A look at CVE-2014-6271 (Shellshock)

Joseph Merrell-White
jam927

October 2014

Introduction
As is often the case when the mainstream media gets involved in an event taking place in the technology world, a lot of misinformation is propagated about said event. Contrary to NBC's reporting, the bug that has everyone's eye right now is not called Bash. Rather, the *software containing the bug* is called Bash. The bug itself has been named “Shellshock” (or less popularly, “Bashdoor”) by security experts. Its more official name is CVE-2014-6271, but the “Shellshock” moniker has expanded to cover a few related bugs[1].

Background
Bash is one of the most popular Unix shells and command processors currently in use. It has been around since the late eighties and is the default shell in Mac OSX and most currently maintained Linux distributions. The software is used to execute commands, either fed to it directly by a user or another program, or from a script. Most programs that execute external commands will feed the commands through bash to be processed, rather than executing them directly.

CVE-2014-6271 was first discovered by Stéphane Chazelas on the 12th of September this year. He contacted Chet Ramey, the current maintainer of Bash to disclose the flaw. An official patch was released on the 24th [1]. On the same date, the bug was given the CVE number and added to the National Institute of Standards and Technology's National Vulnerability Database and RedHat's and MITRE's Common Vulnerability and Exposure databases[5,6,7].

Shellshock
The bug itself is a coding error. Setting an environment variable prior to the execution of bash to contain a function declaration followed by commands causes these commands to be executed when bash is executed. Often, programs will set environment variables prior to executing a bash command or script in order to set data inside the command or script. If the data in these environment variables is provided by the user or some other external source, it becomes possible to exploit this to execute arbitrary commands using the program that calls bash.

One scenario in which this bug could be exploited involves certain web servers. CGI web servers that use bash to handle document requests are exploitable. By sending a request with a specially crafted user agent, it is possible to execute arbitrary commands on the server. Another involves DHCP clients that use bash. In a similar manner, it is possible to send a specially crafted DHCP response to a device trying to obtain an IP address, causing the device to execute arbitrary commands[9].

The easiest way to test for the presence of the original Shellshock bug would be to run the following command:

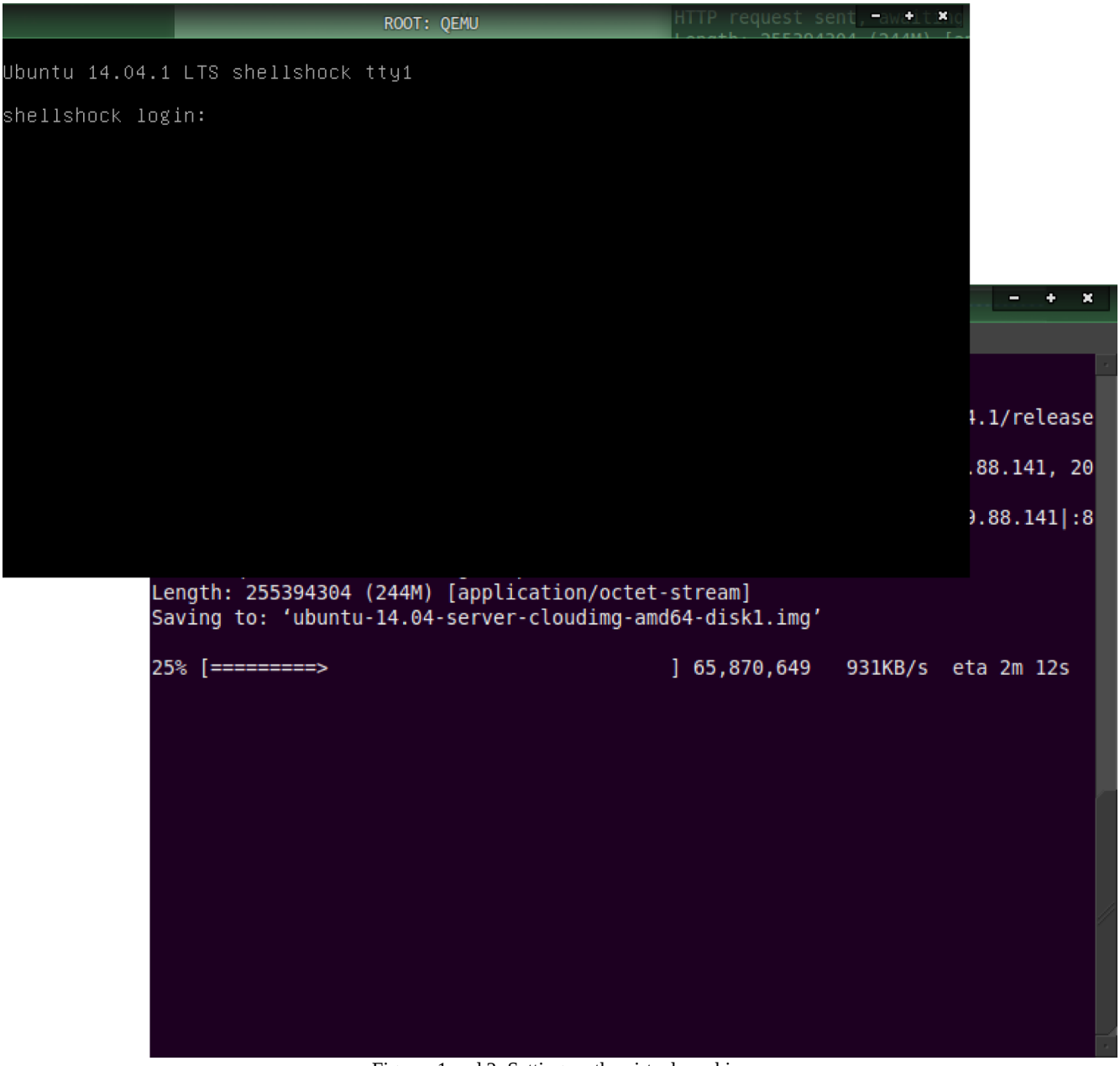
```
env x='() { :;}; echo vunlerable' bash -c “echo this is a test”
```

If the bug is present and the system is vulnerable, the command will display the word “vulnerable”. The first bit of the command, `env x='() { :;}; echo vunlerable'` sets the environment variable x. The name of the variable is not important to the exploitation of the bug. The variable is set to `() { :;}; echo vunlerable`, which is a set of bash commands in and of itself. The first command, `() { :;};`, is an empty function definition. The second command, `echo vunlerable`, is the arbitrary command executed by bash if the bug is present. This just prints the word “vulnerable”, but pretty much any valid command or set of commands will work here. The second half of our command, `bash -c “echo this is a test”`, simply runs bash and tells it to display “this is a test”.

The first official patch for the bug was released on September 24th [2]. Some downstream maintainers had already patched their builds of the program, however[8]. While this did fix CVE-2014-6271, other, similar vulnerabilities were found almost immediately. They were patched in a similar manner[3,4].

Demonstration

I have written a script to quickly construct a Qemu virtual machine running Ubuntu 14.04, using a server image created before the revelation of Shellshock. The script will automatically install apache on the virtual machine and put an exploitable cgi script in the `/var/www/html` directory.



Figures 1 and 2: Setting up the virtual machine

The virtual machine isn't completely finished setting up when the login prompt is presented. Sixty seconds past this point is usually enough time for apache to finish installing and configuring. The password is "`thispasswordissomewhatsecure`", but logging into the virtual machine is not necessary for the demonstration. The IP of the virtual machine can usually be retrieved by running the command "`arp -an`".

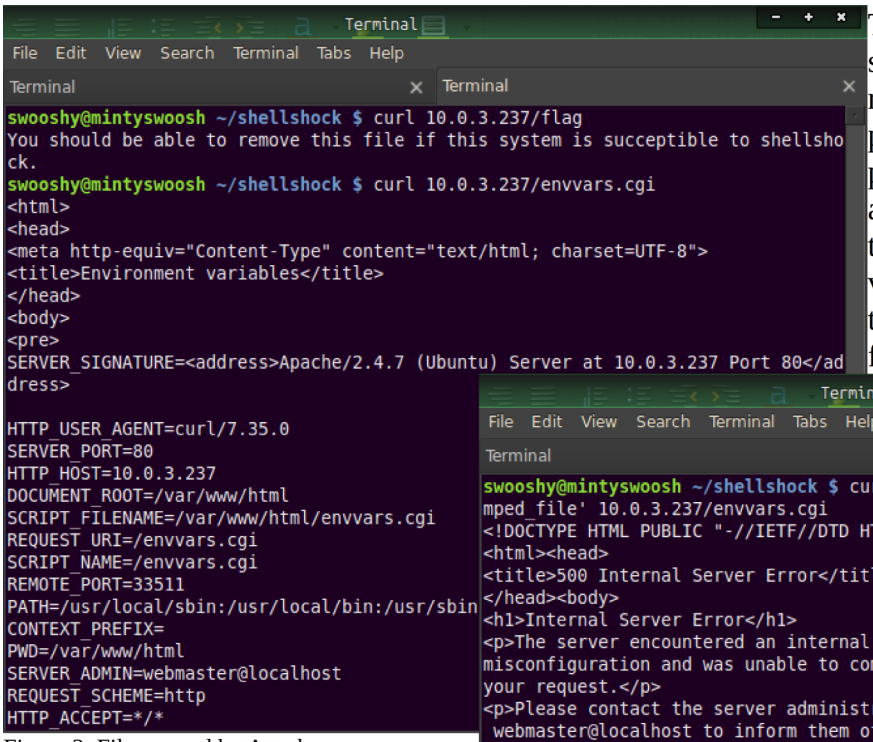


Figure 3: Files served by Apache

By changing the user agent, we are able to exploit the Shellshock vulnerability to copy a file not normally accessible via apache. Here, we copy `/etc/passwd` to `/var/www/html` and then we can view the file.

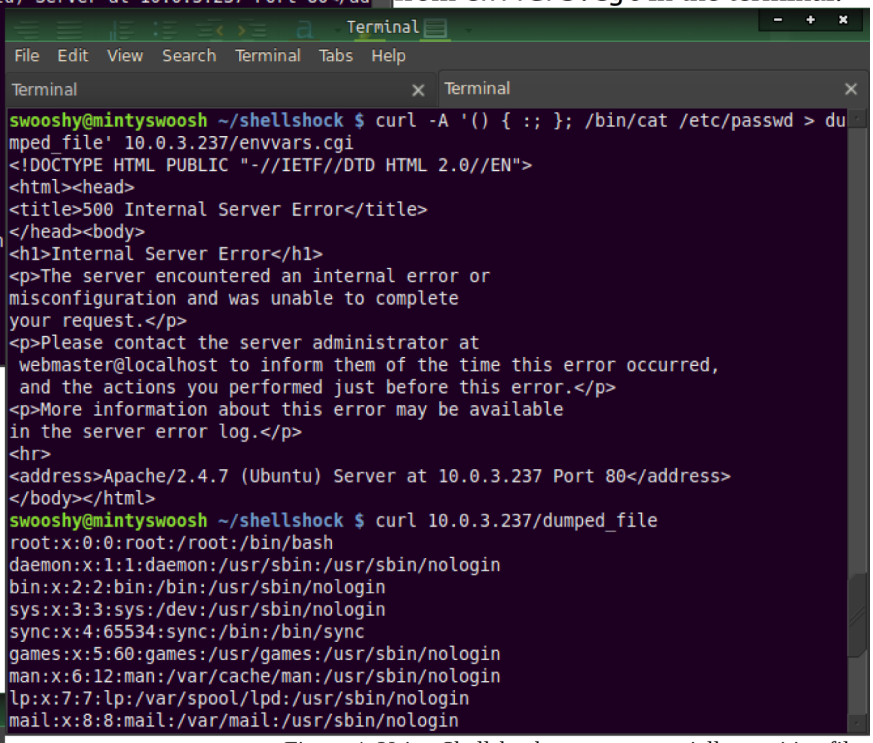


Figure 4: Using Shellshock to copy potentially sensitive files

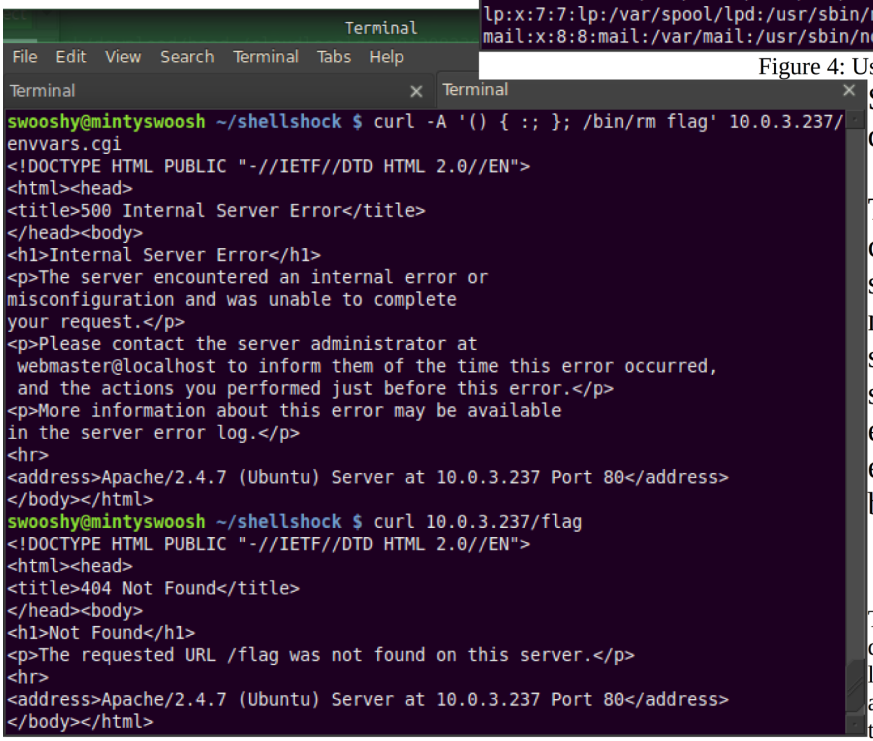


Figure 5: Deleting files using Shellshock

The Apache server runs on the standard port 80. Once it is up and running, a couple of files can be pulled from it. "`flags`" is simply a plain text file containing a string, and "`envvars.cgi`" is a cgi script that returns the environment variables. By using curl, we can see the string from flags and the output from `envvars.cgi` in the terminal.

Similarly, we can use Shellshock to delete files.

This exploit works here because the cgi script `envvars.cgi` is a bash script. The contents of the script do not matter much, aside from the shebang at the beginning. Apache sets the user agent to an environment variable before executing the script, triggering the bug.

The script to generate the virtual machine depends on Qemu 2.0 or greater, sed, wget, libvert, and Bash. An internet connection and about 300MB of disk space is also required, and the script must be run as root.

Conclusions

The widespread usage of bash means that a lot of machines and programs were susceptible to shellshock-related exploits. The bug's discovery was kept quiet until a patch could be released, which somewhat minimized the impact of the bug's presence. The bug can be rather easy to exploit once a vulnerable system has been identified, so keeping an up to date version of bash is crucial for server owners.

References

[1] Perlroth, N. (2014, September 25). Security Experts Expect ‘Shellshock’ Software Bug in Bash to Be Significant.

[2] Ramey, C. (2014, September 24). Bash-4.3 Official Patch 25.

[3] Ramey, C. (2014, September 24). Bash-4.3 Official Patch 27.

[4] Ramey, C. (2014, September 24). Bash-4.3 Official Patch 30.

[5] Red Hat. (2014, September 24). CVE-2014-6271.

[6] MITRE. (2014, September 24). CVE-2014-6271.

[7] NIST. (2014, September 24). CVE-2014-6271.

[8] Ubuntu Changelog for Bash. (2014, October 7).

[9] Walton, G. (2014, September 25). Shellshock DHCP RCE Proof of Concept.