CS6570: Secure Systems Engineering
Assignment-5: Attack Phase

Team: Trojan
Members: Pradeep Peter Murmu (CS24M033), Vishnu K. (CS24M022)

Binary 91

AES Key:
['0xa0', '0x51', '0x38', '0xb9', '0x44', '0xfc', '0x3a', '0xa0',
'0x08', '0xfc', '0xce', '0xf7', '0xa4', '0x21', '0x42', '0xb0']

> Steps

key_obfuscated = [
0xe1, 0x10, 0x79, 0xf8, 0x05, 0xbd, 0x7b, 0xe1,
0x49, 0xbd, 0x8f, 0xb6, 0xe5, 0x60, 0x03, 0xf1
]

xor= 0x41

In the main function:

```
00404b06
00404b06    for (int32_t i = 0; i s<= 7; i += 1)
00404af4        *(&var_38 + sx.q(i)) = *(sx.q(i) + &obfuscated_key_part1) ^ 0x41
00404af4
00404b47    for (int32_t i_1 = 0; i_1 s<= 7; i_1 += 1)
00404b35        *(&var_38 + sx.q(i_1 + 8)) = *(sx.q(i_1) + &obfuscated_key_part2) ^ 0x41
00404b35
00404b50    int64_t result
00404b50
```

```
00498004                00 00 00 00-00 00 00 00 00 00 00 00      ............
00498010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................

00498020  uint64_t obfuscated_key_part1 = 0xe17bbd05f87910e1
00498028  uint64_t obfuscated_key_part2 = 0xf10360e5b68fbd49

00498030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00  ................
```

binary- 24

AES key = {0x03, 0xdf, 0xcd, 0x4b, 0x71, 0x83, 0xe1, 0x21, 0x89, 0xba, 0xc1, 0x6a, 0x6a, 0x23, 0x2e, 0x35}

Egg_params = { {2,1,0,0,0,0},
{3,3,1,3,0,3},
{5,2,0,2,2,0},
{6,4,0,3,1,0},
{9,1,3,3,1,2} }

Compute_gf : return 11 * eggs[1] - 68 * eggs[3] - eggs[4];

> Steps

AES key is stored at data_499080

```
00401620    int64_t main(int32_t arg1, void* arg2)

00401634        void* fsbase
00401634        int64_t rdx = *(fsbase + 0x28)
00401647        int128_t var_38 = data_499080
00401652        int64_t result
00401652
```

Going to data_499080, we get the key stored there.

```
00499080  int128_t data_499080 =
00499080  03 df cd 4b 71 83 e1 21-89 ba c1 6a 6a 23 2e 35  ...Kq..!...jj#.5
```

Egg_params is stored here:

```
004c6220  egg_params:
004c6220  02 01 00 00 00 00 03 03-01 03 00 03 05 02 00 02  ................
004c6230  02 00 06 04 00 03 01 00-09 01 03 03 01 02 00 00  ................
```

```
char var_106_1 = 0
char var_105_1 = 0
global_flag = eggs:1.b * 0xb + (neg.d(zx.d(eggs:4.b))).b - eggs:3.b * 0x44
print_bytes.constprop.0("Ciphertext:", &s)
eggs.b
```

Binary - 77

AES key - { 0xb2, 0x4f, 0x49, 0x6c, 0x75, 0xa4, 0x57, 0x2b, 0xc4, 0x0f, 0x34, 0x53, 0xd5, 0x00, 0x81, 0xa3}

> Steps:

The key was stored plain in .rodata

```
00406120  int128_t data_406120 =
00406120  b2 4f 49 6c 75 a4 57 2b-c4 0f 34 53 d5 00 81 a3  .OIlu.W+..4S....
00406130  int128_t data_406130 =
```

Binary 5

AES key= {
0x71, 0x9b, 0xf8, 0x97, 0x3d, 0x0c, 0xa9, 0x03,
0xdd, 0xbd, 0xc9, 0xaf, 0xd4, 0x82, 0xb9, 0x91}

>Steps:

Checking the .rodata, the key was stored plain.

```
00407140  int128_t data_407140 =
00407140  71 9b f8 97 3d 0c a9 03-dd bd c9 af d4 82 b9 91  q...=...........
00407150  int128_t data_407150 =
```

Binary 99

AES key= {
0x2F, 0x23, 0xAB, 0x16,

0x99, 0x73, 0x9F, 0xD8,
0x19, 0x75, 0xAF, 0x51,
0x4F, 0xEF, 0x1E, 0x7B}

>Steps:

First the obfuscated key were stored like this.

```
00402520    int32_t main(int32_t argc, char** argv, char** envp)

0040254c        int64_t var_d8 = -0x23638a67e557dad2
00402553        int64_t var_d0 = 0x7f1bf14e55ac7718
00402553
```

The obfuscated key was passed to sub_401d1b() function

```
0040264e        void var_c8
0040264e        sub_401d1b(&var_c8, &var_d8)
00402667        sub_401845(&var_d8, 0x10)
```

The function further calls another function.

```
00401d1b    int64_t sub_401d1b(void* arg1, void* arg2)

00401d39        void var_18
00401d39        sub_40133b(arg2, &var_18)
00401d53        return sub_40148d(arg1, &var_18)
```

The obfuscated key is converted to original key in this function:

```
0040133b    void sub_40133b(void* arg1, int64_t arg2)

0040144e        for (int32_t i = 0; i s<= 0xf; i += 1)
0040136a            char var_d_1 = 0
00401372            char var_f = *(arg1 + sx.q(i))
0040137d            uint32_t rax_8 = i s>> 0x1f u>> 0x1e
00401385            int32_t rdx_5 = ((i + rax_8) & 3) - rax_8
00401385
0040138c            if (rdx_5 == 3)
0040142f                var_d_1 = sub_401196(&data_4031a0, 4, &var_f, 0)
0040138c            else if (rdx_5 == 2)
0040140f                var_d_1 = sub_401196(&data_403190, 4, &var_f, 0)
0040139e            else if (rdx_5 == 0)
004013cf                var_d_1 = sub_401196(&data_403170, 4, &var_f, 0)
004013ab            else if (rdx_5 == 1)
004013ef                var_d_1 = sub_401196(&data_403180, 4, &var_f, 0)
004013ef
00401444            *(sx.q(i) + arg2) = var_d_1
```

Reversing the function we get the original key.

#Binary 63

aes_key = {

```
      0x5f, 0x60, 0x7c, 0xce,
      0xdf, 0x8e, 0x26, 0x56,
      0xf5, 0x62, 0x94, 0xb4,
      0x7a, 0x24, 0x3b, 0xb6}
egg_params = {
   {1, 1, 0, 3, 3, 3},
   {2, 4, 3, 1, 1, 2},
   {5, 1, 1, 0, 1, 3},
   {6, 3, 2, 3, 0, 0},
   {8, 4, 3, 1, 2, 3}};

compute gf = return eggs[1] + eggs[2] * 0x22 + eggs[4] * 0x13;
```

> Steps

The person probably didn't obfuscate. Everything was obvious

```
004015b7    int64_t main(int32_t arg1, void* arg2)

004015cf        void* fsbase
004015cf        int64_t rax = *(fsbase + 0x28)
004015de        char var_38 = 0x5f
004015e2        char var_37 = 0x60
004015e6        char var_36 = 0x7c
004015ea        char var_35 = 0xce
004015ee        char var_34 = 0xdf
004015f2        char var_33 = 0x8e
004015f6        char var_32 = 0x26
004015fa        char var_31 = 0x56
004015fe        char var_30 = 0xf5
00401602        char var_2f = 0x62
00401606        char var_2e = 0x94
0040160a        char var_2d = 0xb4
0040160e        char var_2c = 0x7a
00401612        char var_2b = 0x24
00401616        char var_2a = 0x3b
0040161a        char var_29 = 0xb6
00401625        int64_t result
```

```
00401149    uint64_t compute_gf(void* arg1)

0040118d        return zx.q(zx.d(*(arg1 + 1)) + zx.d(*(arg1 + 2)) * 0x22 + zx.d(*(arg1 + 4)) * 0x13)
```

```
006cb090  egg_params:
006cb090  01 01 00 03 03 03 02 04-03 01 01 02 05 01 01 00  ...
006cb0a0  01 03 06 03 02 03 00 00-08 04 03 01 02 03 00 00  ...
```