SetUID ShellShock Attack Lab

Written by Carter Yagemann

Version 0.5

License

Objective

The objective of this lab is to teach students and cybersecurity professionals about the shellshock bash vulnerability by simulating a real world attack scenario. Upon completing this lab, you should have a better understanding on how shellshock can be applied to real world attacks and what environment conditions are early warning indicators of a possible shellshock vulnerability.

Setup

This lab requires one file and one program. The program should be like the one include with this lab (see `rootls.c`) and should belong to root and be set to mode 4755. The file should contain some secret text, belong to root, and be set to mode 400 so only root can read its contents.

Goal

The attacker's goal is to get the contents of the secret text file using only the program `rootls`, which performs directory listings as root.

Example Solution

The Attack

First, we export an environment variable containing our shellshock payload:

```
export foo='() { :; }; cat secret.txt'
```

This payload assumes that the secret file is called `secret.txt`. Next, we simply execute `rootls` and the contents of the secret file should be printed to the screen.

Detailed Explanation

In this attack, the vulnerability is that the program temporarily elevates its privilege to that of root and then performs a system command while acting as root. When it goes to run this command, a new bash session is created and that bash session sees the environment variable foo which looks like a function declaration. The bash session tries to parse this function and because of the shellshock vulnerability, it ends up executing the payload code as root.

Remediation

The problem here arises from an assumption the developer of the program made which is that only the code he passes into the system call will be executed. Due to the shellshock vulnerability, this assumption ends up being wrong. In general, developers should always be extra cautious when writing programs which use temporary privilege escalation and should never couple privilege escalation with the C system call.

Closing Remarks

This lab is currently in need of more refinement because it isn't yet clear which environments will produce the intended results. The lab has been ran against an Ubuntu 12.04 environment and an Arch Linux environment and it only worked correctly in one of the two environments. The reason for this is currently unknown.