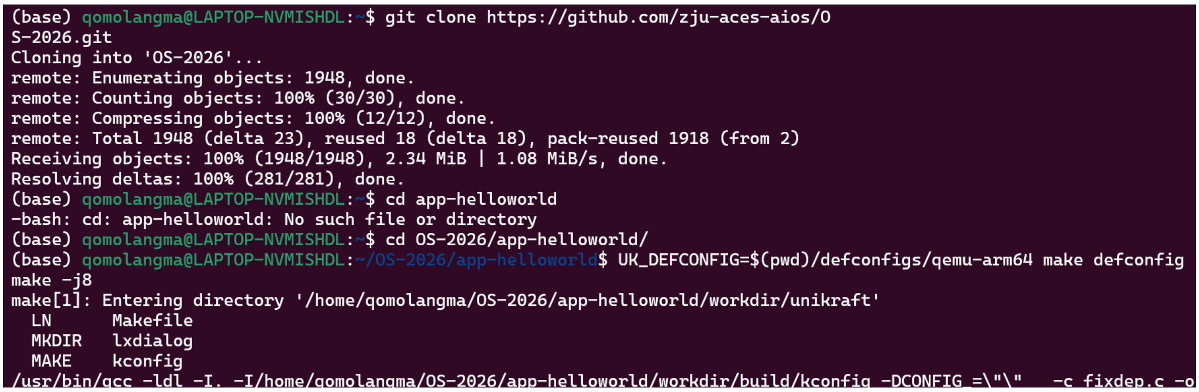
实验过程

```
git clone https://github.com/zju-aces-aios/OS-2026.git
cd OS-2026/app-helloworld
```

```
UK_DEFCONFIG=$(pwd)/defconfigs/qemu-arm64 make defconfig
make -j8
```



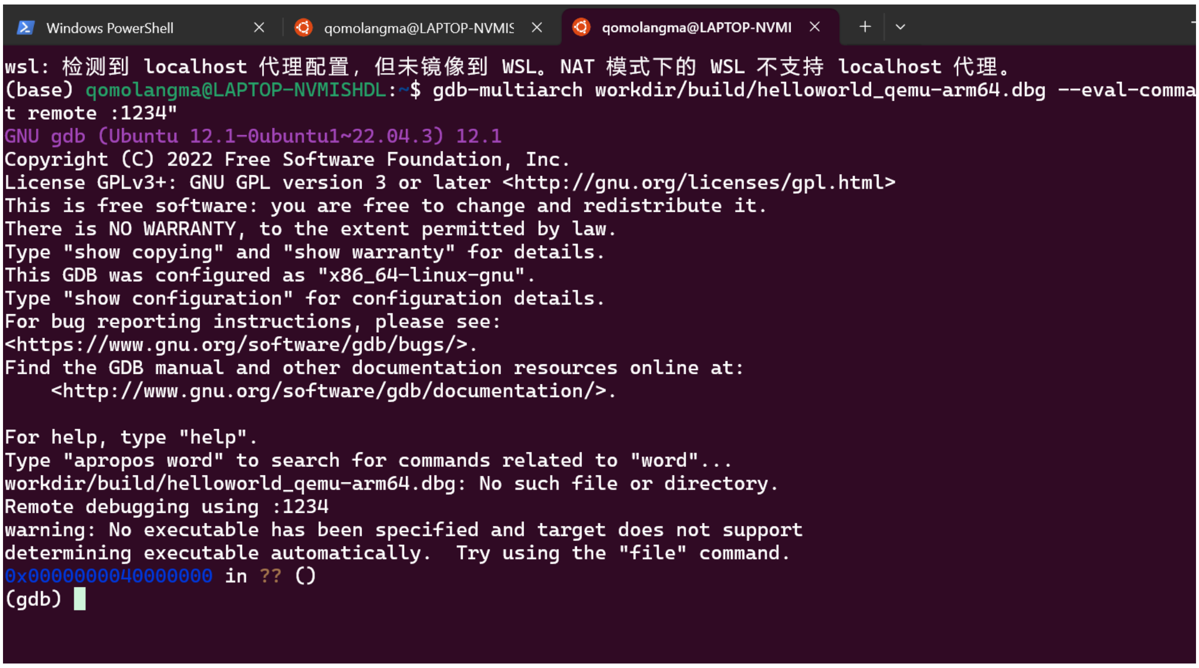
```
(base) qomolangma@LAPTOP-NVMISHDL:~$ git clone https://github.com/zju-aces-aios/O
S-2026.git
Cloning into 'OS-2026'...
remote: Enumerating objects: 1948, done.
remote: Counting objects: 100% (30/30), done.
remote: Compressing objects: 100% (12/12), done.
remote: Total 1948 (delta 23), reused 18 (delta 18), pack-reused 1918 (from 2)
Receiving objects: 100% (1948/1948), 2.34 MiB | 1.08 MiB/s, done.
Resolving deltas: 100% (281/281), done.
(base) qomolangma@LAPTOP-NVMISHDL:~$ cd app-helloworld
-bash: cd: app-helloworld: No such file or directory
(base) qomolangma@LAPTOP-NVMISHDL:~$ cd OS-2026/app-helloworld/
(base) qomolangma@LAPTOP-NVMISHDL:~/OS-2026/app-helloworld$ UK_DEFCONFIG=$(pwd)/defconfigs/qemu-arm64 make defconfig
make -j8
make[1]: Entering directory '/home/qomolangma/OS-2026/app-helloworld/workdir/unikraft'
  LN      Makefile
  MKDIR   lxdialog
  MAKE    kconfig
/usr/bin/gcc -ldl -I. -I/home/qomolangma/OS-2026/app-helloworld/workdir/build/kconfig -DCONFIG_=\"\"   -c fixdep.c -o
```

- 启动 QEMU 并开启 gdb 远程调试端口：

```
qemu-system-aarch64 -kernel workdir/build/helloworld_qemu-arm64 -nographic -
machine virt -cpu cortex-a57 -s -S
```

- 另开一个终端，使用 gdb-multiarch 连接 QEMU：

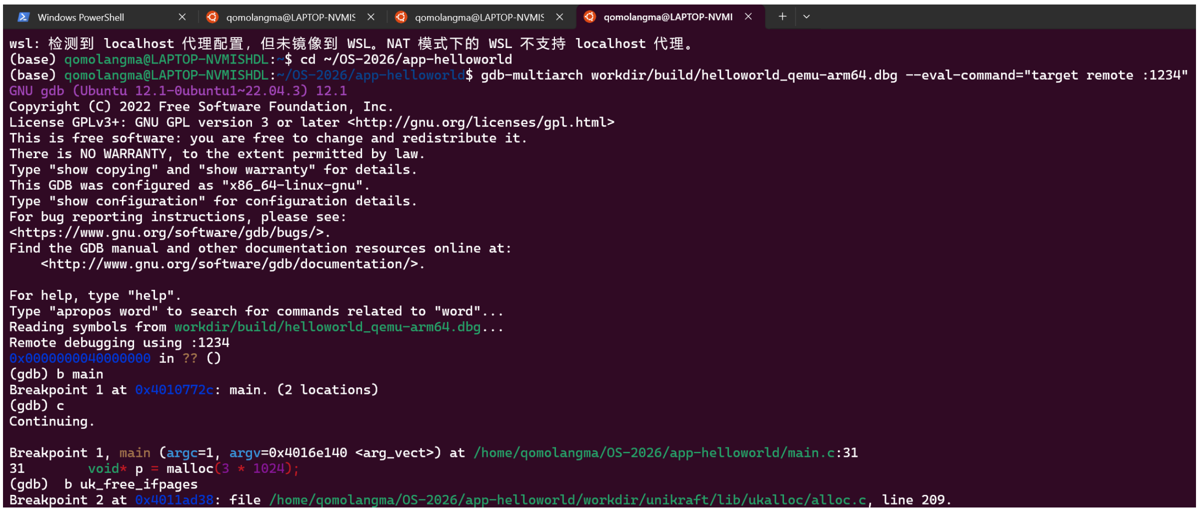
```
gdb-multiarch workdir/build/helloworld_qemu-arm64.dbg --eval-command="target
remote :1234"
```



```
wsl: 检测到 localhost 代理配置，但未镜像到 WSL。NAT 模式下的 WSL 不支持 localhost 代理。
(base) qomolangma@LAPTOP-NVMISHDL:~$ gdb-multiarch workdir/build/helloworld_qemu-arm64.dbg --eval-comma
t remote :1234"
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.3) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
workdir/build/helloworld_qemu-arm64.dbg: No such file or directory.
Remote debugging using :1234
warning: No executable has been specified and target does not support
determining executable automatically.  Try using the "file" command.
0x0000000040000000 in ?? ()
(gdb) 
```

```
(gdb) b main
(gdb) b uk_free_ifpages
```



```
wsl: 检测到 localhost 代理配置，但未镜像到 WSL。NAT 模式下的 WSL 不支持 localhost 代理。
(base) qomolangma@LAPTOP-NVMISHDL:~$ cd ~/OS-2026/app-helloworld
(base) qomolangma@LAPTOP-NVMISHDL:~/OS-2026/app-helloworld$ gdb-multiarch workdir/build/helloworld_qemu-arm64.dbg --eval-command="target remote :1234"
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.3) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
    <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from workdir/build/helloworld_qemu-arm64.dbg...
Remote debugging using :1234
0x0000000040000000 in ?? ()
(gdb) b main
Breakpoint 1 at 0x4010772c: main. (2 locations)
(gdb) c
Continuing.

Breakpoint 1, main (argc=1, argv=0x4016e140 <arg_vect>) at /home/qomolangma/OS-2026/app-helloworld/main.c:31
31          void* p = malloc(3 * 1024);
(gdb)  b uk_free_ifpages
Breakpoint 2 at 0x4011ad38: file /home/qomolangma/OS-2026/app-helloworld/workdir/unikraft/lib/ukalloc/alloc.c, line 209.
```

任务一：内存分配大小分析

请求分配大小	判定依据	分配路径	实际分配大小	分析与说明
96 B	96+32=128≤819	salloc	128	1×128B slab
128 B	128+32=160≤819	salloc	256	2×128B slab
256 B	256+32=288≤819	salloc	384	3×128B slab
4064 B	4064+32=4096>819	palloc	4096	1×4 KB page
4096 B	4096+32=4128>819	palloc	8192	2×4 KB pages

4. 任务二：核心问题逐条回答

Q1. 最小分配单元是什么？如何定义？

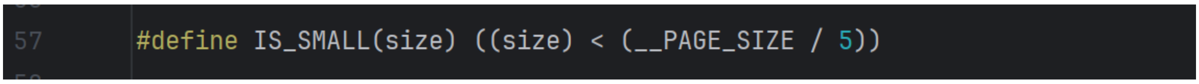
- **页分配（palloc）**：**4KB** 为最小页单位。
- **小块分配（salloc）**：**128B** 为最小小块单位；小块来自于事先从 buddy 借出的 **1 页**并切分为 32 份。

```
#define __S_PAGE_SHIFT 7
#define __S_PAGE_SIZE (1ULL << __S_PAGE_SHIFT)
```

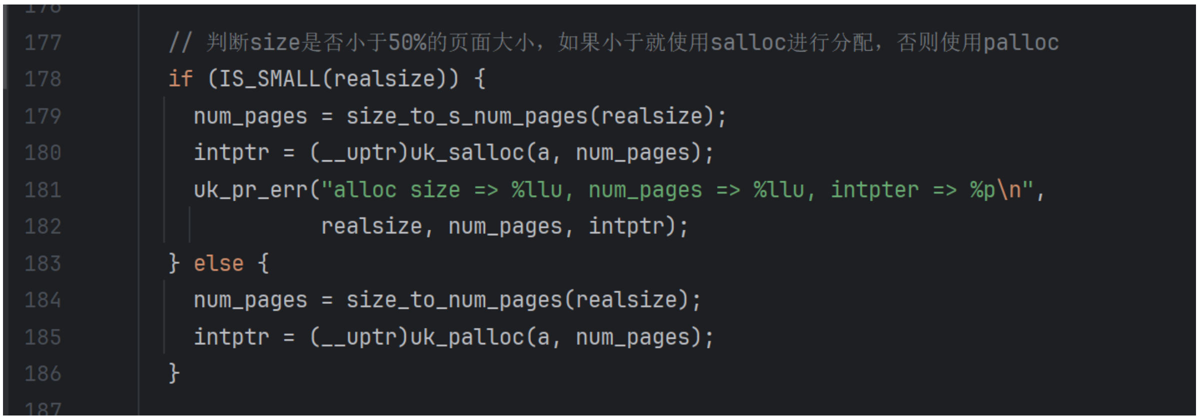
```
#define __S_PAGE_SHIFT 7
#define __S_PAGE_SIZE (1ULL << __S_PAGE_SHIFT)
```

Q2. `uk_malloc()` 何时选择 `palloc`，何时选择 `salloc`？

先将**申请字节 + 32 B 元数据**与 **4096/5=819 B** 比较：不超过则 **salloc**，否则 **palloc**；判断由 `IS_SMALL` 宏与 `alloc.c` 中的分流实现。



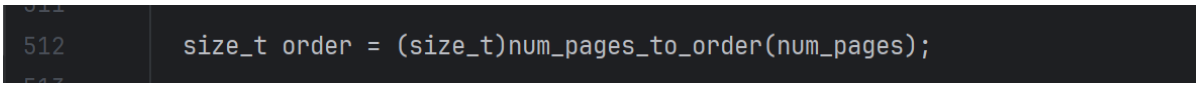
```
#define IS_SMALL(size) ((size) < (__PAGE_SIZE / 5))
```



```
    // 判断size是否小于50%的页面大小，如果小于就使用salloc进行分配，否则使用palloc
    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);
    }
```

Q3. 大内存分配的已知设计问题与 GDB 验证/修复

`free` 链路未正确传递 `small`，致使 `uk_get_metadata` 走错分支并向 `bbuddy_pfree` 传递错误 `num_pages`；在 GDB 中 `set var small=0` 可临时矫正流程并得到正确输出。

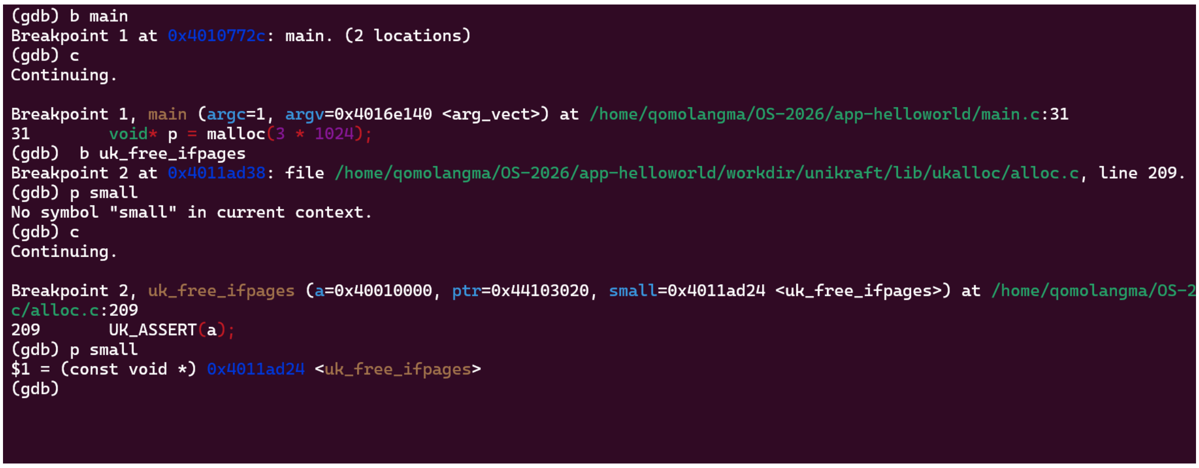


```
    size_t order = (size_t)num_pages_to_order(num_pages);
```

实验过程：

```
b uk_free_ifpages
```

```
p small
```



```
(gdb) b main
Breakpoint 1 at 0x4010772c: main. (2 locations)
(gdb) c
Continuing.

Breakpoint 1, main (argc=1, argv=0x4016e140 <arg_vect>) at /home/qomolangma/OS-2026/app-helloworld/main.c:31
31          void* p = malloc(3 * 1024);
(gdb)  b uk_free_ifpages
Breakpoint 2 at 0x4011ad38: file /home/qomolangma/OS-2026/app-helloworld/workdir/unikraft/lib/ukalloc/alloc.c, line 209.
(gdb) p small
No symbol "small" in current context.
(gdb) c
Continuing.

Breakpoint 2, uk_free_ifpages (a=0x40010000, ptr=0x44103020, small=0x4011ad24 <uk_free_ifpages>) at /home/qomolangma/OS-2
c/alloc.c:209
209         UK_ASSERT(a);
(gdb) p small
$1 = (const void *) 0x4011ad24 <uk_free_ifpages>
(gdb)
```

这里的small是一个未定义的值，用set将small修改为0：

```
set var small = 0
```

即运行成功。