Final_Assignment_Rough_Notes

Rough Notes

Static Analysis

Linux Analysis.

- `file softwarechallenge.exe` shoes me that the file is PE32 executable that is stripped.
- `strings softwarechallenge.exe` shows me that I can see some intersting things
  - I see that I have to figure out a password and I see the corresponding strings. I see that there is a `Result" %d` So I know that it is printing somehting.
  - I see a run time error.
  - I see that there some windows commands that are calling api to find some files

Ghidra Analysis

- 32 bit
- I found that the result prints out 0x539
- I used a online c compiler so that I can see that the result is 1337 when printed out.
- `FindNextFileA` continues from find first file
- `fgets()` is used to get password from the user
- I am maybe seeing some sort of encryption that needs to be solved with the password I type in and the password length.
- I will need to create a python script to decrypt my findings.
- in the `string_for_creation_function` I see that rand() is used to create a string. This means there might not be a static password.
  - **We don't get a random number because he sets the seed so the values can not change. This means there is a static password.
  - I do how ever have to use a debugger. I will want to see what the random values are at each step. I can then create a python script to try and get these numbers.

Code Figures

```
// Online C compiler to run C program online
#include <stdio.h>
#include <stdlib.h>

int main() {
    void *encrypted_string_result;
```

```
    int seeded_integer;
    int i;
    srand(1700000000);
    encrypted_string_result = malloc(9);
    for(i = 0; i < 8; i + 1) {
        seeded_integer = rand();
        printf("Seeded Character (decimal): %d " , seeded_integer);
        printf("NEWLINE\n");
    }
```

```
buffer_1 = [59, 25, 89, 18, 63, 60, 32, 28, 95, 74, 11, 0]
encrypted_string_in_decimal = [107, 120, 42, 97, 72, 83, 82, 120, 107, 120,
42, 97]

# Convert buffer_1 to binary with leading zeros and join them
buffer_1_binary = ' '.join(format(number, '08b') for number in buffer_1)

# Convert encrypted_string_in_decimal to binary with leading zeros and join
them
encrypted_string_in_binary = ' '.join(format(number, '08b') for number in
encrypted_string_in_decimal)

print( buffer_1_binary)
print(encrypted_string_in_binary)
```

Dynamic Analysis

windows

- when I run the program I get asked for the password.
- When I type in 1337 I get it wrong. I think I saw mentions of encrptyion so I will need to step through the program.
-

Ida free debugger

- I am going to use the debugger to generate the seeded numbers so they are more accurate than what I was getting in the online compiler.

Key Ideas

- The program has to be ran to get the seeded numbers (Not sure if these can be found statically).
- There is a set of encryption algorithms to try and change up the password.

    - there seems to be multiple instances I need to crack.
    - The first instance I was able find a seeded string used in the 2nd outer encryption algorithm.
    - It seems that the second encryption algorithm uses a boolean to check if memcmp is true. this is what I need to pass. `memcmp_value = memcmp(local_114,&local_120,0xc);`
    - we want memcmp to be zero which means the memory contents are equal.
- **ida code is:** kx*aHSRx
- **passcode is: Passwrod
- I got stuck because I was not paying enough close attention. When the strlen has no space it can't work. Its good to use my intuition.
- I had to use ascii table from online.
- I used a binary to decimal from online
- I did the bit level operation by hand to come up with the passcode
-

Questions