《软件安全》模拟题

一、选择题

1、有关渗透测试，说法错误的是

A、是通过模拟恶意黑客的攻击方法，来评估计算机网络系统安全的一种评估方法

B、是选择不影响业务系统正常运行的攻击方法而进行的测试

C、是一种不局限于发现软件或系统漏洞的测试

D、是一种以入侵目标系统、获取敏感信息为目的的攻击过程

2、Vulnerability 是指

A、漏洞利用　　B、漏洞挖掘

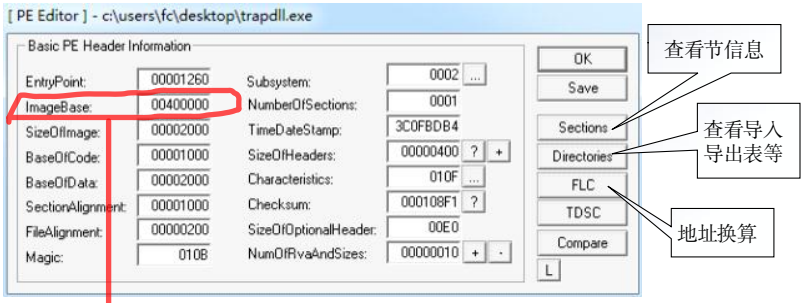
C、漏洞　　D、缺陷

3、ESP 是指

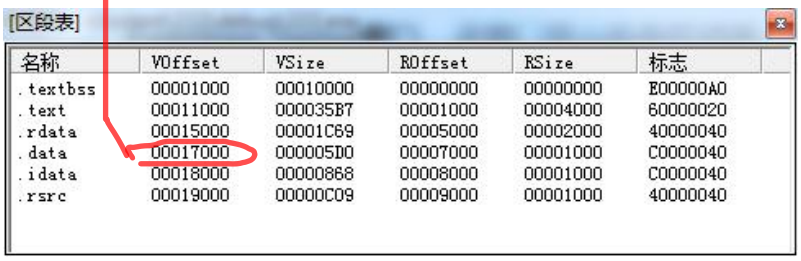
A、栈指针寄存器　　B、指令寄存器

C、基址指针寄存器　　D、数据寄存器

4、对于一个 PE 文件，使用 LordPE 查看基本信息如下：



点击 Sections 按钮，则可以查看节信息：



代码（.data 节）将被加载到的虚拟地址是

A、0x017000　　B、0x217000　　C、0x417000　　D、0x051700

5、CNNVD 是指

A、美国国家漏洞数据库　　B、通用漏洞列表

C、中国国家漏洞数据库　　D、国家互联网应急中心

6、函数 single_func 存在的漏洞类型为

```
void single_func(char *src)
{
    char buf[256];
    int i;
    for(i = 0;i <= 256;i++)
        buf[i] = src[i]; //拷贝 257 个字节到 256 个字节的缓冲区
}
```

A、单字节溢出漏洞　　B、缓冲区溢出漏洞

C、格式化字符串漏洞　　D、整数溢出漏洞

7、造成缓冲区溢出的根本原因是

A、程序功能性错误　　B、测试工作不足够

C、没有使用动态内存分配

D、不对数组边界条件和函数指针引用进行边界检查

8、栈帧地址的分配动态变化时，下列技术中，可以使新的返回地址定位到 shellcode 起始地址的是

A、Heap Spray　　B、slide code

C、NOP　　D、jmp esp

9、VC++编译器中提供的一种缓冲区溢出的检测防护技术是

A、DEP　　B、ASLR

C、GS Stack protection　　D、SafeSEH

10、不属于源代码检测技术的是

A、动态污点分析　　B、符号执行

C、污点传播分析　　D、数据流分析

11、窃取用户的 SessionID 后，使用该 SessionID 登录进入目标账户的攻击方法是

A、会话保持　　B、会话劫持

C、跨站脚本伪造攻击　　D、注入攻击

二、判断题

1、2017 年轰动全球的比特币勒索病毒是一种蠕虫(worm)病毒。(✓)

2、C 语言中，每个栈帧对应着一个未运行完的函数。(✓)

3、在很多字符串操作指令中，用 DS:EDI 指向源串，而 ES:ESI 指向目标串。(×)

4、在使用 OllyDBG 或者 IDA 进行逆向分析时候，所看到的汇编代码中的内存地址都是实际的物理内存地址。(×)

5、软件漏洞本身的存在没有危害，在通常情况下并不会对系统安全造成危害，只有被攻击者在一定条件下利用才会影响系统安全。(✓)

6、整数溢出一般不能被单独利用，而是用来绕过目标程序中的条件检测，进而实现其他攻击。(✓)

7、后渗透攻击指渗透进去目标之后的为了隐藏攻击行为而采取一系列措施。(✓)

8、Metasploit 是一个渗透测试工具，仅能支持多主机操作系统的扫描和渗透，比如 XP 操作系统等。(×)

9、HTTP 协议属于有状态的通信协议。(×)

10、Cookie 的信息可以用来保存登录用户的状态，常被用作会话管理，这些 Cookie 信息往往保存到服务器端。(×)

11、如果一个网站没有对上传的文件类型进行判断，将是非常危险的，容易被上传网页木马。(✓)

得分：

三、简答题

1、要使程序运行后显示 why u r here？请完成程序填空

```
#include <stdio.h>;
#include <stdlib.h>;
//Have we invoked this function?
void why_here(void)
{
    printf("why u r here?!\n");
    exit(0);
}
void f()
{
    int buff ;
    int value=0;
    int * p = &buff;
    _____= (int)why_here;
}
int main(int argc, char * argv[])
{
    f();
    return 0;
}
```

答案：___p[2] 或者 *(p+2)_____

2、请简述：

（1）模糊测试的步骤

确定测试对象和输入数据

生成模糊测试数据

检测模糊测试数据

监测程序异常

确定可利用性

答对每项可得 1 分，最多 4 分

（2）智能动态模糊测试和模糊测试的区别

引入基于符号执行、污点传播分析等可进行程序理解的方法

实现程序理解

有针对性的设计测试数据的生成

比传统的随机模糊测试更高的效率

采用智能技术分析输入数据和执行路径的关系

利用分析获得的输入数据集合，对执行路径集合进行测试

答对每项可得 1 分，最多 3 分

3、请简述两种跨站脚本攻击的名称及其利用方式

反射式 XSS 和 持久式 XSS　　(2 分)

反射式 XSS 将恶意脚本附加到 URL 地址的参数中

存储式 XSS 中的脚本是由 Web 应用程序进行存储的，并且会将其作为内容显示给浏览用户。

从攻击过程来说，反射式 XSS 一般要求攻击者诱使用户点击一个包含 XSS 代码的 URL 链接；

而存储式 XSS 则只需让用户查看一个正常的 URL 链接，而这个链接中存储了一段脚本

(3 分)

得分：

四、综合题

1、对于一个 Url：http://127.0.0.1/info.php，其脚本代码如下：

```
<?php
    $con=mysql_connect("localhost","root","lenovo");
    if(!$con){die(mysql_error());}
    mysql_select_db("products",$con);
    $sql="select * from category where id=$_POST[id]";
    echo $sql."<br>";
    $result=mysql_query($sql,$con);
    while($row=mysql_fetch_array($result,MYSQL_NUM))
    {
        echo $row[0]." ".$row[1]." ".$row[2] ."<br>";
    }
    mysql_free_result($result);
```

```
    mysql_close($con);
?>
```

（1）指出代码中采用的与 HTTP 服务器交互的方式是什么，并对两种主要交互方式的优缺点进行简单分析。

Post （2 分）

Get-暴露信息

Post-封装 安全性高一些（3 分）

（2）分析上述代码存在的漏洞，并简单举例说明该漏洞的利用方式。

SQL 注入（3 分

XSS 脚本攻击（2 分）

2、对于如下程序：

```
#include <iostream>
int sub(int x,int y)
{
    int z=0;
    z=x-y;
    return z;
}
void main()
{
    int n=0;
    n=sub(1,3);
    printf("%d\n",n);
}
```

请补全对应的汇编代码

① _____ EAX _____

②_____ EAX _____

```
#include <iostream>
int sub(int x,int y)
{
004113A0    push        ebp
004113A1    mov         ebp,esp
004113A3    sub         esp,0CCh
004113A9    push        ebx
004113AA    push        esi
004113AB    push        edi
004113AC    lea         edi,[ebp-0CCh]
004113B2    mov         ecx,33h
004113B7    mov         eax,0CCCCCCCCh
004113BC    rep stos    dword ptr es:[edi]
    int z=0;
004113BE    mov         dword ptr [z],0
    z=x-y;
004113C5    mov         eax,dword ptr [x]
004113C8    sub         eax,dword ptr [y]
004113CB    mov         dword ptr [z],eax
    return z;
004113CE    mov         _____①_____,dword ptr [z]
}
004113D1    pop         edi
004113D2    pop         esi
004113D3    pop         ebx
004113D4    mov         esp,ebp
004113D6    pop         ebp
004113D7    ret

void main()
{
    int n=0;
0041140E    mov         dword ptr [n],0
    n=sub(1,3);
```

```
00411415    push        3
00411417    push        1
00411419    call        sub (411096h)
0041141E    add         esp, 8
00411421    mov         dword ptr [n], _____②_____
    printf("%d\n",n);
}
```