

WGMY Fortnite Challenge #0402

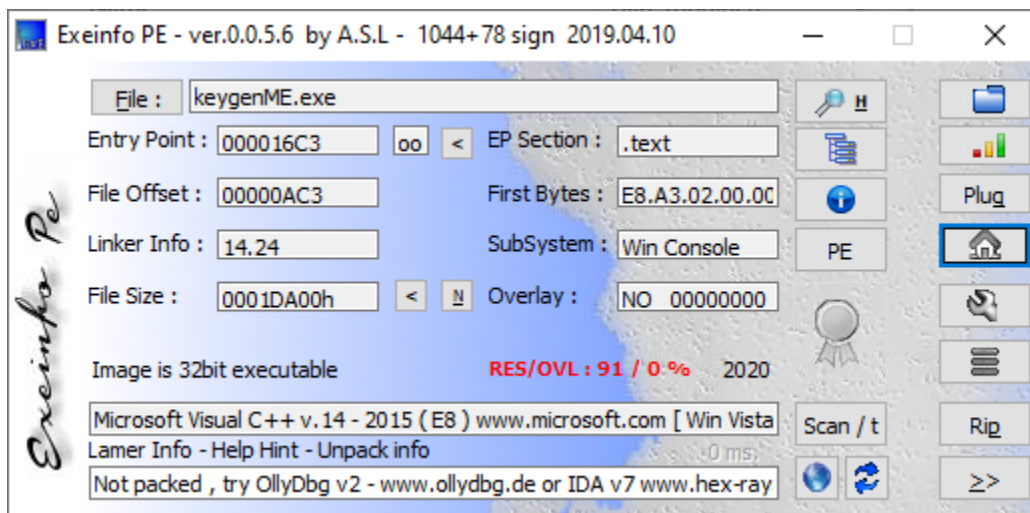
By : H0j3n (UiTM)



Tools : IDA Pro x32 , X32dbg , ExeinfoPe

```
---  
#0402 Keygenme  
  
We need the key badly but too bad, the server is no  
longer up. Can you help us to get the key? Even happier  
of you can give us a keygen.  
---
```

First lets analyze this file using ExeinfoPE .



This file is 32bit executable so lets use Ida Pro to see how the program works!



I set breakpoint before the program end to see what the output .

```
D:\CTF\Wargames Challenge\#1\keygenME.exe
```

```
keygenme - wgmy2uni
serial: 3535353535353535
nope!
```

Yeah its not working mhhh lets take a look inside the function!

```

lea     ecx, [ebp+Buf] ; Str
mov     [ebp+eax+Buf], 0
call    sub_F71000
cmp     eax, 1
mov     edx, offset aNope ; "nope!\n"
mov     ecx, offset aCongratz ; "congratz!\n"
cmovnz  ecx, edx
push    ecx
call    sub_F71240
mov     ecx, [ebp+var_4]
add     esp, 4
xor     ecx, ebp
xor     eax, eax
call    @__security_check_cookie@4 ; __security_check_cookie(x)
mov     esp, ebp
pop     ebp
retn

```

It call `sub_F71000` and under that we found `cmp eax` with 1 and from that functions I feel curious what happens inside that function so lets we take a look!

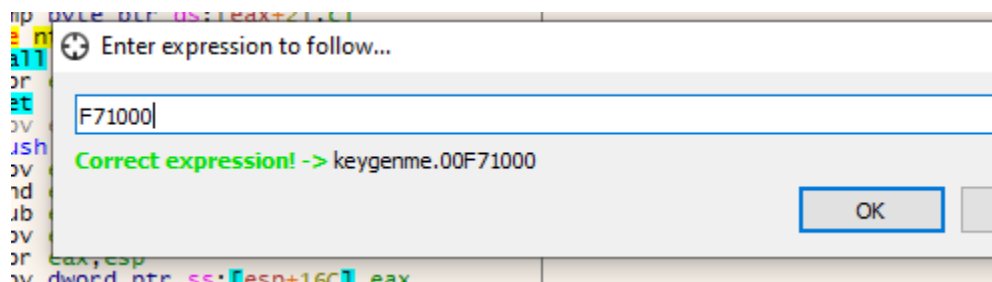
```

v16 = xmmword_F72300;
if ( strlen(Str) == 19 )
{
    v3 = strchr(v2, 45);
    if ( v3 )
    {
        do
        {
            v3 = strchr(v3 + 1, 45);
            ++v1;
        }
        while ( v3 );
        if ( v1 == 4 )
        {
            v4 = strtok(v2, "-");
            if ( !v4 )
                return v1 == 8;
        }
    }
    LABEL_6:
    v5 = 0;
    while ( 1 )
    {
        v10 = v4[v5];
        if ( !isalnum(v10) )
            break;
        v6 = v4[v5];
        if ( v6 > 96 && v6 < 123 )

```

I use decompiler and found out that the serial key must 19 length and if its true they will go through some if statement and we found something like `isalnum()`, `strtok()`, `strchr()` and when I have done a lot of googling with this function and we found out that the format should be like this `XXXX-XXXX-XXXX-XXXX` which satisfy the length and the format which consist of alphanumeric ! How do i know that format because I have try coding back some of the function like `strchr()` with `++v1`; and it needs 3 "-" to go for next if statement. `Strchr()` will find the first occurrence of "-" and if there is any strings left it will continue

with the looping until not strings left .Now what we should do is to find this memory address to look further using x32dbg!



So we know that function just now is **F71000** so in **x32dbg** use **CTRL + G** to find the address and we set the breakpoint in that address ! I have been struggling to understand how this things work so I try but randomly alphanumeric so find out how it works and when I input **aaaa-aaaa-aaaa-aaaa** its not passing all the four if statement inside that function .

```

v10 = v4[v5];
if ( !isalnum(v10) )
    break;
v6 = v4[v5];
if ( v6 > 96 && v6 < 123 )
    break;
if ( v6 > 47 && v6 < 58 )
    break;
v7 = Stra[v6];
Stra[v6] = v7 + 1;
if ( v7 )
    break;
if ( ++v5 >= 4 )
{
    v4 = strtok(0, "-");
    ++v1;
    if ( v4 )
        goto LABEL_6;
    return v1 == 8;
}

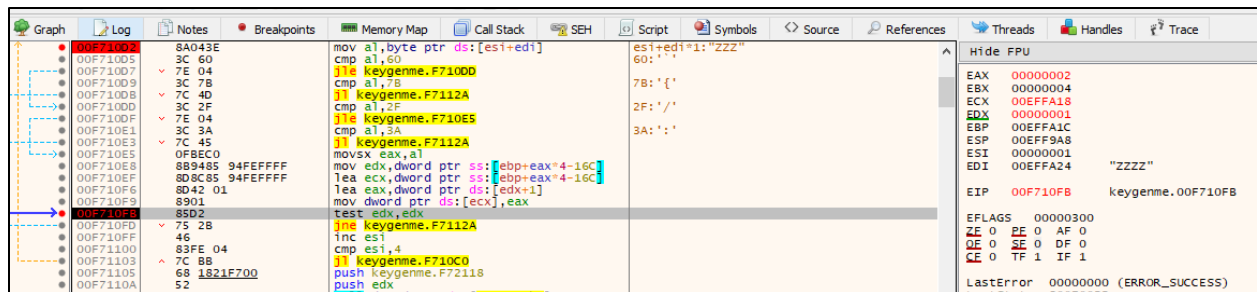
```

```

v5 = 0;
while ( 1 )
{
    v10 = v4[v5];
    if ( !isalnum(v10) )
        break;
    v6 = v4[v5];
    if ( v6 > '`' && v6 < '{' )
        break;
    if ( v6 > '/' && v6 < ':' )
        break;
    v7 = Stra[v6];
    Stra[v6] = v7 + 1;
    if ( v7 )
        break;
    if ( ++v5 >= 4 )
    {
        v4 = strtok(0, "-");
        ++v1;
        if ( v4 )
            goto LABEL_6;
        return v1 == 8;
    }
}

```

The left one with decimal and the right one with char conversion so our serial key must pass this value ! I try a lot of character till I found something which it accept Uppercase character like A-Z and that's when I look at their hex value and its bigger than all the hex so lets try one by one ! when I first put Z in all character inside the serial like this **ZZZZ-ZZZZ-ZZZZ-ZZZZ** it only accept the first Z and the second Z it will return nope!. I check all of this inside x32dbg



I found out that the character that the program accept it will make **edx 0** while false character will make **edx 1** so we just need to make a serial that make all **edx** become 0 till the end of the serial key and it takes time actually till I found the serial key that works which this ! **ZBCD-FHIJ-LNOP-QTUV** . Now when I found this first serial key that works I try to create another one on how this things work and actually it accepts only **[B,C,D,F,H,I,J,K,L,N,O,P,Q,T,U,V,X,Z]** inside the serial key and no duplicate character !

Here some of serial keys that works

- BCDF-HIJK-LNOP-QTUV
- BCDF-HIJK-LNOP-QTUX
- HIFD-LJKN-PQTU-VXZB
- ZBCD-FHIJ-LNOP-QTUV

How about we create many serial keys that works on how much we want right? I try to create it using Python! 😊

```
Warning, you are using the root account, you may harm your system.

import random

listKey = []

inputs = int(input("How many Serial Key you want?: "))
for i in range(0,inputs):
    listAlpha = ["B","C","D","F","H","I","J","K","L","N","O","P","Q","T","U","V","X","Z"]
    random.shuffle(listAlpha)
    strkey = "".join(listAlpha[0:4])+"-"+"".join(listAlpha[4:8])+"-"+"".join(listAlpha[8:12])+"-"+"".join(listAlpha[12:16])
    if strkey not in listKey:
        listKey.append(strkey)

for i in range(0,len(listKey)):
    print("Serial "+str(i+1)+" : "+listKey[i])
```

```
root@kali: ~/Challenge/#1
Serial 196 : IOXT-UKPC-BNQD-LHZJ
Serial 197 : BQCV-UNOF-XKTD-HILP
Serial 198 : BOTL-KZFN-DUJQ-VIHX
Serial 199 : PCID-KQOJ-BNTH-UFZV
Serial 200 : LCXF-DNTZ-UBVQ-OPHK
root@kali: ~/Desktop/CTF/MALAYSIA/Warg
How many Serial Key you want?: 500
Serial 1 : UDBV-JQHL-TMFK-PCXZ
Serial 2 : ZHVF-TNJK-ULCQ-IBOD
Serial 3 : OBDN-VTJC-XKHU-ZILQ
Serial 4 : FLHD-BPJZ-VITK-NXUO
Serial 5 : LHTU-NVKQ-BJXI-CPDF
Serial 6 : QIFP-JULV-OTNZ-BDXK
Serial 7 : NLQI-JFKC-HVOZ-BUPX
Serial 8 : KNUB-FDZL-JHQO-XPIC
Serial 9 : PBIC-VXTH-DNLU-QQFK
Serial 10 : UHBN-TDKQ-JPCO-VXZF
Serial 11 : FTQH-OILB-XCZP-VKJD
Serial 12 : TZNL-QJBW-PKDH-CVOI
Serial 13 : LVBZ-HPUF-ICKO-DJNQ
Serial 14 : XKDI-VCLU-QHPZ-TBNJ
Serial 15 : BZTK-PNXO-IQHC-LDJF
Serial 16 : IPNF-OKHZ-LVTX-DQUB
Serial 17 : QCZF-HLON-UVBP-XIJD
Serial 18 : NXIJ-UCZO-FTBK-LQDV
Serial 19 : KOCT-QPZN-LHDB-VXIF
Serial 20 : CBQZ-UXFV-KPIH-NJOD
Serial 21 : DJHC-TUBZ-0VKX-LFQP
```

Thank you for this great challenge ! 😊