# Bash Shellshock

CVE-2014-6271

1. 背景知识

Bourne Again Shell(简称BASH）是在GNU/Linux上最流行的SHELL实现，于1980年诞生，经过了几十年的进化从一个简单的终端命令行解释器演变成了和GNU系统深度整合的多功能接口。

法国GNU/Linux爱好者Stéphane Chazelas于2014年9月中旬发现了著名SHELL实现BASH的一个漏洞，你可以通过构造环境变量的值来执行你想要执行的脚本代码，据报道称，这个漏洞能影响众多的运行在GNU/Linux上的会跟BASH交互的应用程序，包括:

在sshd配置中使用了ForceCommand用以限制远程用户执行命令，这个漏洞可以绕过限制去执行任何命令。一些Git和Subversion部署环境的限制Shell也会出现类似情况，OpenSSH通常用法没有问题。

Apache服务器使用mod\_cgi或者mod\_cgid，如果CGI脚本在BASH或者运行在子SHELL里都会受影响。子Shell中使用C的system/popen，[Python](https://www.linuxidc.com/topicnews.aspx?tid=17" \o "Python" \t "https://www.linuxidc.com/Linux/2014-09/_blank)中使用 os.system/os.popen，PHP中使用system/exec(CGI模式)和Perl中使用，open/system的情况都会受此漏洞影响。

PHP脚本执行在mod\_php不会受影响。

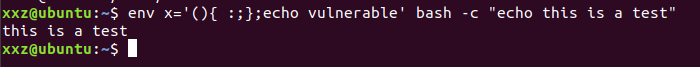
DHCP客户端调用shell脚本接收远程恶意服务器的环境变量参数值的情况会被此漏洞利用。

守护进程和SUID程序在环境变量设置的环境下执行SHELL脚本也可能受到影响。

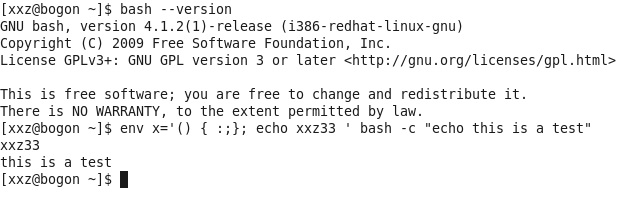
任何其他程序执行SHELL脚本时用BASH作为解释器都可能受影响。Shell脚本不导出的情况下不会受影响。

二、复现

首先要看bash版本当bash版本过高时会出现



当bash 1.14至bash 4.3的Linux/Unix系统存在该漏洞



三、分析与影响

漏洞的关键在于bash会把定义的各种函数放到env里面去，大致的格式如下：

$ function foo { echo bar; }

$ export -f foo

$ env | grep -A1 foo

foo=() { echo bar

}

这样的话，可以放一个假的函数进去，而且，这里的关键在于，这里它执行函数的时候，不是检测完整的“{}”对，而是简单的执行“{”之后的所有bash代码！也包括放进去的代码。

　　那么话说回来了，是不是不执行bash不就完事了？理论上是这样的。但是，事情总有意外，比如，你在编程的时候使用了systcall之类的，去运行了shell指令。你的程序继承了父程序的环境变量，你调用的systemcall空间继承了你程序的环境变量，然后，它就可能顺带的执行被注入到环境变量中的代码。