武汉大学国家网络安全学院

2017—2018 学年第 二 学期

《 软件安全 》 考试试卷（A 卷，开卷）

学号 _____ 姓名 _____ 成绩 _____

请将答案写在答题纸上

一、简答题（每题 6 分，共 24 分）

1. 习近平主席在 2016 年 4 月 19 日讲话中指出“网络安全的本质在对抗，对抗的本质在攻防两端能力较量”，试结合本课程所学知识，浅谈对这句话的理解。
2. 一般 USB 采用 FAT32 文件格式，其存储的一个 JPG 照被无意删除，试给出手动恢复的原理和过程。
3. 简述 PE 结构中重定位节的结构，以及重定位的作用。
4. 简述 GS、DEP、ASLR 机制在对抗漏洞利用攻击时的作用机理。

二、计算题（每题 7 分，共 28 分）

1. 以下是某硬盘的分区表信息，计算所有分区大小。（按 1K=1000 计算，小数点后取 1 位，四舍五入，给出计算过程）

```
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 01
C1 FF 07 FE FF FF 3F 00  00 00 FC 8E 33 02 00 FE
FF FF 05 FE FF FF 80 29  54 02 80 29 54 02 00 00
00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00  00 00 00 00 00 00 55 AA
```

2. 下图为某 PE 文件的 16 进制数据（Windows XP 可正常运行），分析该数据，计算该程序载入内存中后 MessageBoxA 的函数地址存放位置（1），该程序的第一条指令位置（2），该程序运行之后，将弹出一个对话框，此时 0x4000B0—0x4000B3 位置四个字节的值（3）。给出简要的分析与计算过程。

```
          0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00000000h: 4D 5A 00 00 50 45 00 00 4C 01 01 00 B8 86 00 40 ; MZ..PE..L...筦.@
00000010h: 00 66 C7 00 6D 59 EB 64 28 00 02 00 0B 01 4D 65 ; .f?mY雅(.....Me
00000020h: 73 73 61 67 65 42 6F 78 41 00 00 00 0C 00 00 00 ; ssageBoxA.......
00000030h: 55 53 45 52 33 32 00 00 00 00 40 00 04 00 00 00 ; USER32....@.....
00000040h: 04 00 00 00 30 00 00 00 64 00 00 00 04 00 00 00 ; ....0...d.......
00000050h: 0C 00 00 00 50 EB 15 00 0C 00 00 00 00 00 00 00 ; ....P?.........
00000060h: 02 00 00 00 1C 00 00 00 00 00 00 00 53 C7 40 16 ; ............S芾.
00000070h: 3A 29 BB DB FF 50 CA C3 02 00 00 00 33 DB 53 50 ; :)蓍  P拭....3篮P
00000080h: 04 14 EB D0 38 00 00 00 6D 69 6E 69 45 58 45 2C ; ..胄8...miniEXE,
00000090h: 73 69 7A 65 3A 32 30 30 42 00 CE E4 B4 F3 D0 C5 ; size:200B.武大信
000000a0h: B0 B2 50 45 D7 F7 D2 B5 28 D0 D5 C3 FB 3A D3 DA ; 安PE作业(姓名:于
000000b0h: 00 00 00 00 2C D1 A7 BA C5 3A 32 30 31 31 33 30 ; ....,学号:201130
000000c0h: 32 35 33 30 30 37 38 29                         ; 2530078)
```

3. 下图为某 PE 程序的部分 16 进制数据截图，请分析该文件 data 节的具体信息(在文件及内存中的开始位置及大小)，并计算内存中 RVA 地址 0000B341H 在该 PE 文件中的文件偏移地址。[给出 16 进制数据]

```
00000140h: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ; ................
00000150h: 00 00 00 00 00 00 00 00 63 6F 64 65 00 00 00 00 ; ........code....
00000160h: 60 77 00 00 00 10 00 00 00 78 00 00 00 04 00 00 ; `w.......x......
00000170h: 00 00 00 00 00 00 00 00 00 00 00 00 20 00 00 60 ; ............ ..`
00000180h: 64 61 74 61 00 00 00 00 84 12 00 00 00 90 00 00 ; data....?...?.
00000190h: 00 06 00 00 00 7C 00 00 00 00 00 00 00 00 00 00 ; .....|..........
000001a0h: 00 00 00 00 40 00 00 C0 63 6F 6E 73 74 00 00 00 ; ....@..縿onst...
000001b0h: 60 2C 00 00 00 B0 00 00 00 2E 00 00 00 82 00 00 ; `,...?......?.
000001c0h: 00 00 00 00 00 00 00 00 00 00 00 00 40 00 00 40 ; ............@..@
000001d0h: 2E 72 73 72 63 00 00 00 D0 35 00 00 00 E0 00 00 ; .rsrc...?...?.
000001e0h: 00 36 00 00 00 B0 00 00 00 00 00 00 00 00 00 00 ; .6...?.........
000001f0h: 00 00 00 00 40 00 00 40 2E 69 64 61 74 61 00 00 ; ....@..@.idata..
```

4. 下图为某程序的.rdata 节（开始位置 RVA：2000，文件偏移量：800H）在内存中的主要数据。试分析计算 MessageBox 函数的真实地址, wsprintfA 函数的真实地址，该文件 800H-803H 偏移处的值，文件 808H-80BH 偏移处的值。(给出计算思路和结果)

地址	HEX 数据																ASCII
00402000	E2	BB	F8	76	00	00	00	00	11	EA	89	76	47	3F	85	76	忄 鴿....■陟vG?鄂
00402010	00	00	00	00	50	20	00	00	00	00	00	00	00	00	00	00	....P ----------
00402020	72	20	00	00	00	20	00	00	58	20	00	00	00	00	00	00	r --- --X ------
00402030	00	00	00	00	9A	20	00	00	08	20	00	00	00	00	00	00	....?--■ ------
00402040	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	----------------
00402050	64	20	00	00	00	00	00	00	8C	20	00	00	80	20	00	00	d ------?--■ --
00402060	00	00	00	00	80	00	45	78	69	74	50	72	6F	63	65	73	....■.ExitProces
00402070	73	00	6B	65	72	6E	65	6C	33	32	2E	64	6C	6C	00	00	s.kernel32.dll..
00402080	62	02	77	73	70	72	69	6E	74	66	41	00	9D	01	4D	65	b wsprintfA.?Me
00402090	73	73	61	67	65	42	6F	78	41	00	75	73	65	72	33	32	ssageBoxA.user32
004020A0	2E	64	6C	6C	00	00	00	00	00	00	00	00	00	00	00	00	.dll..........
004020B0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	----------------
004020C0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	----------------
004020D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	----------------
004020E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	----------------
004020F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	----------------

三、分析题（每题 8 分，共 24 分）

1. 以下代码为某病毒用于搜索 kernel32.dll 加载基地址的代码，试分析该段代码可能的失效场景并简要说明原因。

```
getK32Base:
dec     eax                         ;逐字节比较验证，速度比较慢，不过功能一样
mov     dx,word  ptr [eax+IMAGE_DOS_HEADER.e_lfanew]   ;就是 ecx+3ch
test    dx,0f000h                   ;Dos Header+stub 不可能太大,超过 4096byte
jnz     getK32Base                  ;加速检验，下一个
cmp     eax,dword ptr [eax+edx+IMAGE_NT_HEADERS.OptionalHeader.ImageBase]
jnz     getK32Base                  ;看 Image_Base 值是否等于 ecx 即模块起始值
mov     [ebp+k32Base],eax ;如果是,就认为找到 kernel32 的模块装入地址
lea     edi,[ebp+aGetModuleHandle] ;edi 指向 API 函数地址存放位置
lea     esi,[ebp+lpApiAddrs] ;esi 指向 API 函数名字串偏移地址（此地址需重定位）
```

2. 下面代码中存在漏洞，请分析其漏洞类型、 漏洞成因和利用方式

```
int main(int argc, char** argv) {
   printf(argv[1]);
   return 0;
}
```

3. 给出下面代码的输出结果；指出漏洞类型并分析该类型漏洞的可能利用方式

```
void main() {
      int sa = 0x7FFFFFFF;
      unsigned int ua = 0x7FFFFFFF;
      sa = sa + 2;
      ua = ua + 2;
      printf("sa: %d \t ua: %d\n", sa, ua);
      if (sa < 1)    printf("sa is overflow\n");
      if (ua < 1)    printf("ua is overflow\n");
}
```

四、综合题（每题 12 分，共 24 分）

1、设某程序采用双向链表维护其数据，用空节点表示链首（HEAD），请阅读以下代码，其中，edit 和 remove 函数为该程序定义的维护链表的函数，其先后调用顺序受用户输入的控制，分析该代码，指出该代码存在的漏洞和可能的利用方法。

```
struct node {
     struct node * flink; //前一块的指针;
     struct node * blink; //后一块的指针;
     char content[256];
     }
```

```
int edit (struct node * p_node) {
   char buf [400];
   read(0, &buf, 400);
   if (p_node) {
      strcpy(p_node->content, buf);
   }}
```

```
int remove (struct node * p_node) {
   if (p_node && pnode != HEAD) {
      p_node ->blink->flink = p_node ->flink;
      p_node ->flink->blink = p_node ->blink;
   }}
```

2、设某进程的运行时信息如下，rbp = 0x601800, EIP 指向的指令为 leave; ret; 进程当前的栈及相关内存布局如下图所示，

0x601800	0x0
	0x4005DE
	0x610000
	0x4005FE
	0x610010
	0x40060E
	0x0
	0x40061E
	0x3B
	0x400200
	\x00\x00\x00…

0x4005DE	pop rdi
	ret
0x4005FE	pop rsi
	ret
0x40060E	pop rdx
	ret
0x40061E	pop rax
	ret
0x400200	syscall

0x610000	“/bin/sh”\x00
0x610008	\x00\x00\x00\x00 \x00\x00\x00\x00
0x610010	0x610000
0x610018	\x00\x00\x00\x00 \x00\x00\x00\x00

题目说明：

1) Leave 指令的语义等价于 mov %rbp, %rsp; pop %rbp;
2) 对于 64 位程序，当参数传递小于 7 个时，参数的传递顺序为 rdi, rsi, rdx, rcx, r8, r9。
3) 0x3B 对应的系统调用为 sys_execve
4) sys_execve 函数原型为

```
sys_execve(const char __user *, filename,
           const char __user *const __user *, argv,
           const char __user *const __user *, envp);
```

其中，filename 指向要执行的程序，argv 为传递的参数，envp 为环境变量(通常为 0)。

请根据当前内存布局，分析进程的后续执行过程及执行结果。(12 分)