PENETRATION TEST REPORT

DATE : 28/09/2023

MACHINE : 0DAY (Linux System)

ENUMERATION

+ Nmap

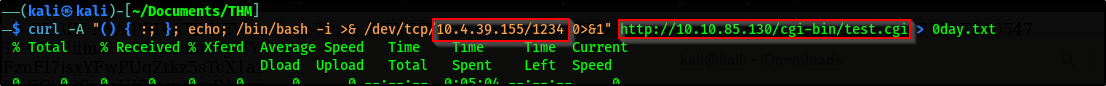
+ Nikto

EXPLOITATION

Before we start taking advantage of this vulnerability. Let’s dissect the above curl command. There are two parts to this command: with -A we’re specifying that we want to use a custom User-Agent. This is where we inject the payload, simply because the user-agent is a default header which we know will be read by the script. The second part of the command is standard — specifying the target: http://<MACHINE-IP>/cgi-bin/test.cgi. With that established, let’s break our payload down a little. There are technically four commands being run here:

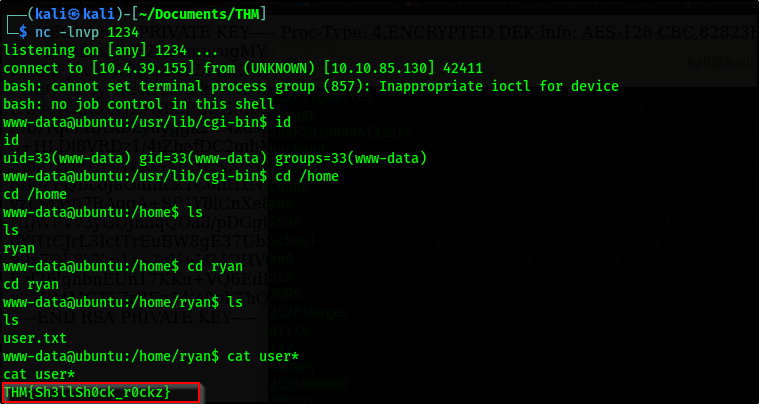
1. () { :;};
   * This defines an empty bash function. It’s there to activate the vulnerability, as shellshock relies on a function being declared prior to all subsequent commands being executed.
2. echo Content-Type: text/html; echo;
   * This section of the command is used to prevent the server from crashing when the vulnerability is exploited. A properly formatted HTTP response will contain a Content-Type header, and a blank line before the body of the response is displayed. Without these, the server will return a “500 Internal Server” error. Thus, we use this section of the payload to simulate the Content-Type header and subsequent blank line.
3. /bin/cat /etc/passwd;
   * Finally, the meat of the payload. These are the commands we actually want to be executed. In this instance this is merely outputting the contents of /etc/passwd; however, we could add any other commands we wished here.

Let’s use this vulnerability to obtain a reverse shell on our machine :



We ran our listener on our attacker machine and after some seconds we gain access

GAINING ACCESS



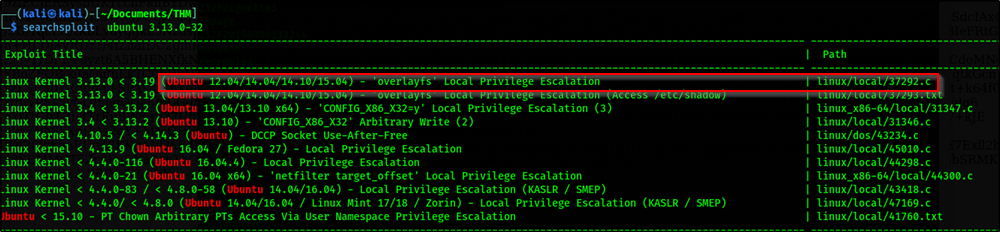
The first flag is highlighted

PRIVILEGE ESCALATION

First, we look at the version of the system and found: Ubuntu 3.13.0-13



Searching any known vulnerability or exploit with searchploit:

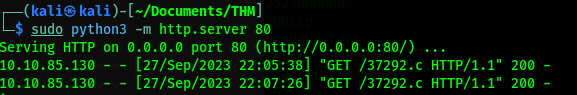


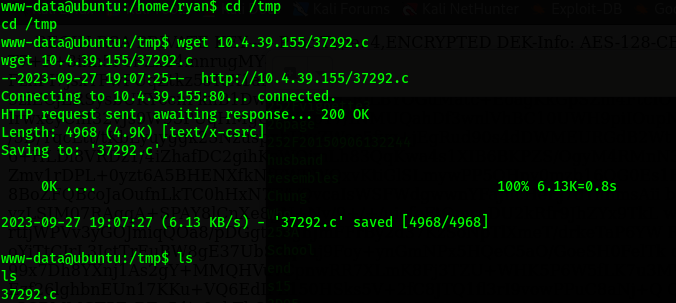
Found an exploit for Local privilege escalation: /Linux/local/37292.c

The next step is to send to exploit to the machine and execute it on it. So, let’s use python server.





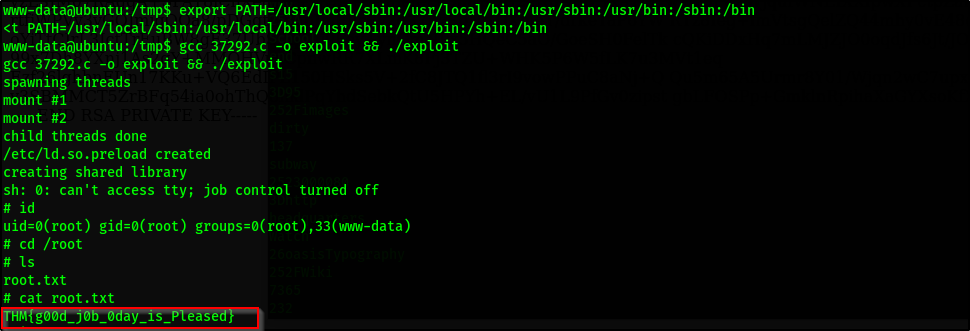




After we got the script we compile it and try to execute it:



An error message occurred and we fix it and get root access finally on the machine



Lessons

We’ve rooted the box, but there are some important lessons to be learnt from this room. For a start it serves as a reminder to never use outdated software — given that both exploits exploited here were as a result of the operating system being very outdated. This is not unrealistic, as many sysadmins find full server updates stressful, and the potential downtime serves as incentive not to attempt an upgrade. This, however, should be common knowledge in a community of hackers. What is talked about significantly less amongst the TryHackMe community is the concept of kernel exploits. Very few boxes on the site cover these — and they are extremely powerful: as demonstrated above. It is worth noting, however, that they also have the potential to be extremely disruptive to the server. Many kernel exploits result in a system crash if something goes wrong, or even after exploitation is completed and the resulting shell has died. Equally, kernel exploits tend to be extremely version specific, with kernel versions needing to match up pretty much identically in many cases. When it comes to Linux, we also need to factor in the plethora of distributions available — for a variety of reasons, a kernel exploit that works on the kernel of one distro, may well not work on the same kernel version used in a different distro

So, what’s the essential take-away from all this?

If you’ve learnt one thing here: kernel exploits are incredibly powerful, but should only be used as a method of last resort. Be cautious, be smart, and *always* review the source code for the exploit thoroughly before compiling and executing the code.

MACHINE : ANONYMOUS (Linux System)