Cyber@UC Meeting 92

Senior Designs and MBE crackmes

If You’re New!

- Join our Slack: **cyberatuc.slack.com**
- Check out our website: **cyberatuc.org**
- Organization Resources on our Wiki: wiki.cyberatuc.org
- **SIGN IN!** *(Slackbot will post the link in #general every Wed@6:30)*
- Feel free to get involved with one of our committees:

  Content *Finance* *Public Affairs* *Outreach* *Recruitment* *Lab*
- Ongoing work in our research lab!

Announcements

- Organization planning meeting Sunday, all are welcome to attend
- New Lab Head, Aaron Boyd
- Outdoor event, 27th near dabney
- Shirts and Hoodies, 25$ and 35$ respectively
- Battelle visit this Saturday
    - Pay attention to the slack for carpooling/details

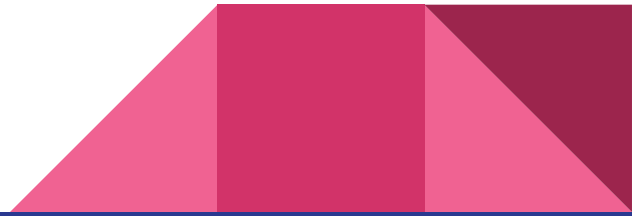

April 20th
CTF + MMORPG
11AM - 4PM
COLUMBUS, OH
BATTELLE

The Topics Today Go Something Exactly Like This

- Cyber@UC SOC
- Install GHIDRA if you haven’t already
- Walkthroughs and analysis for the first 6 MBE problems

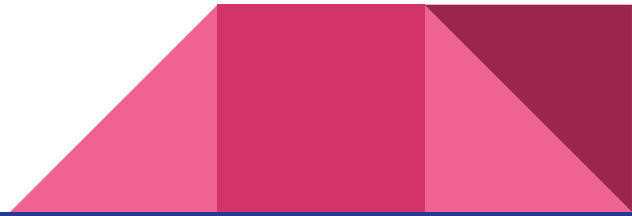
Cyber@UC SOC

Here we go...



SIG ALL IN ONE

Here We Go… But Better



Install GHIDRA

From their website:

ghidra-sre.org

From our gitlab:

gitlab.com/cyberatuc/ghidra

GHIDRA requires having JDK 11 as well.

Get the MBE problems

- https://github.com/RPISEC/MBE
- Their github has a link called “course website”
- Find “challenges.zip” from the course website
- Unzip and open in GHIDRA

crackme0x00a

```
                    pass.1685
0804a024 67 30 30       undefine...
         64 4a 30
         42 21 00
     0804a024 67            undefined167h          [0]
     0804a025 30            undefined130h          [1]
     0804a026 30            undefined130h          [2]
     0804a027 64            undefined164h          [3]
     0804a028 4a            undefined14Ah          [4]
     0804a029 30            undefined130h          [5]
     0804a02a 42            undefined142h          [6]
     0804a02b 21            undefined121h          [7]
     0804a02c 00            undefined100h          [8]
```

```
1
2  undefined4 main(void)
3
4  {
5    int iVar1;
6    int in_GS_OFFSET;
7    char local_2d [25];
8    int local_14;
9
10   local_14 = *(int *)(in_GS_OFFSET + 0x14);
11   while( true ) {
12     printf("Enter password: ");
13     __isoc99_scanf(&DAT_08048651,local_2d);
14     iVar1 = strcmp(pass.1685,local_2d);
15     if (iVar1 == 0) break;
16     puts("Wrong!");
17   }
18   puts("Congrats!");
19   if (local_14 != *(int *)(in_GS_OFFSET + 0x14)) {
20                     /* WARNING: Subroutine does not return */
21     __stack_chk_fail();
22   }
23   return 0;
24 }
```

Scanf (user input) a string and compare it to the bytes at 0x0804a024:

67 30 30 64 4A 30 42 21 = g00dJ0B!

crackme0x01

```
undefined4 main(void)

{
  int local_8;

  printf("IOLI Crackme Level 0x01\n");
  printf("Password: ");
  scanf("%d",&local_8);
  if (local_8 == 0x149a) {
    printf("Password OK :)\n");
  }
  else {
    printf("Invalid Password!\n");
  }
  return 0;
}
```

Scanf user input into local_8 as a decimal
Compare local_8 to 0x149a
We can use python to pipe our input as decimal in

```
python3 -c "print(int('149a',base=16))" | ./crackme0x01
```

crackme0x03

```
undefined4 main(void)

{
  undefined4 local_8;

  printf("IOLI Crackme Level 0x03\n");
  printf("Password: ");
  scanf("%d",&local_8);
  test(local_8,0x52b24);
  return 0;
}
```

```
void test(int param_1,int param_2)

{
  if (param_1 == param_2) {
    shift("Sdvvzrug#RN$$$#=,");
  }
  else {
    shift("Lqydolg#Sdvvzrug$");
  }
  return;
}
```

```
void shift(char *input_str)

{
  size_t str_len;
  uint index;
  char built_string [120];

  index = 0;
  while( true ) {
    str_len = strlen(input_str);
    if (str_len <= index) break;
    built_string[index] = input_str[index] + -3;
    index = index + 1;
  }
  built_string[index] = 0;
  printf("%s\n",built_string);
  return;
}
```

rot3

Similar scanf and comparison although now we have a custom function test.

Going into test shows we pass it two parameters and do an simple comparison then deobfuscate a corresponding result string through the shift function.

```
python3 -c "print(int('52b24',base=16))" | ./crackme0x03
```

crackmex04

```
undefined4 main(void)

{
  undefined local_7c [120];

  printf("IOLI Crackme Level 0x04\n");
  printf("Password: ");
  scanf("%s",local_7c);
  check(local_7c);
  return 0;
}
```

```
void check(char *user_input)

{
  size_t string_length;
  char current_char;
  uint index;
  int counter_1;
  int current_int;

  counter_1 = 0;
  index = 0;
  while( true ) {
    string_length = strlen(user_input);
    if (string_length <= index) {
      printf("Password Incorrect!\n");
      return;
    }
    current_char = user_input[index];
    sscanf(&current_char,"%d",&current_int);
    counter_1 = counter_1 + current_int;
    if (counter_1 == 0xf) break;
    index = index + 1;
  }
  printf("Password OK!\n");
                    /* WARNING: Subroutine doe
  exit(0);
}
```

Similar to the last one, we have a custom check function to validate the password.

We have a counter that increments from the characters in our input as integers, then if we reach 0xf (16) before the end of the string, our password is valid

crackmex05

```
undefined4 main(void)

{
  undefined local_7c [120];

  printf("IOLI Crackme Level 0x05\n");
  printf("Password: ");
  scanf("%s",local_7c);
  check(local_7c);
  return 0;
}
```

```
void check(char *input)

{
  size_t sVar1;
  char chr;
  uint count;
  int sum;
  int charValue;

  sum = 0;
  count = 0;
  while( true ) {
    sVar1 = strlen(input);
    if (sVar1 <= count) break;
    chr = input[count];
    sscanf(&chr,"%d",&charValue);
    sum = sum + charValue;
    if (sum == 0x10) {
      parell(input);
    }
    count = count + 1;
  }
  printf("Password Incorrect!\n");
  return;
}
```

```
void parell(char *param_1)

{
  uint local_8;

  sscanf(param_1,"%d",&local_8);
  if ((local_8 & 1) == 0) {
    printf("Password OK!\n");
                    /* WARNING:
    exit(0);
  }
  return;
}
```

```
1001 = 9   1000 = 8
0001 &     0001 &
0001 = 1   0000 = 0
```