Beijing Forest Studio
北京理工大学信息系统及安全对抗实验中心

Fast bin attack & Unsorted bin attack

硕士研究生 侯留洋

2019年01月13日

内容提要

- 基础知识
  - 堆管理基础
- 利用技巧
  - Fast bin attack
  - Unsorted bin attack
  - Fast for Unsorted
- 总结分析
- 参考资料

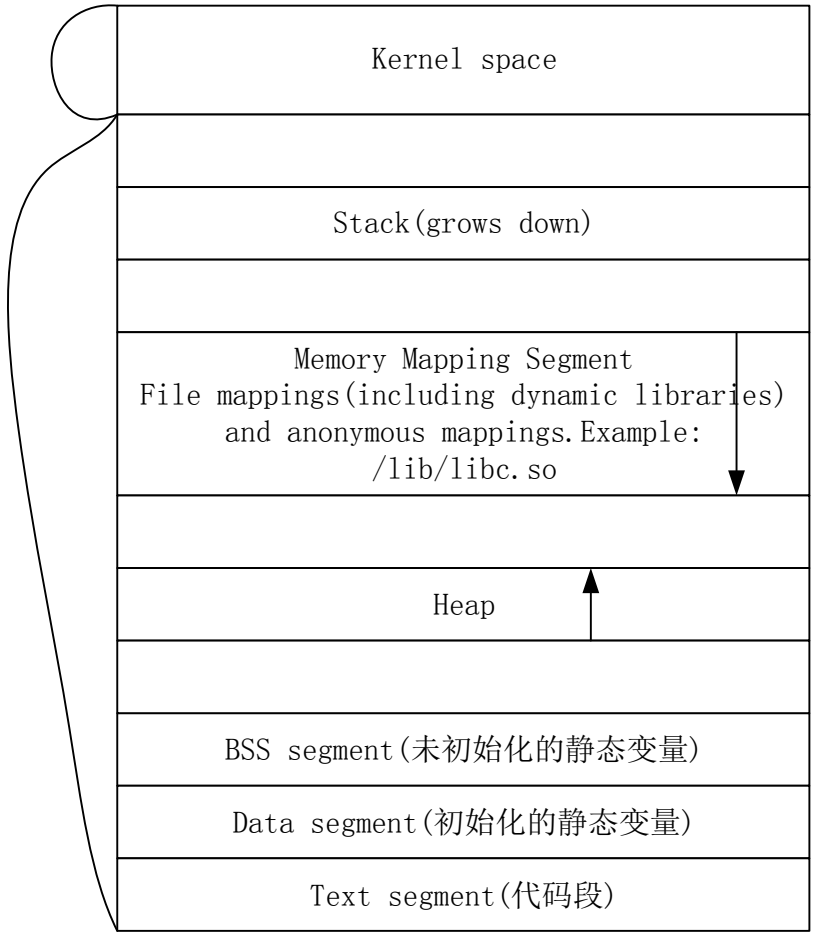
预期收获

- 预期收获
  - 1. 了解堆结构及堆内存的分配回收机制
  - 2. 了解堆上常见漏洞利用技巧

堆管理基础

- 堆的位置

```
int *a = malloc(0x10);
int *b = malloc(0x20);
int *c = malloc(0x100);
```



Linux堆内存管理-曲乐炜-2016-11-27 19_00_00

堆管理基础

- Heap allocator管理堆的分配和回收
- Why?
  - 减少系统调用次数
  - 提高效率

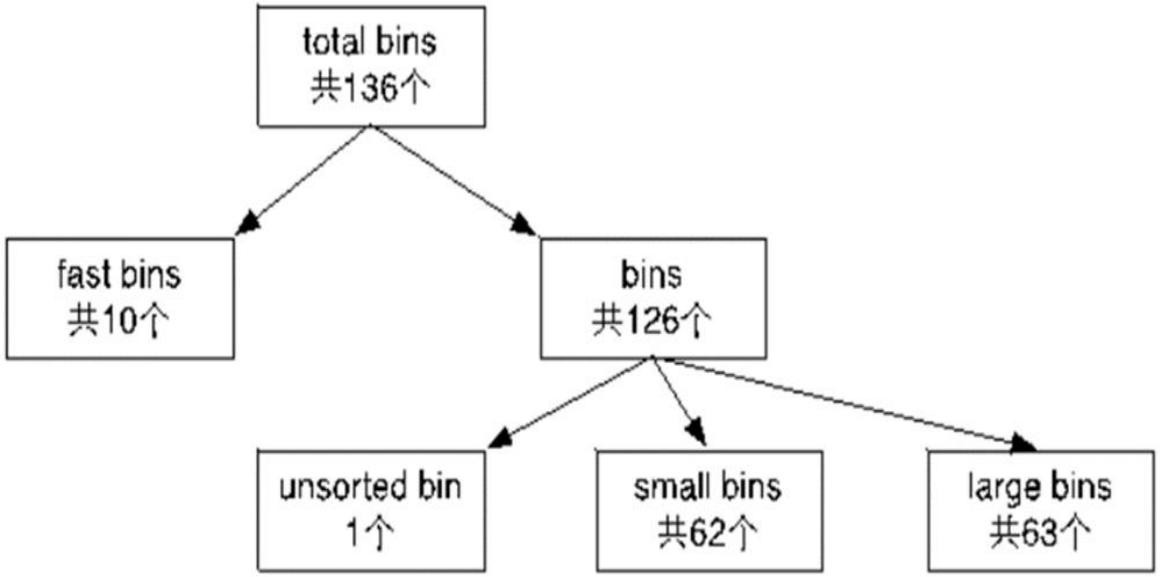
堆管理基础

- 不同的系统有不同的管理方式

- Linux下：Glibc — ptmalloc2
  - 主要结构：chunk、bin、arena
  - 使用bins来管理空闲堆块
  - Arena管理bins

堆管理基础

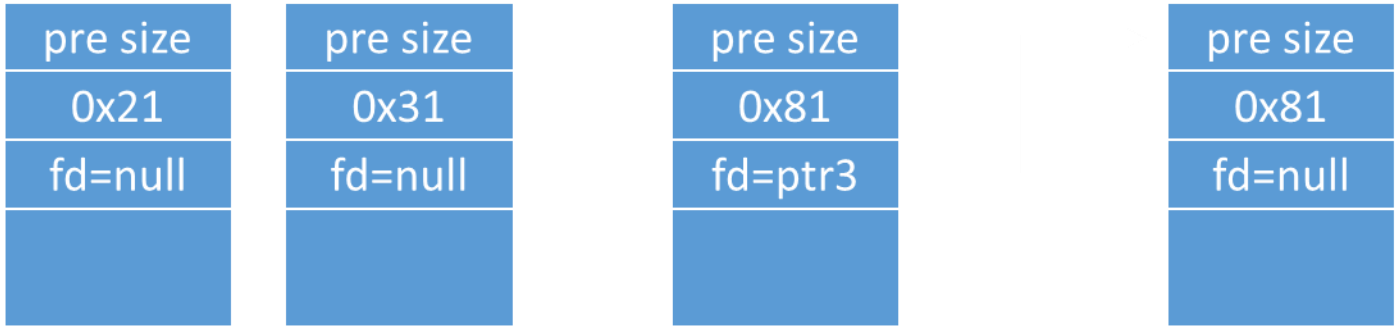
- 用来管理空闲块链表结构
  - Arena
    - 对应的数据结构是malloc_state
- 不同大小的free chunk属于不同的bin
  - Fast bins
  - Flagd+topchu
  - Unsorted bins
  - Small bins
  - Large bins



Fast bin attack

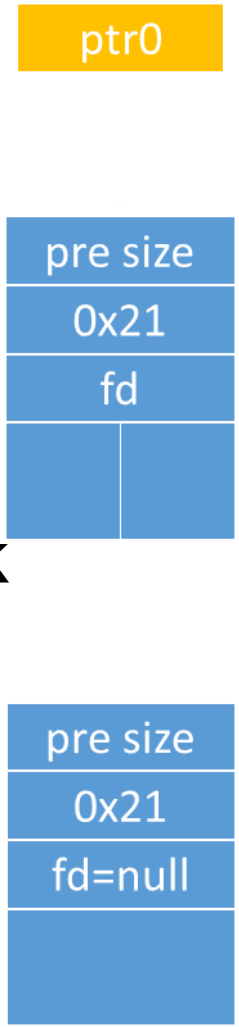
Fast bin

- 管理free fast bin chunk，单链表结构，FILO(first in last out)
- 总共有10个bins，每个bin中chunk大小一样，范围0x20~0x80(64bit)，0x10递增
- 仍被标记pre inuse位，不会进行合并



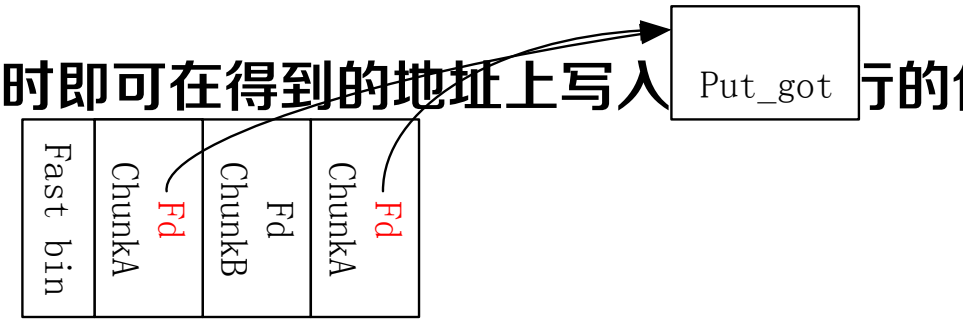
Fast bin attack

- 关键点：控制fd指针
- Size检查
  - 利用misalignment伪造size => 0x7fXXXXXXXXXX
- Overwrite GOT
- Overwrite realloc_hook / malloc_hook



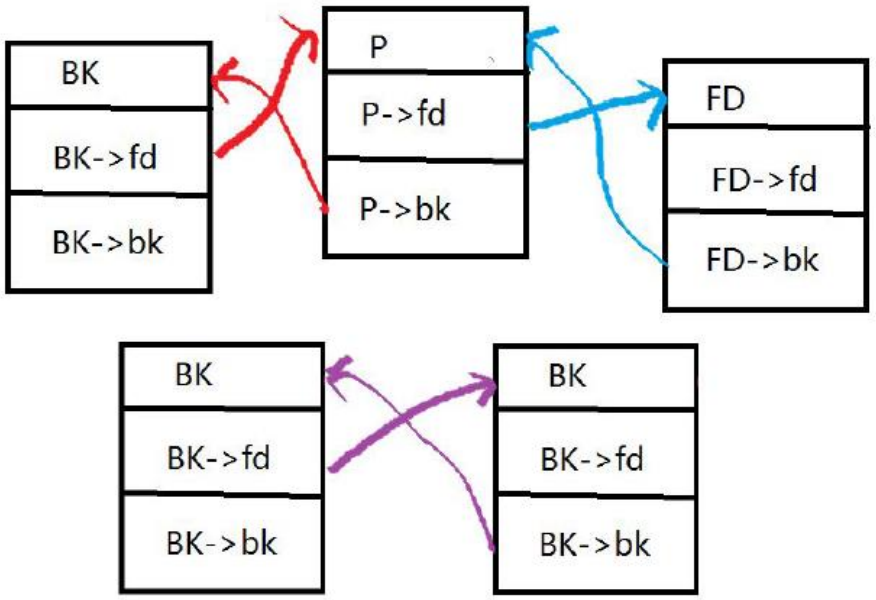
Fast bin attack

- 一种攻击场景
  - 构造fast bin ->chunkA->chunkB->chunkA
  - 申请chunkA，并edit其fd指针为目标地址，如malloc_hook_addr（malloc_hook函数地址）
  - 此时fast bin ->chunkB->chunkA->malloc_hook_addr
  - 再Malloc第三次得到malloc_hook_addr开始的内存地址
    - 此时即可在得到的地址上写入Put_got行的代码的地址



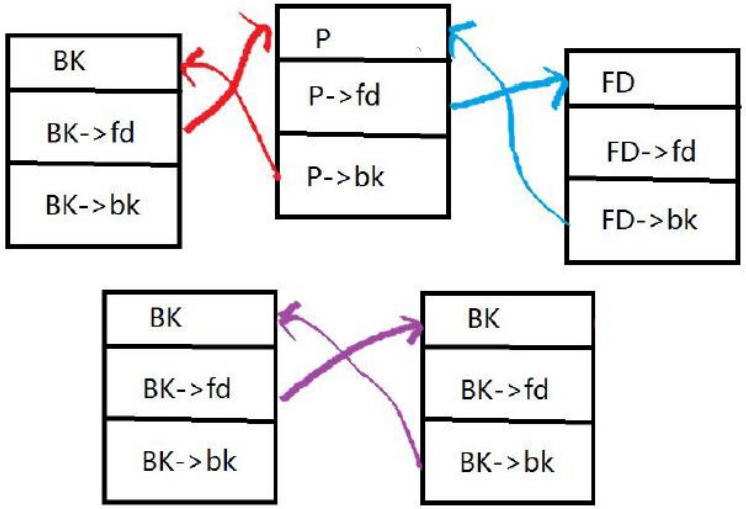
Unsorted bin attack

- 卸载unsort bin chunk
  - 如图所示
- 卸载过程
  - BK=P->bk
  - FD=P->fd
  - FD->bk=BK
  - BK->fd=FD



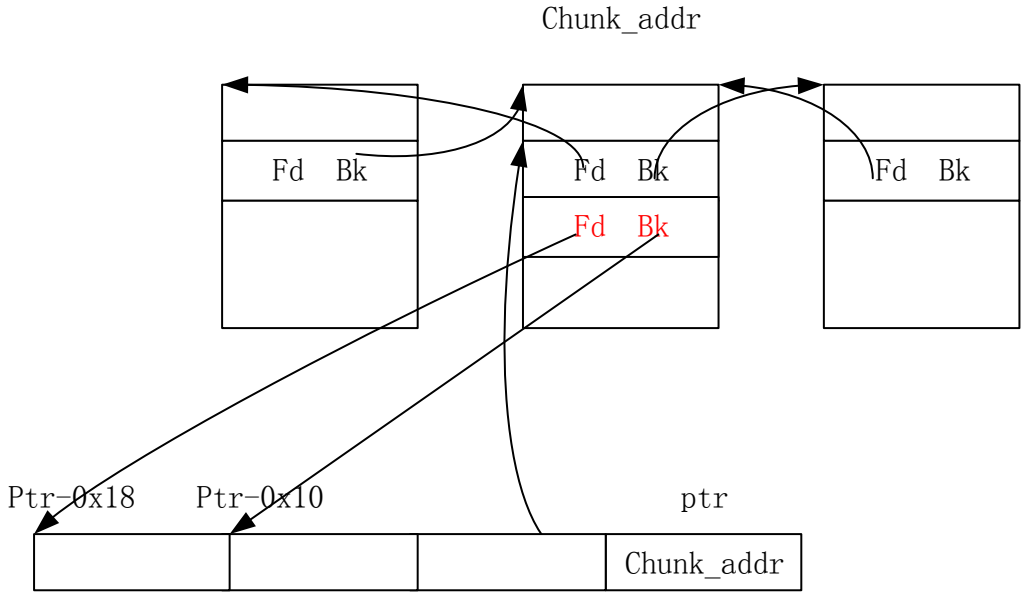
Unsorted bin attack

- 卸载过程
  - FD->bk=BK
    - *[*(P->fd)+0x18)]= *(P->bk),即在P的fd指针的值的偏移0x18地址处，放p的bk指针的值
  - BK->fd=FD
    - *[*(P->bk)+0x10]= *(P->fd)),即在P的bk指针的值的偏移0x10地址处，放p的fd指针的值
- 结果
  - [[P->bk]+0x10]=[P->fd]
    - 可以在p->bk处存放想要改写的地址，如put_got；p->fd处放想要写入的内容。



Unsorted bin attack

- Safe unlink
  - 在卸载chunk时，检测自己是否在链表上，即前一个chunk的后向指针和后一个chunk的前向指针都需要指向自己的地址



  - 一种绕过方法
    - 一般申请的chunk地址都会放到某变量，用该变量的地址绕过检测

Fast for Unsorted

Fast for Unsorted

- 背景
  - Malloc:能创建10个大小为0x30的堆块,并能进行写入
  - Edit:只能进行三次edit
  - Free:存在UAF漏洞
  - Show:可进行信息泄露

```
if ( (unsigned int)v1 <= 9 && !*(_QWORD *)&ptr[8 * v1] )
{
  *(_QWORD *)&ptr[8 * v1] = malloc(0x20uLL);
```

图1. 10次malloc

```
if ( (unsigned int)v1 <= 0x1F && *(_QWORD *)&ptr[8 * v1] && dword_6020B0 != 3 )
{
  printf("Content:", &s);
  sub_40092B(*(_QWORD *)&ptr[8 * v1], 0x20u);
```

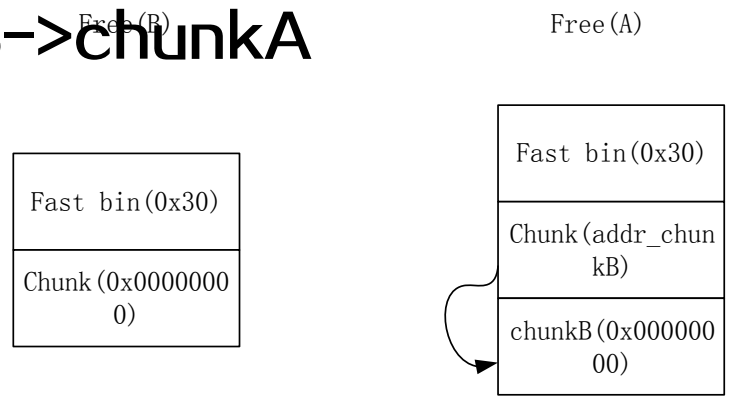
图2. 3次edit

- 目的
  - 如何利用10次0x30大小的堆块创建及三次的edit完成任意地址的写入?

堆溢出漏洞利用原理分析-尚海-2017-07-23 19_00_00

Fast for Unsorted

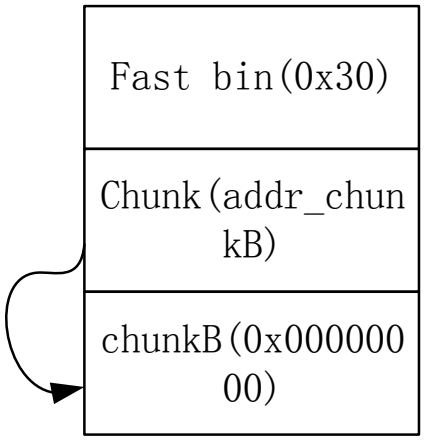
- 地址泄漏
  - A=malloc(0x20),B=malloc(0x20), C=malloc(0x20)
  - Free(B),Free(A)
    - Fast bin->chunkB->chunkA
  - Show(B)
- 用掉3次malloc

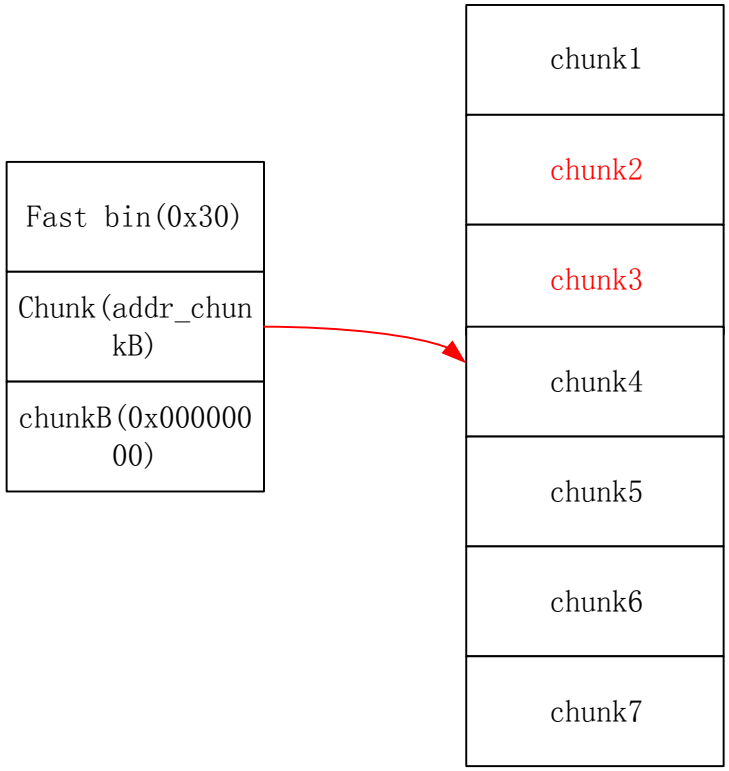



```
0000000000082D030  00 00 00 00 00 00 00 00  31 00 00 00 00 00 00 00
0000000000082D040  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0000000000082D050  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
0000000000082D060  00 00 00 00 00 00 00 00  31 00 00 00 00 00 00 00
0000000000082D070  30 D0 82 00 00 00 00 00  00 00 00 00 00 00 00 00
0000000000082D080  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

Fast for Unsorted

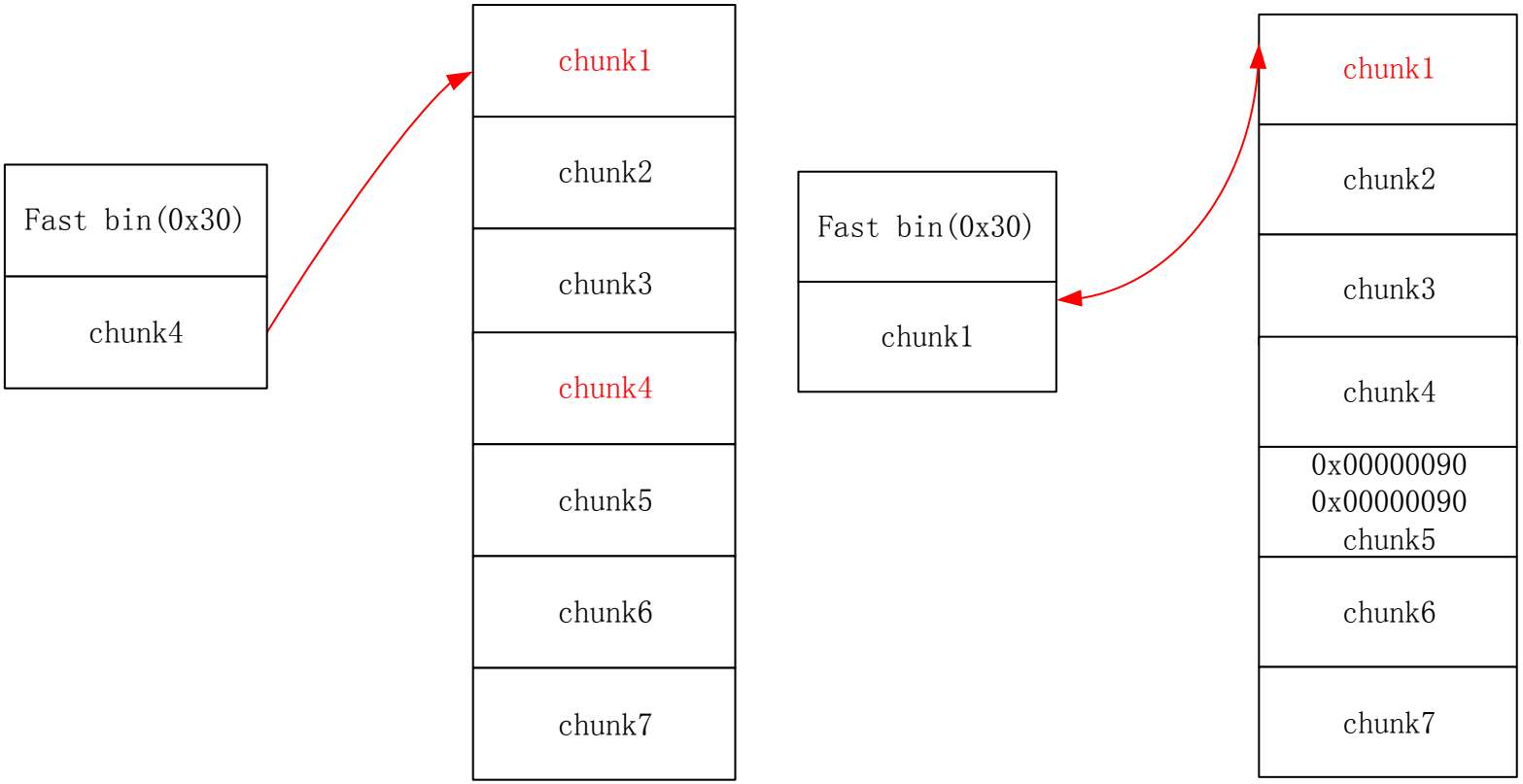
- 构造unsorted chunk
  - Malloc(1),Malloc(2),…,Malloc(7)
  - Free(2),Free(3),show(3)
  - Edit1(3,p64(chunk1_addr+0xa0))





Fast for Unsorted

- 构造unsorted chunk
  - Malloc(8), Edit2(4,p64(0x31)*2+p64(heap+0x20))
  - Malloc(9, p64(0x90)*3+chr(0x90))



Fast for Unsorted

- 构造unsorted chunk
  - Malloc(0, p64(0x0)+p64(0x91)+p64(0x6020a8-0x18)+p32(0x6020a8-0x10))

存储堆块地址的变量

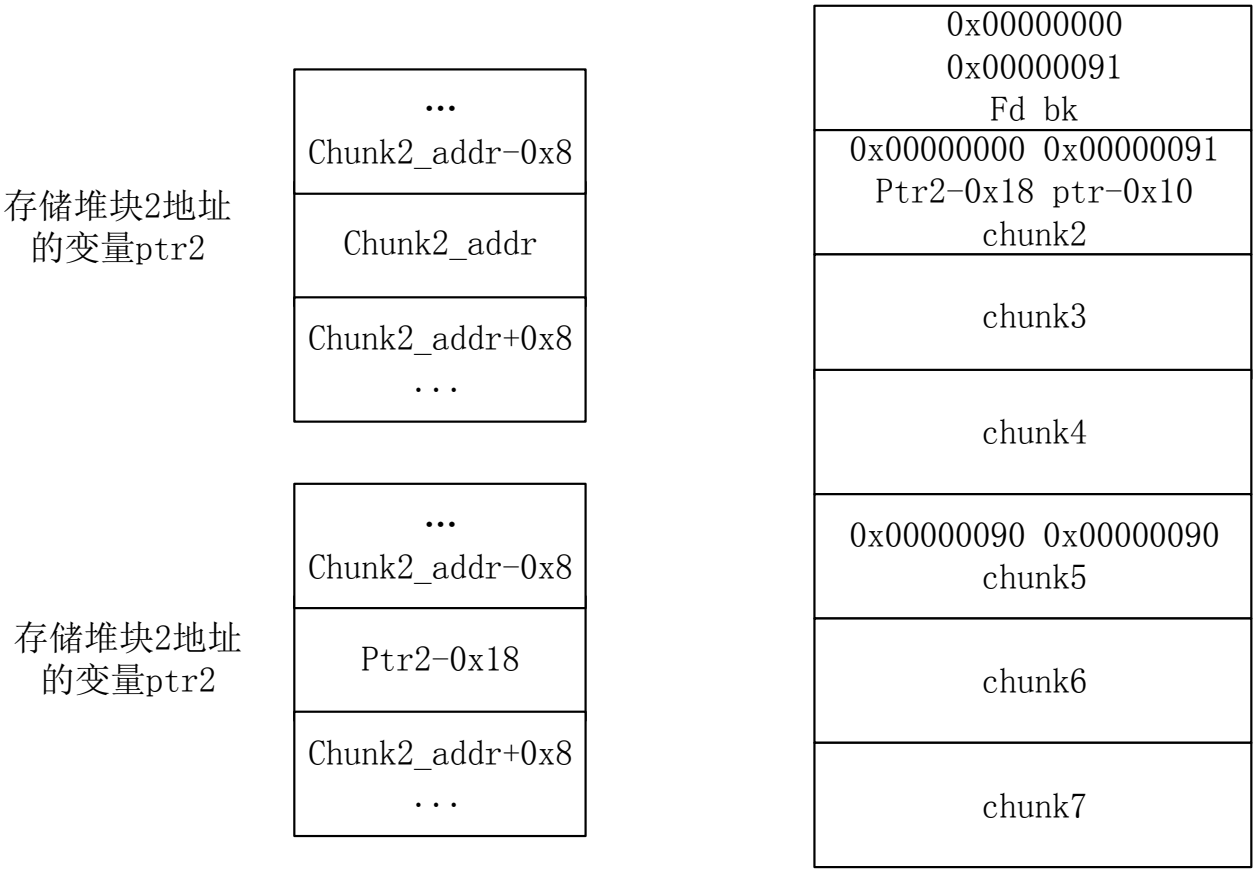
… Ptr2-0x8
ptr2
Ptr2+0x8 …

0x00000000 0x00000091 Fd bk
0x00000000 0x00000091 Ptr2-0x18 ptr-0x10 chunk2
chunk3
chunk4
0x00000090 0x00000090 chunk5
chunk6
chunk7

- Unsorted bin attack
  - Free(5)
    - 触发unlink

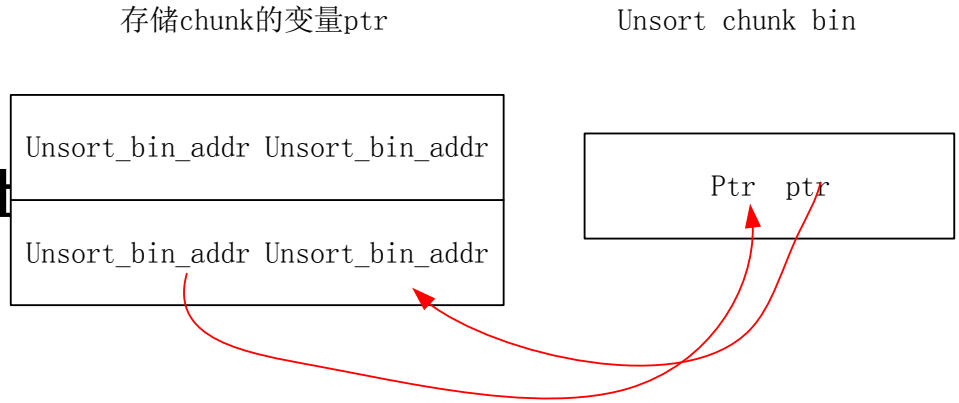
Fast for Unsorted

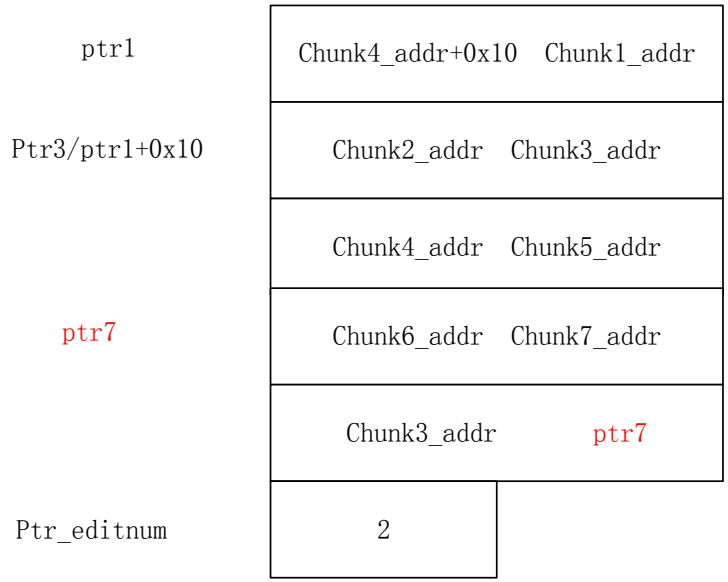
- 存放堆块地址的变量被修改

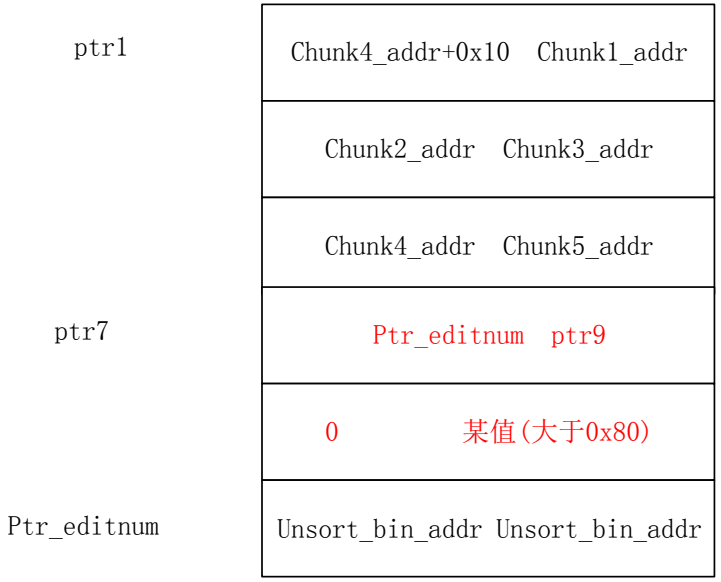


Fast for Unsorted

- Ptr处的任意地址写
  - Free unsort chunk
    - 得到unsort_bin地址
    - 同时覆写了edit限制



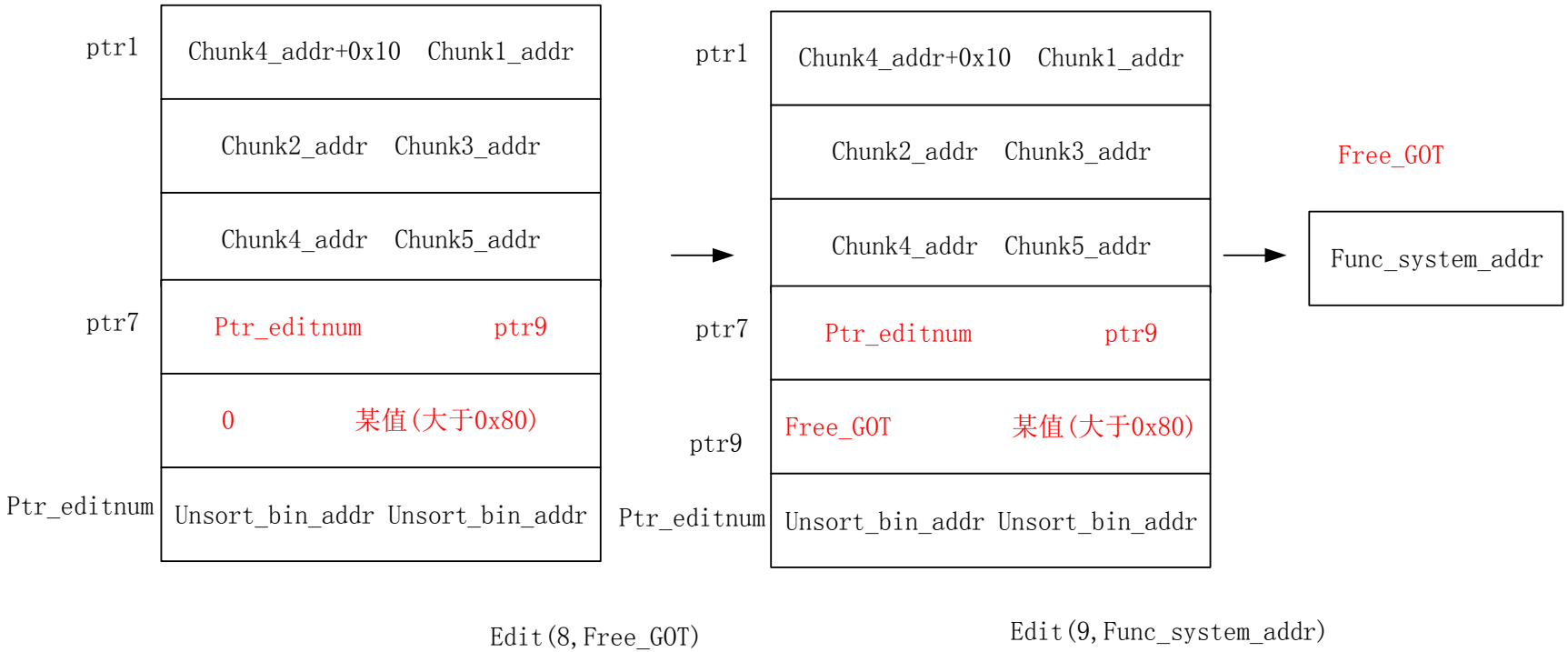




常规unsort bin attack 突破edit限制

Fast for Unsorted

- 任意地址写
  - Free_GOT表写入system函数地址



Fast for Unsorted

- Free被hook为system函数
  - Edit(1,” /bin/sh”)
  - Free(1)
    - System(“/bin/sh”)

总结分析

总结分析

- Fast bin attack
  - 单链表卸载chunk块，由指针操作，覆写指针可实现任意地址写
- Unsorted bin attack
  - 双向链表卸载chunk块，指针操作，第二步覆写实现任意地址的写
- 二者结合
  - 都有一定验证保护
  - 由于二者有chunk块大小不同的限制，可以在某些场合下进行相互辅助
    - Unsort bin attack分配大的chunk，可以在其上构造fast bin attack的攻击场景

参考文献

参考文献

- https://bbs.pediy.com/thread-246391.htm
- https://bbs.pediy.com/thread-218313.htm

上善若水。水善利万物而不争，处众人之所恶，故几於道。居善地，心善渊与善仁，言善信，正善治，事善能，动善时。夫唯不争，故无尤。

谢谢！