# PWN - Glibc Heap Basics

Pedro Bernardo
@bl4ck_pwn

# Heap: Overview

- Pool of memory used for dynamic allocation at runtime
  - malloc - grabs memory from the heap
  - free - releases memory on the heap

# Heap: Overview

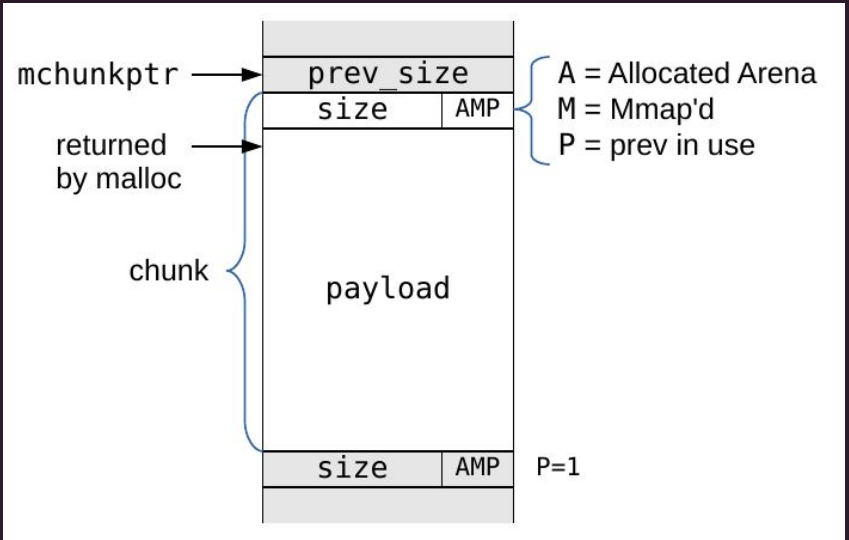
```
gef➤  vmmap
[ Legend:  Code | Heap | Stack ]
Start              End                Offset             Perm Path
0x0000555555554000 0x0000555555555000 0x0000000000000000 r-- /home/bl4ck/wip/how2heap/glibc_2.31/fastbin_dup
0x0000555555555000 0x0000555555556000 0x0000000000001000 r-x /home/bl4ck/wip/how2heap/glibc_2.31/fastbin_dup
0x0000555555556000 0x0000555555557000 0x0000000000002000 r-- /home/bl4ck/wip/how2heap/glibc_2.31/fastbin_dup
0x0000555555557000 0x0000555555558000 0x0000000000002000 r-- /home/bl4ck/wip/how2heap/glibc_2.31/fastbin_dup
0x0000555555558000 0x0000555555559000 0x0000000000003000 rw- /home/bl4ck/wip/how2heap/glibc_2.31/fastbin_dup
0x0000555555559000 0x000055555557a000 0x0000000000000000 rw- [heap]
0x00007ffff7dd2000 0x00007ffff7dd5000 0x0000000000000000 rw-
0x00007ffff7dd5000 0x00007ffff7dfb000 0x0000000000000000 r-- /usr/lib/libc-2.33.so
0x00007ffff7dfb000 0x00007ffff7f46000 0x0000000000026000 r-x /usr/lib/libc-2.33.so
0x00007ffff7f46000 0x00007ffff7f92000 0x0000000000171000 r-- /usr/lib/libc-2.33.so
0x00007ffff7f92000 0x00007ffff7f95000 0x00000000001bc000 r-- /usr/lib/libc-2.33.so
0x00007ffff7f95000 0x00007ffff7f98000 0x00000000001bf000 rw- /usr/lib/libc-2.33.so
0x00007ffff7f98000 0x00007ffff7fa1000 0x0000000000000000 rw-
0x00007ffff7fa1000 0x00007ffff7fa3000 0x0000000000000000 r-- /usr/lib/libdl-2.33.so
0x00007ffff7fa3000 0x00007ffff7fa5000 0x0000000000002000 r-x /usr/lib/libdl-2.33.so
0x00007ffff7fa5000 0x00007ffff7fa6000 0x0000000000004000 r-- /usr/lib/libdl-2.33.so
0x00007ffff7fa6000 0x00007ffff7fa7000 0x0000000000004000 r-- /usr/lib/libdl-2.33.so
0x00007ffff7fa7000 0x00007ffff7fa8000 0x0000000000005000 rw- /usr/lib/libdl-2.33.so
0x00007ffff7fa8000 0x00007ffff7faa000 0x0000000000000000 rw-
0x00007ffff7fc7000 0x00007ffff7fcb000 0x0000000000000000 r-- [vvar]
0x00007ffff7fcb000 0x00007ffff7fcd000 0x0000000000000000 r-x [vdso]
0x00007ffff7fcd000 0x00007ffff7fce000 0x0000000000000000 r-- /usr/lib/ld-2.33.so
0x00007ffff7fce000 0x00007ffff7ff2000 0x0000000000001000 r-x /usr/lib/ld-2.33.so
0x00007ffff7ff2000 0x00007ffff7ffb000 0x0000000000025000 r-- /usr/lib/ld-2.33.so
0x00007ffff7ffb000 0x00007ffff7ffd000 0x000000000002d000 r-- /usr/lib/ld-2.33.so
0x00007ffff7ffd000 0x00007ffff7fff000 0x000000000002f000 rw- /usr/lib/ld-2.33.so
0x00007ffffffde000 0x00007ffffffff000 0x0000000000000000 rw- [stack]
0xffffffffff600000 0xffffffffff601000 0x0000000000000000 --x [vsyscall]
```

# Heap: Chunks

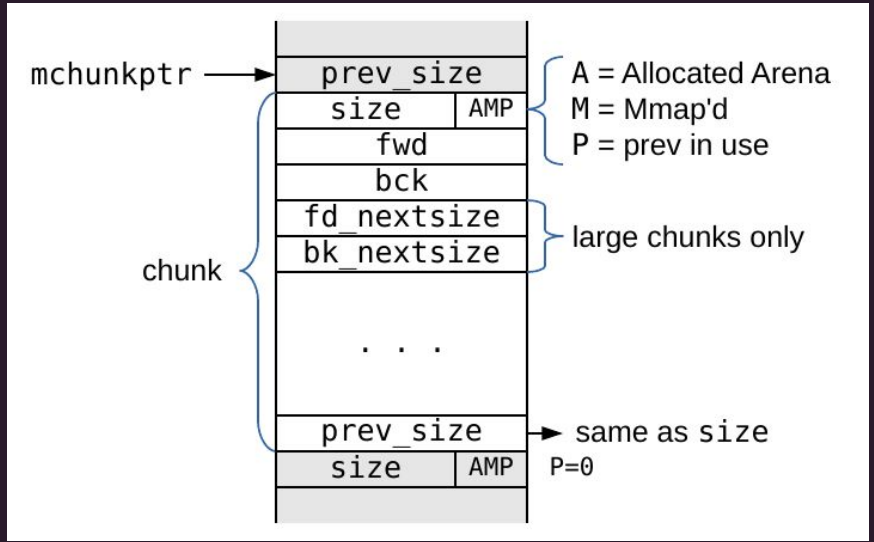
- Consists of **Heap Chunks**, and there are different types:
  - Allocated Chunk
  - Free Chunk
  - Top Chunk
  - Last Remainder Chunk

# Heap: Chunks

## Allocated Chunk




## Free Chunk




Source: https://www.sourceware.org/glibc/wiki/MallocInternals

# Heap: Top Chunk

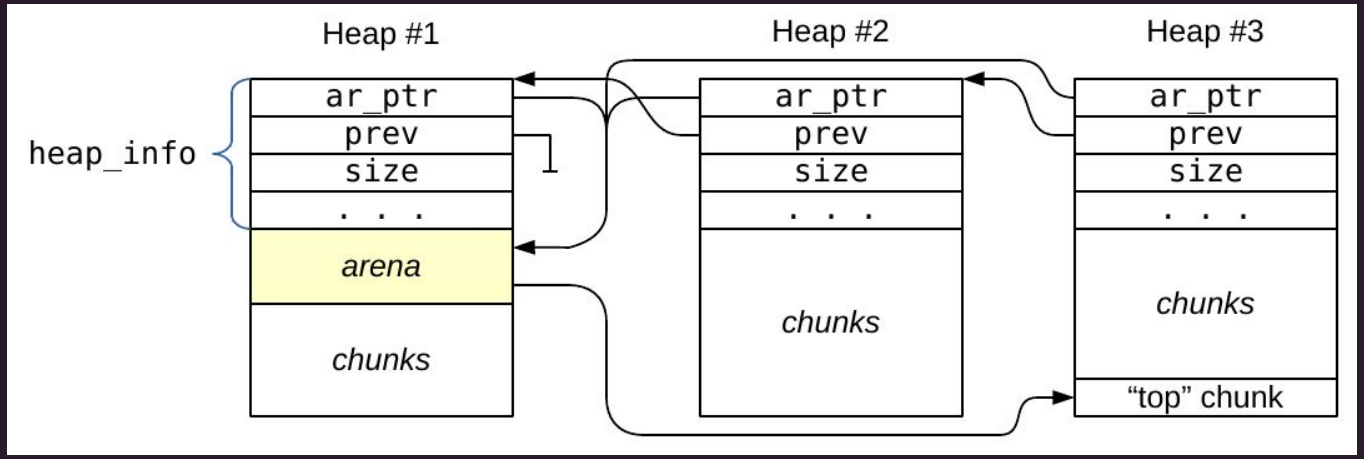
- Used to service user requests when there are **NO FREE CHUNKS**
- Features:
  - If top_chunk->size > requested->size, it is split in two
    - User chunk (requested size)
    - Remainder Chunk (of remaining size)
  - Else the top chunk is extended using sbrk or mmap

# Heap: Coalescing

- Two chunks which are free can’t be adjacent
  - Combined into a single free chunk
- Why?
  - Eliminates fragmentation
  - Makes free slower

# Arenas and Heaps

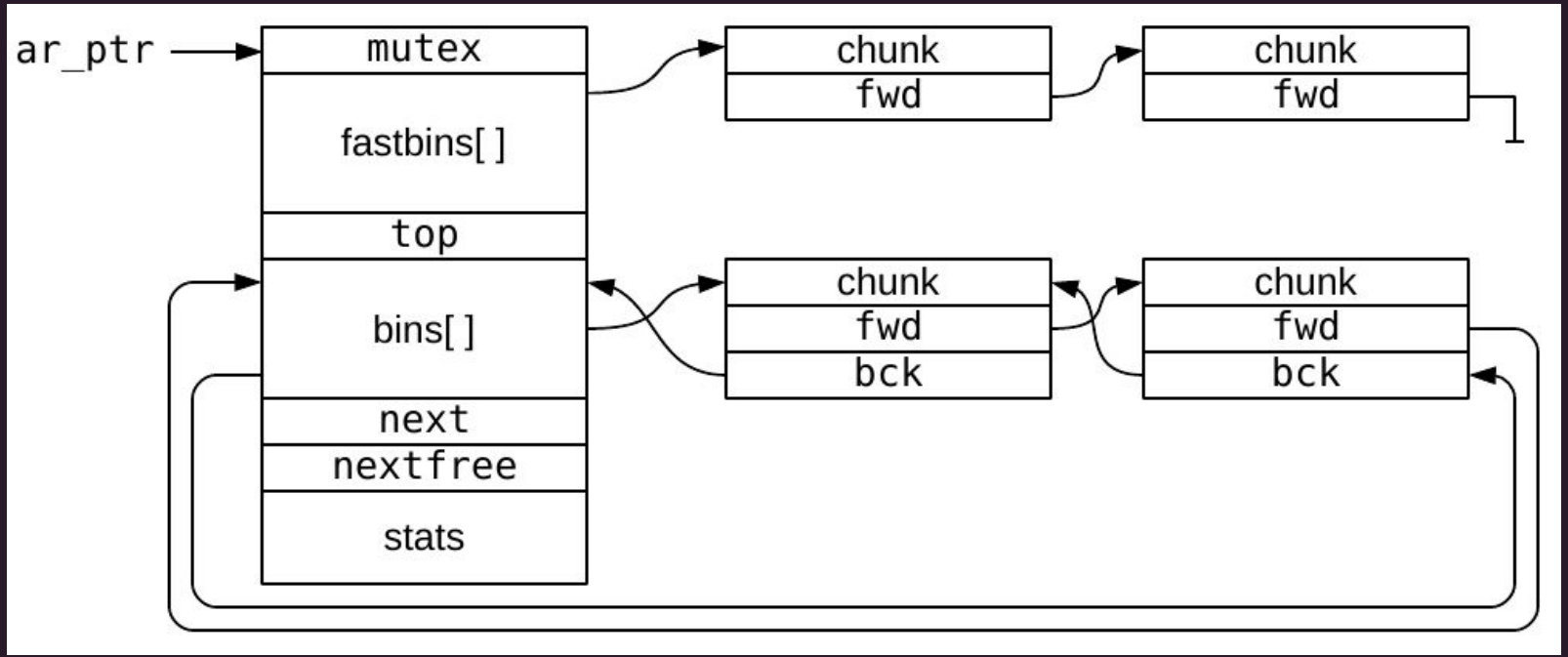
- Main arena - Initial  heap's arena
  - We are usually interested in this one
- More than one region can be managed by malloc
  - Why? Multi-threading
  - These regions are called Arenas



# Main Arena: Bins

- Bins:
  - Free-list structures
  - Hold free chunks
- Different types, based on chunk size and history:
  - Fast bin (not coalesced)
  - Small bin
  - Large bin
  - Unsorted bin

# Main Arena: Bins



# Main Arena: Bins

| Bins | Linked List Type | Chunk Size Range | Coalescing |
|---|---|---|---|
| Fast | Singly-linked | 16 – 80 bytes | ❌ |
| Small | Doubly-linked | 80 – 512 bytes | ✅ |
| Large | Doubly-linked | 512+ bytes | ✅ |
| Unsorted | Doubly-linked | Small and Large chunks | ❌ |

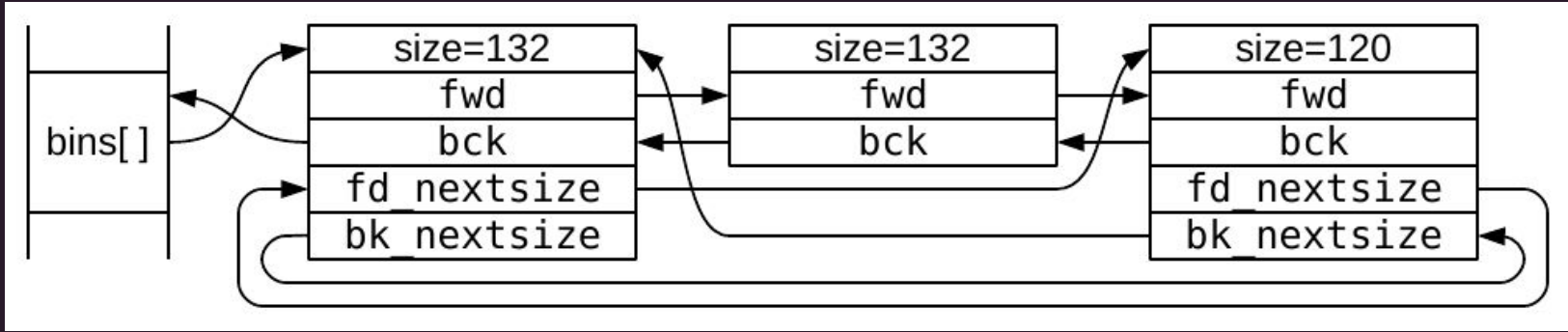
# Unsorted Bins

- When a small or large chunk gets freed, it is added to the Unsorted Bin
- Why?
  - Helps speed up memory allocation

# Large Chunks: Special Cookies

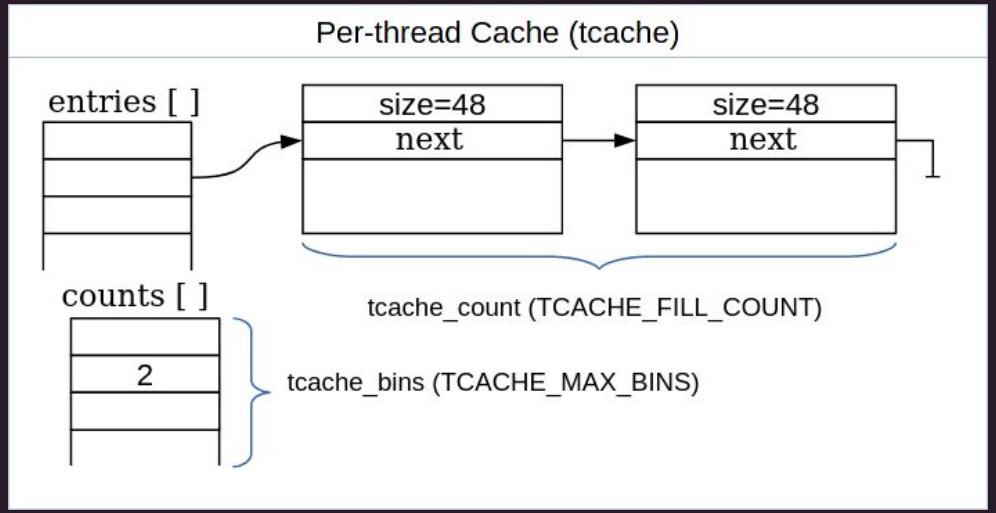
Large chunks have extra fields so they can best fit user requests:

- Size - to keep an ordered list (descending order)
- Allows malloc to quickly search for the first *big enough* chunk



# Thread Local Cache (tcache)

- Introduced in glibc 2.26 to improve heap performance
- Structure stored on the heap
- Similar to Fast Bins



# Thread Local Cache: tcache_put

```
static __always_inline void
tcache_put (mchunkptr chunk, size_t tc_idx)
{
  tcache_entry *e = (tcache_entry *) chunk2mem (chunk);

  /* Mark this chunk as "in the tcache" so the test in _int_free will
      detect a double free.  */
  e->key = tcache;

  e->next = tcache->entries[tc_idx];
  tcache->entries[tc_idx] = e;
  ++(tcache->counts[tc_idx]);
}
```

# Thread Local Cache: tcache_get

```
static __always_inline void *
tcache_get (size_t tc_idx)
{
  tcache_entry *e = tcache->entries[tc_idx];
  tcache->entries[tc_idx] = e->next;
  --(tcache->counts[tc_idx]);
  e->key = NULL;
  return (void *) e;
}
```

# Heap Vulnerabilities

3 main categories:

- Double-free
- Use-After-Free
- Overflow

Techniques:

- Unsafe-unlink (kind of patched?)
- Fastbin and Tcache dup/poison
- Poison null-byte
- Overlapping chunks
- House of <insert random word>
  - House of Force
  - House of Heinerjar
  - House of Spirit

Source: https://github.com/shellphish/how2heap

# Resources

- Malloc security checks - https://heap-exploitation.dhavalkapil.com/diving_into_glibc_heap/security_checks
- Malloc internals - https://www.sourceware.org/glibc/wiki/MallocInternals
- How2heap - https://github.com/shellphish/how2heap
- Glibc source code - https://elixir.bootlin.com/glibc/latest/source
- Temple of PWN - https://www.youtube.com/playlist?list=PLiCcguURxSpbD9M0ha-Mvs-vLYt-VKlWt
- LiveOverflow - https://www.youtube.com/playlist?list=PLhixgUqwRTjxglIswKp9mpkfPNfHkzyeN
- GEF gdb extension - https://github.com/hugsy/gef