 있는 프로세스의 성격과 관련이 높다. 해당 프로세스가 주로 하는 작업의 패턴을 확인 해 본 결과 특정 파일들을 다량으로 생성하고 이 데이터를 가공처리하는 작업을 반복하고 있었다. 다만, 이러한 작업 과정에서 생성되고 삭제되는 파일이 매우 많은 것으로 확인 되었다.

앞서 이야기 한대로 Linux 커널은 파일시스템의 성능을 나아가 시스템의 성능을 개선하기 위해 inode와 dentry를 메모리에 캐시한다. 하지만, 파일을 빈번하게 생성/삭제 하거나 다량의 파일을 다루는 시스템의 경우 해당 파일을 자주 재활용하지 않는다면 (즉, 생성/기록 후에 데이터를 지속해서 접근하여 읽지 않는 경우) 캐시에 메모리를 사용하기 보다는 I/O를 위한 버퍼 또는 프로세스에 할당되어 활용 되는 편이 좋다.

Linux Slab 자료구조의 상태에 대해서 자세히 살펴 볼 수 있는 slabtop이란 명령이 존재한다. 이 명령을 실행 해 보면 inode 캐시(ext3혹은 ext4_inode_cache라는 이름)와 dentry_cache의 현재 크기를 알 수 있다.

Slab에 대해서 조금 더 알고 싶으면 이 글을 추천한다.

vfs_cache_pressure

Linux 커널의 vm 구조와 관련된 파라미터로 `vfs_cache_pressure`라는 것이 존재한다. 이 파라미터는 디렉토리와 inode 오브젝트에 대한 캐시로 사용된 메모리를 반환(reclaim)하는 경향의 정도를 지정하는 항목이다. 기본 값은 100.

이 값을 0으로 설정하게 되면 Linux 커널은 오브젝트에 대한 캐시를 반환하려고 하지 않을 것이며 얼마 지나지 않아 시스템은 Out of Memory 상태를 호소할 것이다. (커널이 메모리를 다 먹어버렸다고!!)

그리고, 100 이상의 값을 주면 Linux 커널은 오브젝트에 대한 캐시를 가급적 반환하려고 하며 (다른 말로 가급적 캐시해서 보관하려고 하지 않으려 든다) 이를 이용하면 inode와 dentry 캐시를 줄일 수가 있다.

메모리를 되찾자

`vfs_cache_pressure`를 이용해서 Linux 커널에게 캐시 데이터를 반환하도록 요구해 보자. 100이상의 값을 설정하면 되는데 경험상 10000정도로 설정하면 큰 문제 없이 (즉, 커널이 반환한 캐시 데이터로 인해 성능이 저하되는 문제 등) 운영할 수 있다. 이 값에 대해서는 각자 사용하는 시스템에 값을 바꾸어 보면서 확인하는게 가장 좋다.

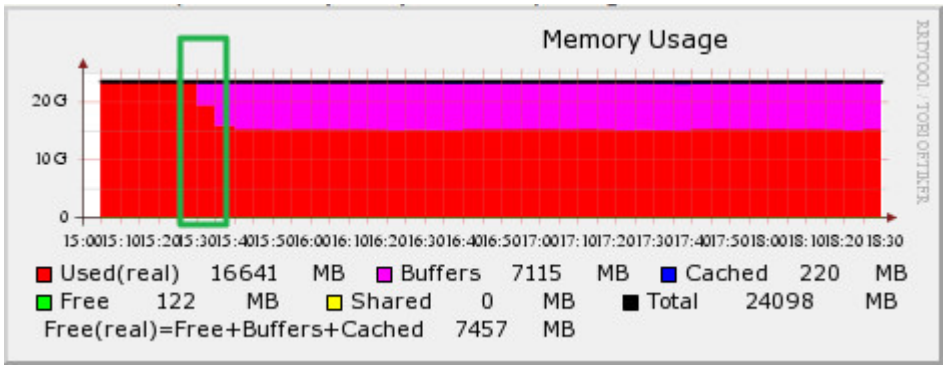
해당 값의 변경은 아래와 같이 할 수 있다. (이 글에 관심이 있는 분이라면 당연히 알만한 내용이지만)

```
echo 10000 > /proc/sys/vm/vfs_cache_pressure
또는
$ sysctl vm.vfs_cache_pressure=10000
```

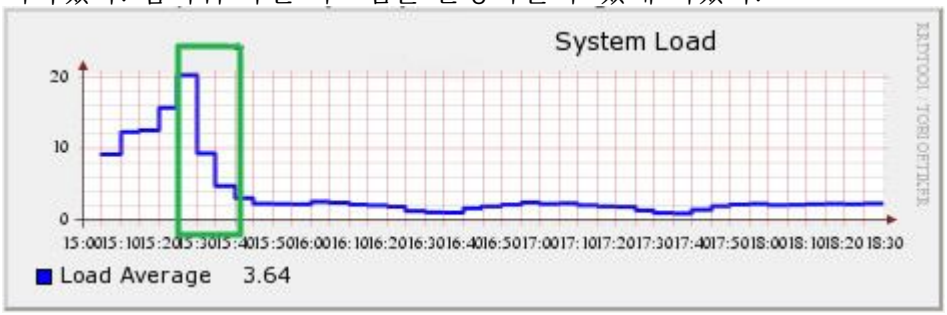
영구적으로 설정하기 위해서는 `/etc/sysctl.conf` 파일에 아래와 같이 추가

```
vm.vfs_cache_pressure = 10000
```

앞서 살펴본 시스템에서 값을 변경하고 나니 아래와 같이 바뀌었다. 9GB 정도의 Real 영역이 해제되고 이 영역이 Buffer로 활용되기 시작하였다.



즉, 커널이 잡고 있던 캐시를 해제하여 Buffer로 활용되고 나니 시스템의 부하 값 (Load 값)이 낮아지기 시작했다. 힘겨워 하던 시스템을 안정시킬 수 있게 되었다.



Load 값 : 현재 시스템에서 실행 중인 프로세스와 non-interruptible 상태에 있는 프로세스의 숫자에 대한 평균 값으로 시스템의 부하 정도를 가늠하는 지표로 사용된다. 좀 더 재밌는 설명을 원한다면 이 글을 추천한다.

재미없는 이야기로 기술되었지만 요약하자면 기본적으로 Linux 커널은 유휴 메모리가 있다면 캐시하려고 들기 때문에 어떤 경우에는 이렇게 사용되는 메모리의 양을 조절하면 더 좋은 효과를 얻을 수 있다는 이야기 이다. (이 말이 더 어렵다)

추가로 알아두면 좋은 팁

앞서 `vfs_cache_pressure`를 통해서 Linux 커널의 VFS 관련 오브젝트 캐시경향을 조절하는 방법외에도 이러한 정책을 세워서 효과를 보기보다 지금 당장 캐시를 비우기만 하고 싶을 때는 아래와 같은 방법이 있다.

```
echo 1 > /proc/sys/vm/drop_caches
echo 2 > /proc/sys/vm/drop_caches
echo 3 > /proc/sys/vm/drop_caches
```

각각의 숫자 값은 아래와 같은 의미를 가지고 있으며 해당 값이 설정되면 영구적으로 지속되는 것이 아니라 값이 설정되는 순간만 그 값에 따라서 반영 될 뿐이다.

> 주의 : 아래 명령을 수행하기 전에 반드시 sync 등을 통해서 캐시에 휘발성으로 담긴 데이터를 실제 저장 장치에 반영시키도록 해야 한다.

- drop_caches = 1

Page cache를 해제 한다.

- drop_caches = 2

inode, dentry cache를 해제 한다.

- drop_caches = 3

Page cache, inode cache, dentry cache를 모두 해제 한다.

만약, 3번으로 설정하면 시스템이 잠시 멈추는 증상을 경험 할 수도 있다. (모든 캐시를 비우기 위해서 혼신의 힘을 다할테니깐)

출처

- http://tumblr.lunatine.net/post/28546340998/faq-linux-vfs-cache-pressure

From:
https://atl.kr/dokuwiki/ - **AllThatLinux!**

Permanent link:
https://atl.kr/dokuwiki/doku.php/linux_%EB%A9%94%EB%AA%A8%EB%A6%AC_%ED%9A%A8%EC%9C%A8%EC%9D%84_%EC%9C%84%ED%95%9C_vfs_cache_pressure

Last update: **2018/08/07 09:03**