ptmalloc2

> 메모리의 효율적인 관리

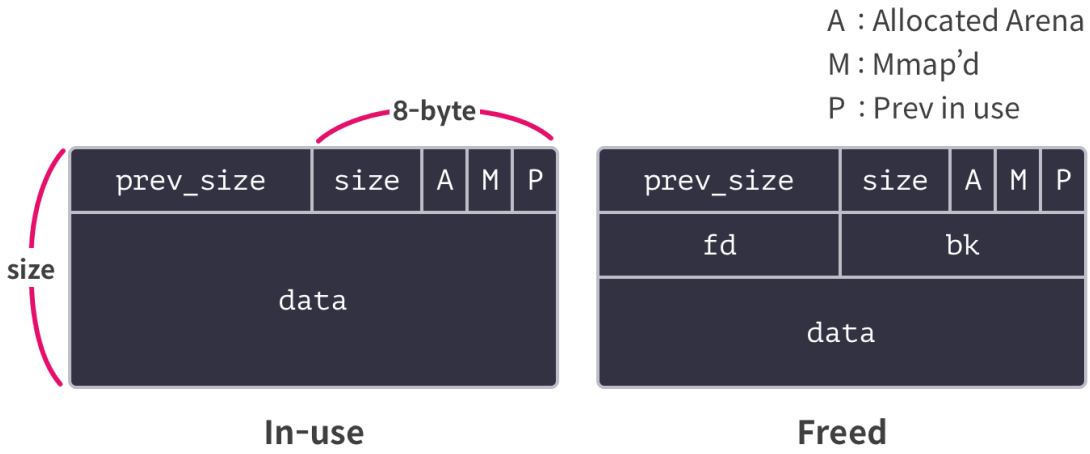
ptmalloc2는 동적 메모리를 관리하는 즉, Memory Allocator를 구현한 리눅스의 알고리즘이다.

ptmalloc2의 객체로 Chunk, bin, tcache, arena가 있다.

▼ Chunk

Chunk(청크)는 ptmalloc2가 **할당한 메모리**의 **공간**이다.

할당이 된 메모리인지 해체된 메모리인지에 따라 in-use와 freed 두 가지로 나뉜다.



출처: dreamhack_Background:ptmalloc2

청크 헤더

- data영역 외의 부분으로 청크 관리에 필요한 정보들을 담음
- in-use에서는 8바이트, freed에서는 16바이트

prev_size

- 8바이트
- 인접한 직전 청크의 크기를 저장
- 청크의 병합과 관련

size

- 8바이트
- 현재 청크의 크기
- 데이터의 크기 + 헤더의 크기

- (64비트 환경에서 할당된 청크의 size는) 데이터의 크기 + 16바이트
- 64비트 환경에서 청크의 size는 16바이트의 배수, 32비트에서는 8바이트의 배수 → size의 뒷 세 자리는 000대신 flags의 정보를 표기

flags

- 3비트
- **PREV_INUSE (P)**
  - 직전 청크가 할당된 상태 or fastbin인가
- **IS_MAPPED(M)**
  - mmap() 함수를 통해 할당된 청크인가
- **NON_MAIN_ARENA(N)**
  - 멀티쓰레드 환경에서 청크가 main에 위치하지 않았는가

fd

- 8바이트
- forward pointer
- 다음 청크의 주소

bk

- 8바이트
- backward pointer
- 이전 청크의 주소

▼ bin

bin은 **해체된 청크**들이 저장되는 공간이다.

ptmalloc 기준 128개의 bin이 정의되어 있다.

bin의 종류로 unsortedbin, smallbin, largebin, fastbin이 있다.

bin	name
0	-
1	unsortedbin(1개)
2~63	smallbin(62개)
64~126	largebin(63개)
127	-

fastbin

- 일정 크기 이하의 **작은 청크들**에 대해 **빠른 속도**의 할당/해체를 위함
- **32바이트 ~ 176바이트**의 범위로 16바이트 단위 총 10개의 fastbin이 존재
- 리눅스 기준 **32바이트 ~ 128바이트**의 범위로 **7개**의 fastbin 사용

- **단일** 연결리스트 → `unlink`가 필요없어 속도가 빠름
- **LIFO** 방식
- **병합 x**
- `global_max_fast` 변수를 조작하여 처리하는 메모리의 최대 크기를 조작할 수 있음

unsortedbin

- **비슷한 크기**의 청크 반복 할당 및 해제를 **효율적**으로 처리하기 위함
- **분류되지 않은 청크**들을 보관
- **하나**만 존재
- **원형 이중** 연결리스트 → `unlink` 필요
- fastbin에 들어가지 않는 **모든 청크들이 대상**
- **smallbin 크기**의 경우 fastbin과 smallbin을 탐색한 후 unsortedbin을 탐색
- **largebin 크기**의 경우 unsortedbin을 먼저 탐색

smallbin

- **32바이트 이상 ~ 1024바이트 미만**의 청크를 보관하는 bin
- **하나의 인덱스마다 같은 크기**의 청크를 보관
- smallbin[0] : 32바이트, smallbin[61] : 1008바이트 (**62**개 존재)

	속도	파편화
LIFO	fast	many
FIFO	-	-
address-ordered	slow	less

- **원형 이중** 연결리스트 → `unlink` 필요
- **FIFO** 방식
- **병합 o** → smallbin의 두 해체된 청크가 메모리상에서 인접할 시 `consolidation` 발생

largebin

- **1024바이트 이상**의 청크를 보관하는 bin
- **63**개의 largebin이 존재
- 하나의 인덱스마다 **일정 범위 안의 청크들**을 보관 → 크기를 **내림차순으로 정렬**
- 재할당 요청 발생시 크기가 **가장 비슷한 청크**(best-fit)를 재할당
- **이중** 연결 리스트 → `unlink` 필요
- **병합 o** → largebin의 두 해체된 청크가 메모리상에서 인접할 시 `consolidation` 발생

▼ arena

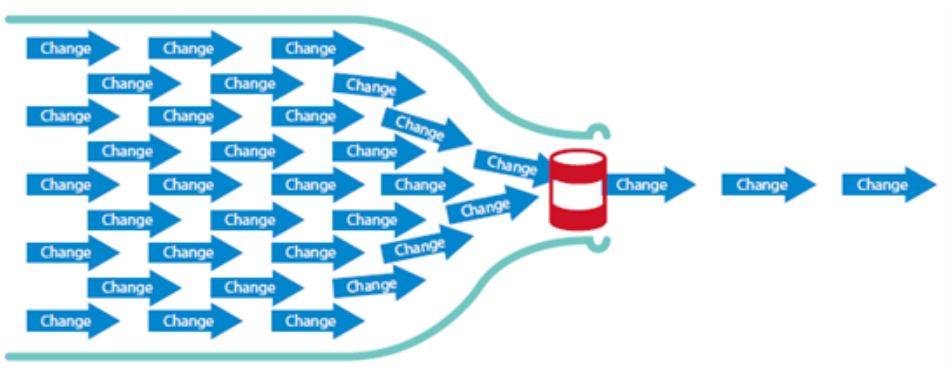
fastbin, smallbin,largebin 등의 정보를 담고 있는 공간이다.

멀티 쓰레드 환경을 지원하기 위해 존재한다.

최대 64개로 제한되어 있다.

레이스 컨디션

멀티 쓰레드 환경에서 여러 쓰레드가 같은 공간을 접근하면 삭제나 수정 과정에서 오류가 발생한다.
(레이스 컨디션)→ 락의 개념을 도입하여 삭제나 수정 시 다른 쓰레드를 대기시킨다.



다만 이러한 방식의 레이스 컨디션 해결은 병목 현상을 불러온다.

데드락 : 서로 락 상태가 걸려 어떤 쓰레드도 락을 해제하지 못하는 상황

▼ tcache

- **thread local cache**. 각 쓰레드에 **독립적**으로 할당되는 캐시 저장소 (**고유한 정보**)
- **32바이트 이상 ~ 1040바이트 이하** (**64개** 존재)
- 이 범위의 크기라면 tcache를 **가장 먼저** 할당 및 해체
- **단일** 연결리스트, 각 tcache당 **7개의 갯수**로 제한
- **병합 x**
- **LIFO** 방식
- 고유의 저장소 → **레이스 컨디션 x, 병목 현상 x**
- **보안 검사가 많이 없어 취약점 많음**