jamk.fi

Reverse Engineering TTC6510-3002

Joonatan Ovaska

A K M MAHMUDUL HAQUE
AB0208

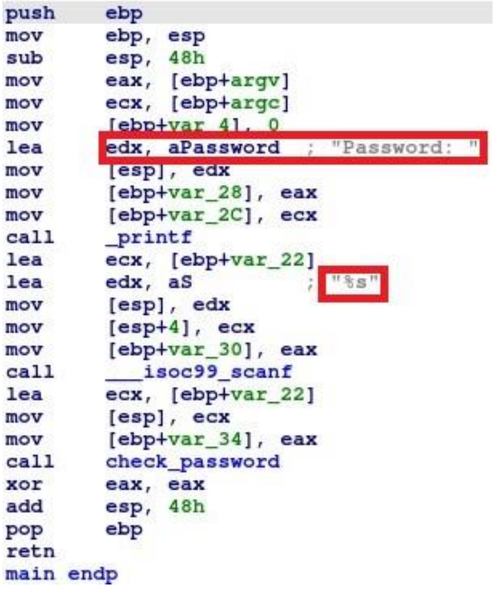
Student number: 2110841

Lab03

Date: 13.09.2023

Jyväskylän ammattikorkeakoulu
JAMK University of Applied Sciences

First Step

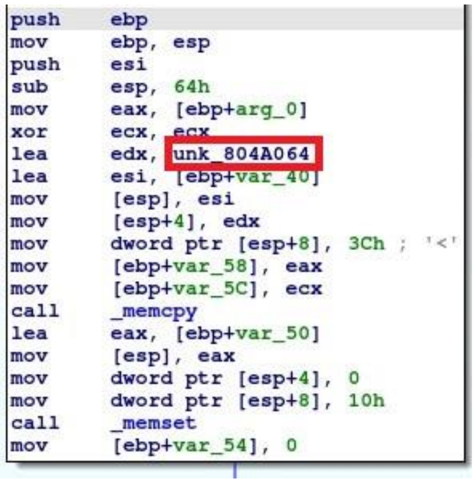


```
push    ebp
mov     ebp, esp
sub     esp, 48h
mov     eax, [ebp+argv]
mov     ecx, [ebp+argc]
mov     [ebp+var_41], 0
lea     edx, aPassword  ; "Password: "
mov     [esp], edx
mov     [ebp+var_28], eax
mov     [ebp+var_2C], ecx
call    _printf
lea     ecx, [ebp+var_22]
lea     edx, aS           ; "%s"
mov     [esp], edx
mov     [esp+4], ecx
mov     [ebp+var_30], eax
call    ___isoc99_scanf
lea     ecx, [ebp+var_22]
mov     [esp], ecx
mov     [ebp+var_34], eax
call    check_password
xor     eax, eax
add     esp, 48h
pop     ebp
retn
main endp
```

- In the **main** function **"aPassword"** with the comment on its side (**"Password"**) caught my attention at the first glance.

- The **%s** also gives an idea that the password is a string value.

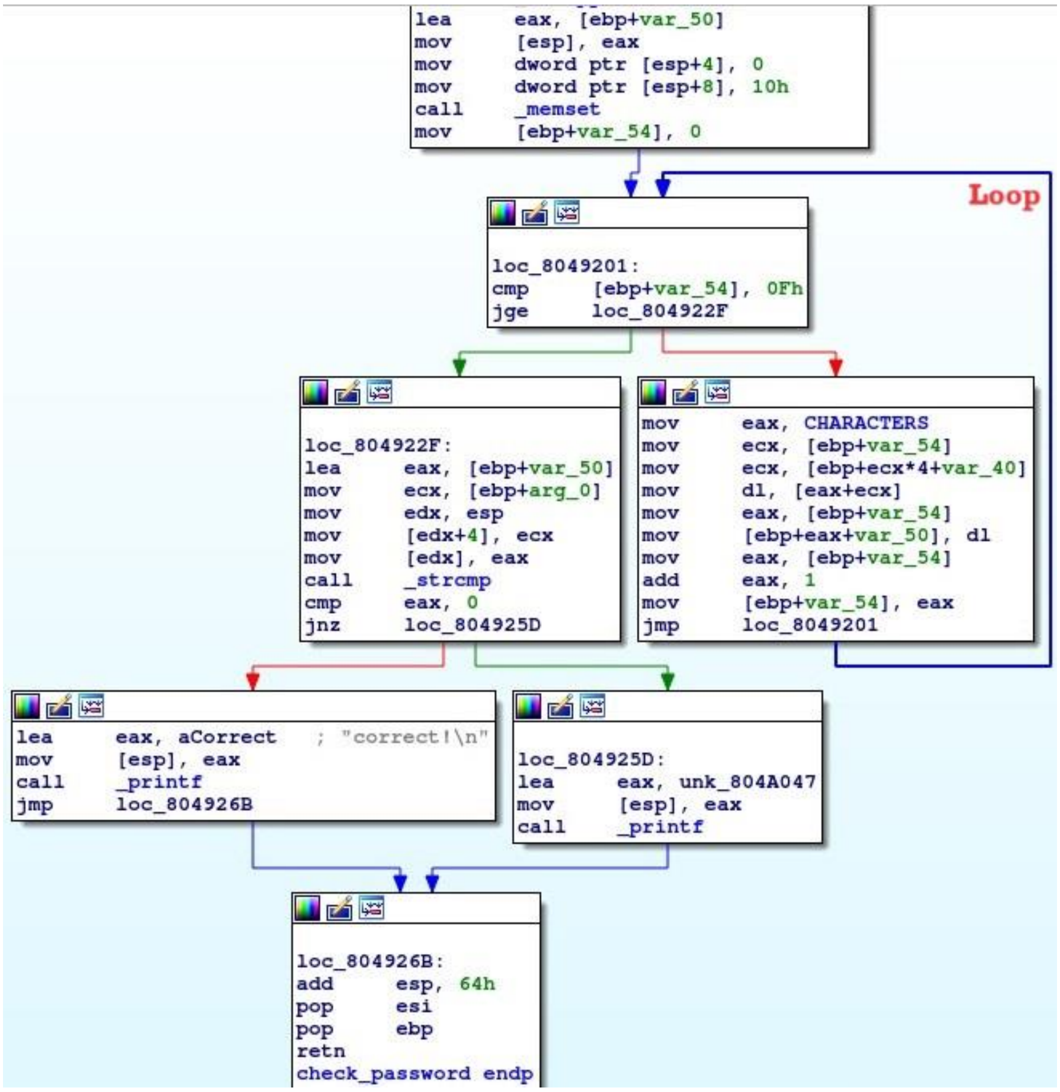
Second Step



```
push    ebp
mov     ebp, esp
push    esi
sub     esp, 64h
mov     eax, [ebp+arg_0]
xor     ecx, ecx
lea     edx, unk_804A064
lea     esi, [ebp+var_40]
mov     [esp], esi
mov     [esp+4], edx
mov     dword ptr [esp+8], 3Ch ; '<'
mov     [ebp+var_58], eax
mov     [ebp+var_5C], ecx
call    _memcpy
lea     eax, [ebp+var_50]
mov     [esp], eax
mov     dword ptr [esp+4], 0
mov     dword ptr [esp+8], 10h
call    _memset
mov     [ebp+var_54], 0
```

- It looked a little complicated with logic brunching and a loop in the **check_password** function.

- The only unusual suspicious context found in **unk_804A064.** Rest can be understood that how the password is filtered in different functions and how the _memcpy, _memset etc. are prepared.



```
lea     eax, [ebp+var_50]
mov     [esp], eax
mov     dword ptr [esp+4], 0
mov     dword ptr [esp+8], 10h
call    _memset
mov     [ebp+var_54], 0
```

```
loc_8049201:
cmp     [ebp+var_54], 0Fh
jge     loc_804922F
```

```
loc_804922F:
lea     eax, [ebp+var_50]
mov     ecx, [ebp+arg_0]
mov     edx, esp
mov     [edx+4], ecx
mov     [edx], eax
call    _strcmp
cmp     eax, 0
jnz     loc_804925D
```

```
mov     eax, CHARACTERS
mov     ecx, [ebp+var_54]
mov     ecx, [ebp+ecx*4+var_40]
mov     dl, [eax+ecx]
mov     eax, [ebp+var_54]
mov     [ebp+eax+var_50], dl
mov     eax, [ebp+var_54]
add     eax, 1
mov     [ebp+var_54], eax
jmp     loc_8049201
```

```
lea     eax, aCorrect    ; "correct!\n"
mov     [esp], eax
call    _printf
jmp     loc_804926B
```

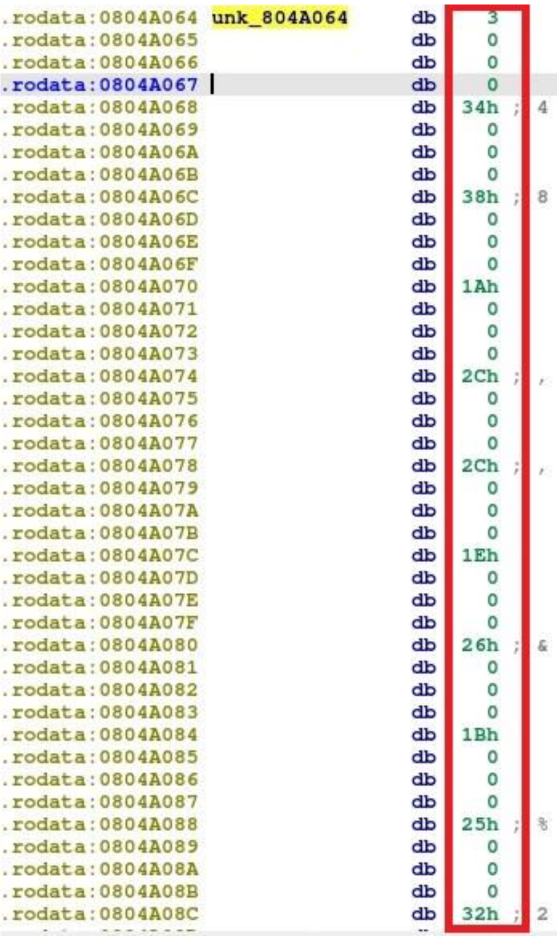
```
loc_804925D:
lea     eax, unk_804A047
mov     [esp], eax
call    _printf
```

```
loc_804926B:
add     esp, 64h
pop     esi
pop     ebp
retn
check_password endp
```

- “**lea edx, unk_804A6064** refer to read-only data related to the **CHARACTERS** array.”

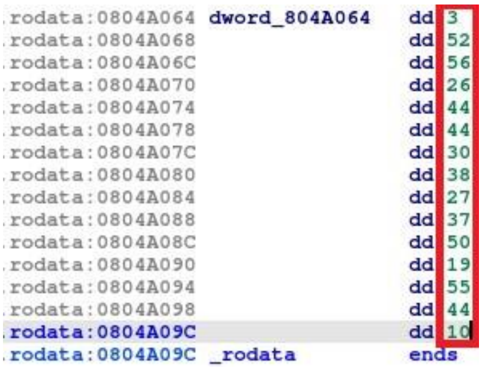
- Digging on with **unk_804A6064** found the hexadecimal **rodata** (Read only Data) values in list.

Third Step



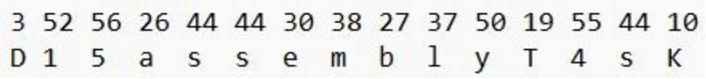
```
.rodata:0804A064 unk_804A064     db    3
.rodata:0804A065                 db    0
.rodata:0804A066                 db    0
.rodata:0804A067 |               db    0
.rodata:0804A068                 db  34h ; 4
.rodata:0804A069                 db    0
.rodata:0804A06A                 db    0
.rodata:0804A06B                 db    0
.rodata:0804A06C                 db  38h ; 8
.rodata:0804A06D                 db    0
.rodata:0804A06E                 db    0
.rodata:0804A06F                 db    0
.rodata:0804A070                 db  1Ah
.rodata:0804A071                 db    0
.rodata:0804A072                 db    0
.rodata:0804A073                 db    0
.rodata:0804A074                 db  2Ch ; ,
.rodata:0804A075                 db    0
.rodata:0804A076                 db    0
.rodata:0804A077                 db    0
.rodata:0804A078                 db  2Ch ; ,
.rodata:0804A079                 db    0
.rodata:0804A07A                 db    0
.rodata:0804A07B                 db    0
.rodata:0804A07C                 db  1Eh
.rodata:0804A07D                 db    0
.rodata:0804A07E                 db    0
.rodata:0804A07F                 db    0
.rodata:0804A080                 db  26h ; &
.rodata:0804A081                 db    0
.rodata:0804A082                 db    0
.rodata:0804A083                 db    0
.rodata:0804A084                 db  1Bh
.rodata:0804A085                 db    0
.rodata:0804A086                 db    0
.rodata:0804A087                 db    0
.rodata:0804A088                 db  25h ; %
.rodata:0804A089                 db    0
.rodata:0804A08A                 db    0
.rodata:0804A08B                 db    0
.rodata:0804A08C                 db  32h ; 2
```

- Here the list of hexadecimal values is found.



```
.rodata:0804A064 dword_804A064   dd 3
.rodata:0804A068                 dd 52
.rodata:0804A06C                 dd 56
.rodata:0804A070                 dd 26
.rodata:0804A074                 dd 44
.rodata:0804A078                 dd 44
.rodata:0804A07C                 dd 30
.rodata:0804A080                 dd 38
.rodata:0804A084                 dd 27
.rodata:0804A088                 dd 37
.rodata:0804A08C                 dd 50
.rodata:0804A090                 dd 19
.rodata:0804A094                 dd 55
.rodata:0804A098                 dd 44
.rodata:0804A09C                 dd 10
.rodata:0804A09C _rodata         ends
```

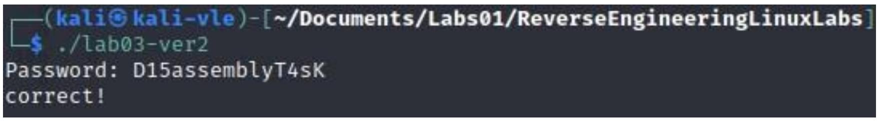
- Then the values are converted to decimal which each represent the corresponding in the Alphabet shown below.



```
3 52 56 26 44 44 30 38 27 37 50 19 55 44 10
D 1  5  a  s  s  e  m  b  l  y  T  4  s  K
```

Password

- There gives the expected string for password which later confirmed in the terminal.



```
┌──(kali㉿kali-vle)-[~/Documents/Labs01/ReverseEngineeringLinuxLabs]
└─$ ./lab03-ver2
Password: D15assemblyT4sK
correct!
```

Topic	Time
Lab03	10 hours
Report writing	3 hours