请求分配大小	实际分配大小	分析与说明
96 字节	128B	使用SLAB分配器，以128B为最小分配单位， 96B+32B元数据正好为128B
128 字节	256B	需要128+32B空间，分配128B*2
256 字节	384B	需要256+32B空间，分配128B*3
4064 字节	4096B	使用伙伴算法页分配器，分配4064+32B
4096 字节	8192B	需要4096+32B空间，分配两个4KB页

最小分配单元： slab为128B，伙伴算法分配单元为一个页。

分配器选择：

```
#define IS_SMALL(size) ((size) < (__PAGE_SIZE / 5)) //判断分配请求是否大于页面大小/5
...
if (IS_SMALL(realsize)) {
    num_pages = size_to_s_num_pages(realsize);
    intptr = (__uptr)uk_salloc(a, num_pages);
    uk_pr_err("alloc size => %llu, num_pages => %llu, intpter => %p\n",
              realsize, num_pages, intptr);
  } else {
    num_pages = size_to_num_pages(realsize);
    intptr = (__uptr)uk_palloc(a, num_pages);
  } //根据realsize选择slab或者页
```

大内存分配问题：

`uk_free_ifpages(struct uk_alloc *a, void *ptr, const void *small)` 函数中small为空值，导致

```
if (small) {
    uk_sfree(a, metadata->base, metadata->num_pages);
  } else {
    uk_pfree(a, metadata->base, metadata->num_pages);
  }
```

进入错误分支， gdb中set small参数的值即可进入正确分支

```
SUCCESS: Memory freed correctly after GDB intervention!
[    5.125128] ERR:  [libukallocbbuddy] <bbuddy.c @
 320> Free list 8 is empty.
[    5.125870] ERR:  [libukallocbbuddy] <bbuddy.c @
 322> Free list 9 (Order 9):

   - SUCCESS: Memory freed correctly after GDB inter
vention!

================================================
Analysis finished. The system will now halt.
```