Category: Rev

Challenge: Lost Keys

Writeup:

This is an 64 bit elf executable. Is I usually do, I run the binwalk command. It didn't show any packing. I can check it manually with hex editor or with other stuff, but there is no packing. There is no valuable data obtained with strings command. So, I decided to decompile with ghidra.

main function is like following

```
undefined8 main(void)
{
  int iVar1;
  char *__s1;
  long in_FS_OFFSET;
  undefined local_78 [104];
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  printf("I lost my keys. Can you find it?: ");
  __isoc99_scanf(&DAT_0010204b,local_78);
  __s1 = (char *)turn(local_78,"asdfjhgvbdjkgfhnjksdffgdfjh","asdfjhgvbdjkgfhnjksdffgdfjh");
  iVar1 = strcmp(__s1,"WWJ$]MJ|IJQ_A[\\cUc|V@VAI\"DM");
  if (iVar1 == 0) {
    puts("CONGRATS ");
  }
  else {
    puts("NOPE ");
  }
  if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return 0;
}
```

basically, it takes one string from us. processes it in the turn function. Returns the processed string. if that string equals "WWJ$]MJ|IJQ_A[\\cUc|V@VAI\"DM" it shows congrats message. So lets analyze the turn function:

```
undefined1 * turn(long param_1,long param_2)
{
  long in_FS_OFFSET;
  int local_2c;
  int local_28;
  int local_24;
  int local_20;
  char local_16 [6];
  long local_10;

  local_10 = *(long *)(in_FS_OFFSET + 0x28);
  local_2c = 0;
  local_24 = 0;
  local_28 = 0;
  while (*(char *)(param_1 + local_28) != '\0') {
    local_24 = local_24 + 1;
    local_28 = local_28 + 1;
  }
  local_28 = 0;
  while (local_28 < ((local_24 / 3) * 3) / 3) {
    local_16[3] = *(undefined *)(param_2 + local_28 * 3);
    local_16[4] = *(undefined *)(param_2 + (long)(local_28 * 3) + 1);
    local_16[5] = *(undefined *)(param_2 + (long)(local_28 * 3) + 2);
    local_16[0] = *(char *)(param_1 + (long)(local_28 * 3) + 2);
    local_16[1] = *(undefined *)(param_1 + (long)(local_28 * 3) + 1);
    local_16[2] = *(undefined *)(param_1 + local_28 * 3);
    local_20 = 0;
    while (local_20 < 3) {
      turned.2497[local_2c] = (local_16[(long)local_20 + 3] - local_16[local_20])
 + '9';
      local_2c = local_2c + 1;
      local_20 = local_20 + 1;
    }
    local_28 = local_28 + 1;
  }
  if (local_10 != *(long *)(in_FS_OFFSET + 0x28)) {
                    /* WARNING: Subroutine does not return */
    __stack_chk_fail();
  }
  return turned.2497;
}
```

- in turn function it takes two parameters.
- splits the first parameter 3 by 3.
- shuffles these parts like 1 to 3 and 3 to 1.
- subtracts the ASCII value of the characters of shuffled string from second parameter string's characters and adds ascii '9' (57)
- returns the resulting string.

let's reverse this process with a python code. Note that resulting string is "WWJ$]MJ|IJQ_A[\\cUc|V@VAI\"DM" and second parameter string is "asdfjhgvbdjkgfhnjksdffgdfjh"

```python
a = "asdfjhgvbdjkgfhnjksdffgdfjh"
b = "WWJ$]MJ|IJQ_A[\\cUc|V@VAI\"DM"

flag_p1 = ""

for i in range(len(a)):
    c = ord(a[i]) - ord(b[i]) + 57
    c = chr(c)
    flag_p1 += c


flag_p2 = ""
for i in range(len(flag_p1)//3):
    s = flag_p1[i*3:i*3+3]
    s = s[::-1]
    flag_p2 += s

print(flag_p2) # output = SUCTF{R3VERSED_AND_G0T_IT_}
```