Test SHA-256

Localisation: 0040331B **Fonction:** ?
Type: Hash **Sévérité:** Moyenne

```
.text:00401F6F ; __unwind { // sub_474F60
.text:00401F6F                 lea     eax, [ebp+var_A]
.text:00401F72                 mov     [esp+8], eax    ; int
.text:00401F76                 mov     dword ptr [esp+4], offset
   aHelloWorld ; "Hello, World!"
.text:00401F7E                 mov     dword ptr [esp], offset
   dword_565030 ; int
.text:00401F85                 mov     [ebp+fctx.call_site], 2
.text:00401F8C                 call    sub_448120
.text:00401F91                 mov     dword ptr [esp], offset
   sub_4020A0 ; _onexit_t
.text:00401F98                 call    sub_409C20
.text:00401F9D                 lea     eax, [ebp+var_A+1]
.text:00401FA0                 mov     [esp+8], eax    ; int
.text:00401FA4                 mov     dword ptr [esp+4], offset
   aBea8e217036cb3 ; "bea8e217036cb3b738e207fe5d40266828bc196"...
.text:00401FAC                 mov     dword ptr [esp], offset
   dword_565034 ; int
.text:00401FB3                 mov     [ebp+fctx.call_site], 1
.text:00401FBA                 call    sub_448120
.text:00401FBF                 mov     dword ptr [esp], offset
   sub_401FF0 ; _onexit_t
.text:00401FC6                 call    sub_409C20
.text:00401FCB                 lea     eax, [ebp+fctx]
.text:00401FCE                 mov     [esp], eax      ; lpfctx
.text:00401FD1                 call    _Unwind_SjLj_Unregister
.text:00401FD6                 leave
.text:00401FD7                 retn
```

Analyse

Dans la fonction ci-dessus, le programme stocke la valeur

bea8e217036cb3b738e207fe5d40266828bc1969fd8538d533ea39f4e40ffc8f

qui semble être un digest (SHA-256) en `dword_565034`.

```
.text:00403770 loc_403770:
.text:00403770                 cmp     ecx, ecx
.text:00403772                 repe cmpsb
.text:00403774                 jnz     loc_403333
.text:0040377A                 mov     eax, [ebp+var_24]
.text:0040377D                 mov     edi, [eax-0Ch]
.text:00403780                 lea     edx, [eax-0Ch]
.text:00403783                 test    edi, edi
.text:00403785                 jnz     loc_403996
```

Analyse

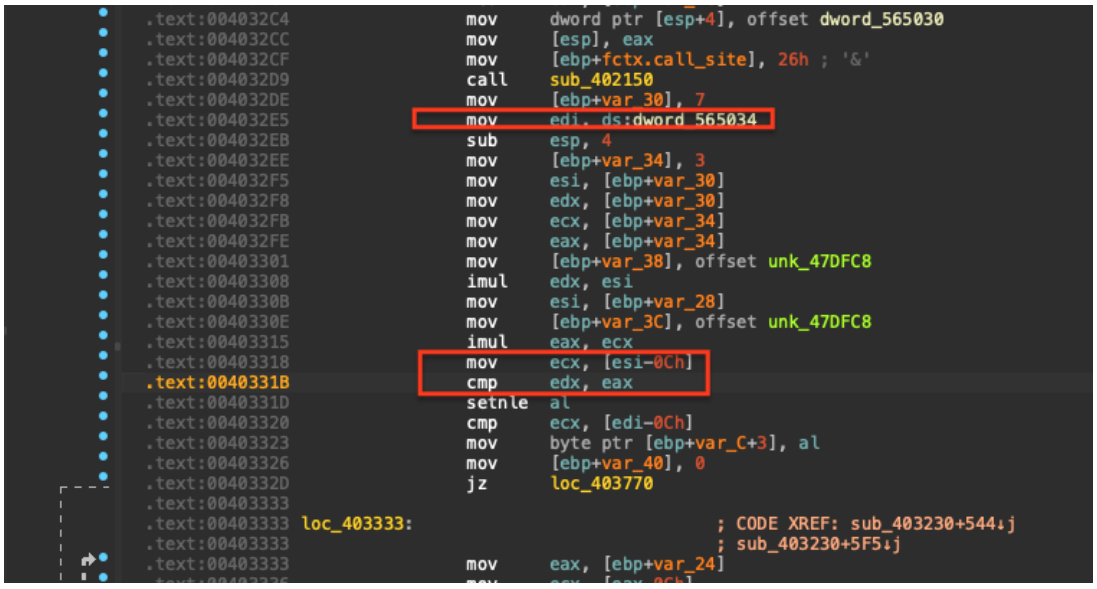
Ensuite, le programme compare le SHA-256 de l'entrée utilisateur avec la valeur de hash. Si les deux sont identiques, le programme saute à la fonction situé en `loc_403996`

```
loc_403996:
.text:00403996                  mov     esi, [edx+8]
.text:00403999                  test    esi, esi
.text:0040399B                  js      short loc_4039B5
.text:0040399D                  lea     eax, [ebp+var_24]
.text:004039A0                  mov     [esp], eax
.text:004039A3                  mov     [ebp+fctx.call_site], 25h ; '%'
.text:004039AD                  call    sub_445EE0
.text:004039B2                  mov     eax, [ebp+var_24]
.text:004039B5
.text:004039B5 loc_4039B5:
.text:004039B5                  movzx   eax, byte ptr [eax]
.text:004039B8                  mov     dword ptr [esp], offset Format
    ; "Byte 0: %02x\n"
.text:004039BF                  mov     [ebp+fctx.call_site], 25h ; '%'
.text:004039C9                  mov     [esp+4], eax
.text:004039CD                  call    printf
.text:004039D2                  jmp     loc_40378B
```

Analyse

La fonction `loc_4039B5` permet d'afficher le premier octet de l'entrée.

Screenshot

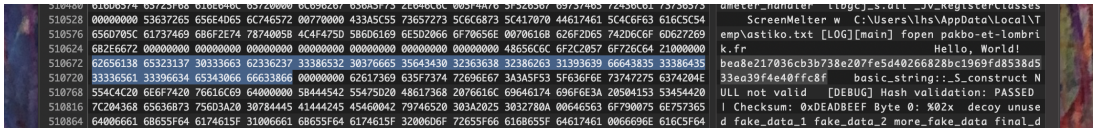


Analyse

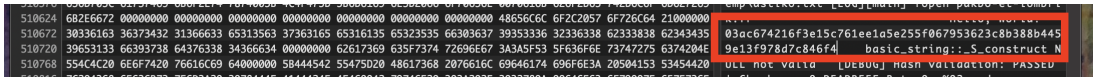
Après avoir identifé ce potentiel comportement, nous avons tenté de le produire. Pour cela nous avons cherché le hash directement dans l'executable afin de le modifier afin de le remplacer par un hash de notre choix : le SHA-256 de 1234

03ac674216f3e15c761ee1a5e255f067953623c8b388b4459e13f978d7c846f4

Screenshot



Screenshot



Analyse

Cependant, en effectuant `m.exe 1234` le comportement reste inchangé. Deux hypothèses sont donc à considérer :

- le programme effectue un test d'intégrité sur son contenu ;
- le hash identifié n'impacte en aucun cas la fonction légitime `echo`.