La rétro-ingénierie de code malveillant dans la CTI

Analyse de l'évolution d'une chaîne d'infection

Charles Meslay

TDR by sekoia
THREAT DETECTION & RESEARCH

Cyber Threat Intelligence

~~Cyber Threat Intelligence~~

Renseignement sur la menace cyber

TDR by sekoia

Étudier le contexte et les capacités techniques des attaquants pour mieux se défendre

Moyens techniques

INFRASTRUCTURE

Quelle infrastructure est utilisée par l'acteur malveillant pour communiquer avec ses implants, effectuer des actions de reconnaissance ou exfiltrer des données du réseau ?

Moyens techniques

INFRASTRUCTURE

Quelle infrastructure est utilisée par l'acteur malveillant pour communiquer avec ses implants, effectuer des actions de reconnaissance ou exfiltrer des données du réseau ?

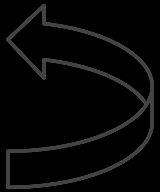
CODES MALVEILLANTS

Quels outils, codes malveillants et vecteurs d'intrusion sont utilisés par un acteur malveillant afin de compromettre ses cibles, de se latéraliser et de réussir sa mission ?

Processus d’analyse d’un code

Processus d’analyse d’un code

> Comprendre ce qu’il fait
> Signer le code
> Pivoter / Trouver des nouveaux fichiers



(> Extraire les serveurs C2)

Processus d’analyse d’un code

> Comprendre ce qu’il fait
> Signer le code
> Pivoter / Trouver des nouveaux fichiers

(> Extraire les serveurs C2)

TDR by sekoia

Processus d’analyse d’un code

> Comprendre ce qu’il fait
> Signer le code
> Pivoter / Trouver des nouveaux fichiers

(> Extraire les serveurs C2)

Processus d’analyse d’un code

> Comprendre ce qu’il fait
> Signer le code
> Pivoter / Trouver des nouveaux fichiers

(> Extraire les serveurs C2)

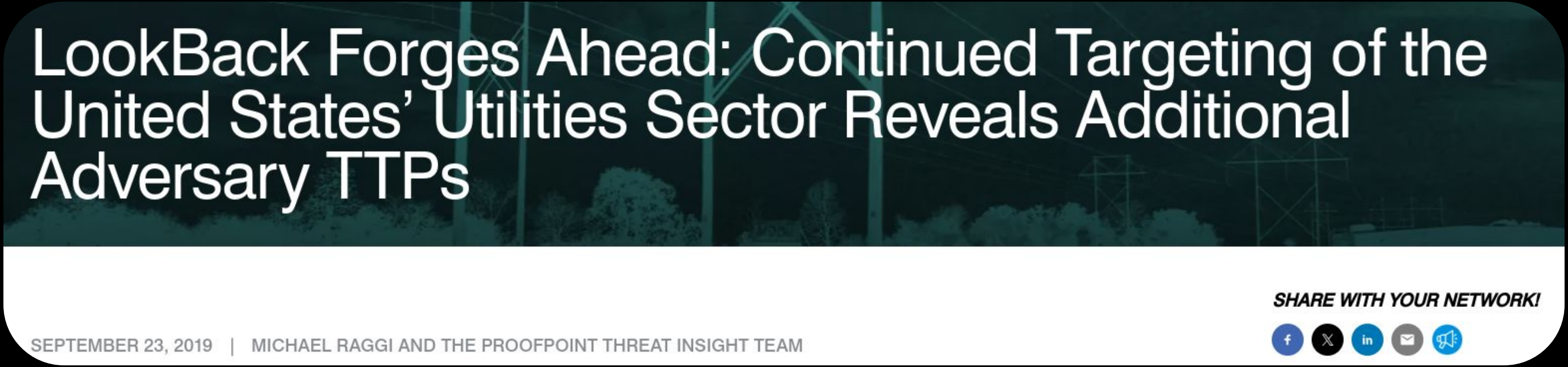
TDR by sekoia

Processus d'analyse d'un code

> Comprendre ce qu'il fait
> Signer le code
> Pivoter / Trouver des nouveaux fichiers

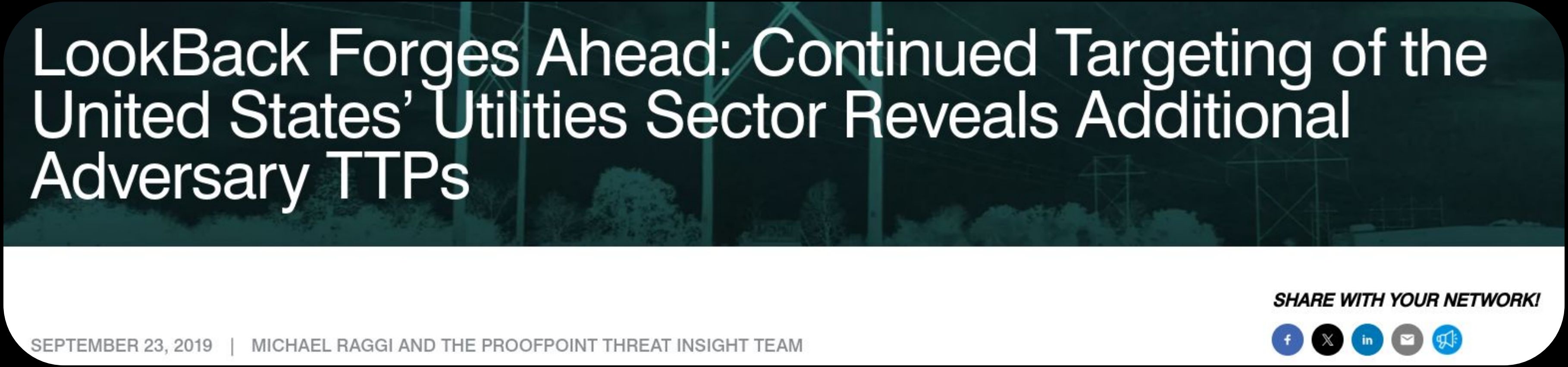
(> Extraire les serveurs C2)

Exemple : TA410



LookBack Forges Ahead: Continued Targeting of the United States' Utilities Sector Reveals Additional Adversary TTPs

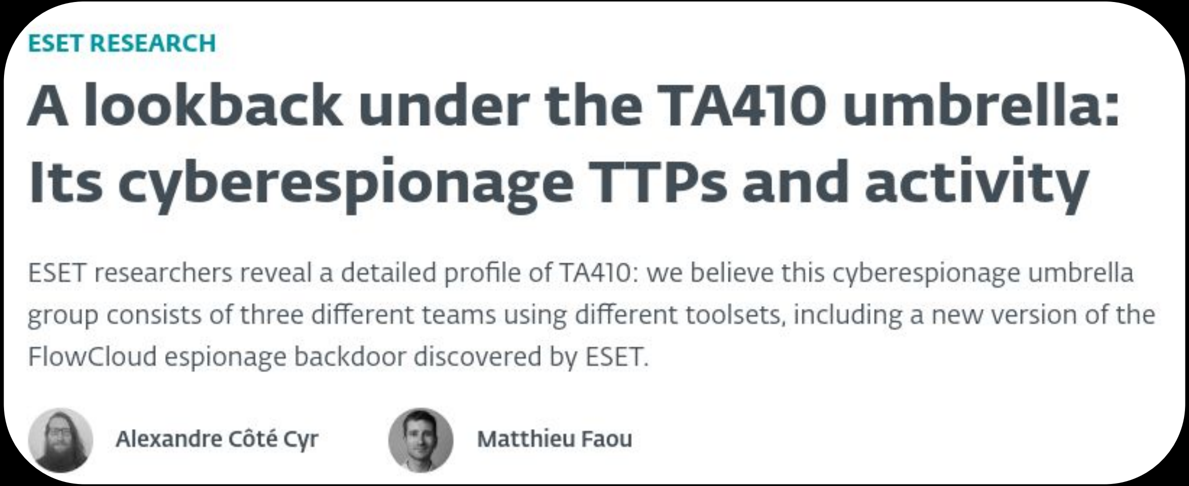
SEPTEMBER 23, 2019 | MICHAEL RAGGI AND THE PROOFPOINT THREAT INSIGHT TEAM



LookBack Forges Ahead: Continued Targeting of the United States' Utilities Sector Reveals Additional Adversary TTPs

SEPTEMBER 23, 2019 | MICHAEL RAGGI AND THE PROOFPOINT THREAT INSIGHT TEAM

SHARE WITH YOUR NETWORK!



ESET RESEARCH

A lookback under the TA410 umbrella: Its cyberespionage TTPs and activity

ESET researchers reveal a detailed profile of TA410: we believe this cyberespionage umbrella group consists of three different teams using different toolsets, including a new version of the FlowCloud espionage backdoor discovered by ESET.

Alexandre Côté Cyr   Matthieu Faou

Threat Actor 410

- Menace supposée d’origine Chinoise.
    - Associée à APT10

- Sous-groupe: FlowingFrog, LookingFrog, JollyFrog

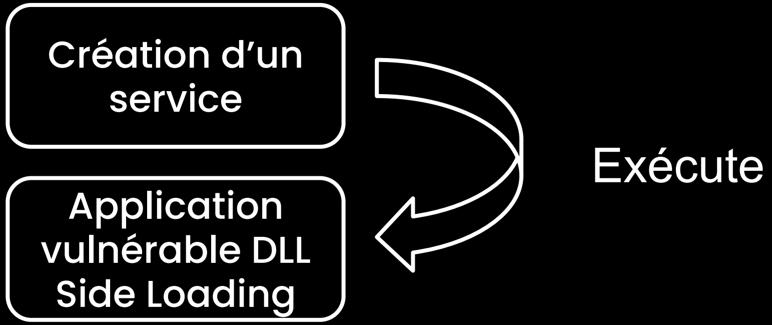
- Arsenal: Tendyron, FlowCloud, X4, LookBack, PlugX, QuasarRAT

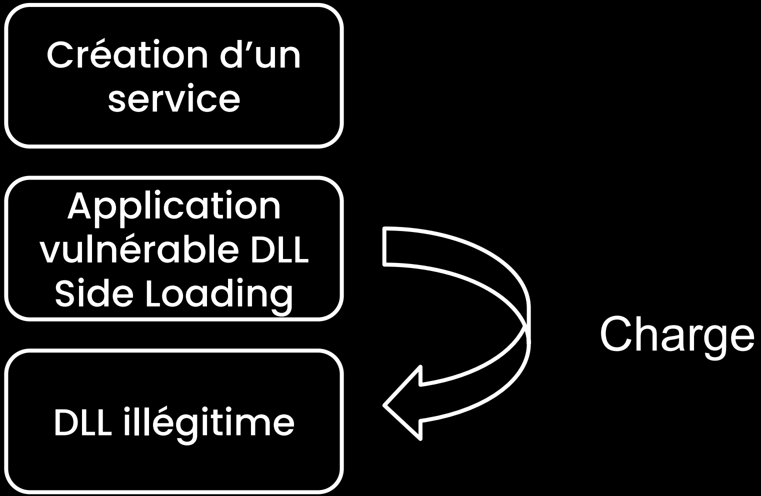
Threat Actor 410

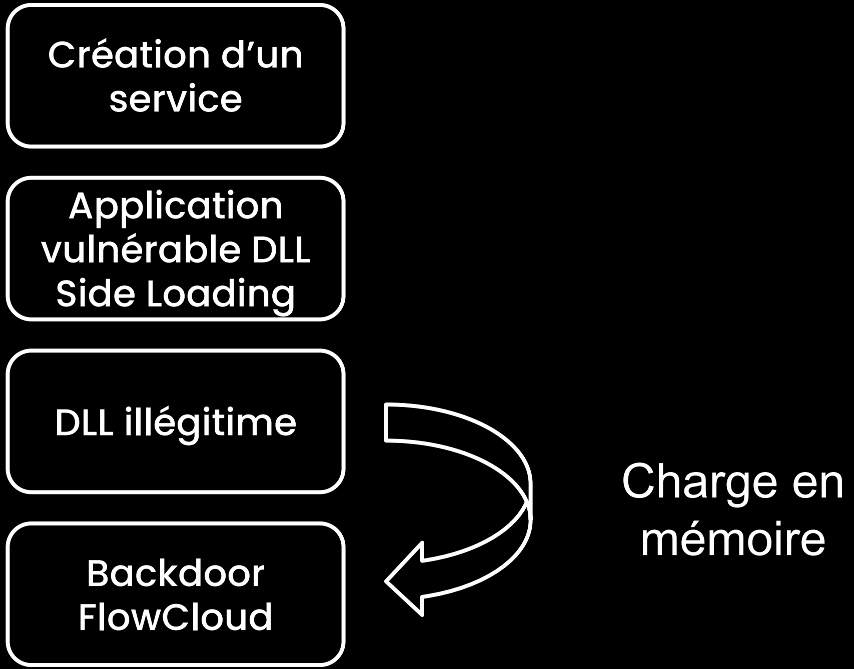
- Menace supposée d’origine Chinoise.
  - Associée à APT10
- Sous-groupe: FlowingFrog, LookingFrog, JollyFrog
- Arsenal: Tendyron, FlowCloud, X4, LookBack, PlugX, QuasarRAT

Chaîne d’exécution

Création d’un service


Création d’un service
Exécute
Application vulnérable DLL Side Loading


Création d'un service
Application vulnérable DLL Side Loading
Charge
DLL illégitime


Création d’un service
Application vulnérable DLL Side Loading
DLL illégitime
Backdoor FlowCloud
Charge en mémoire

Création d’un service

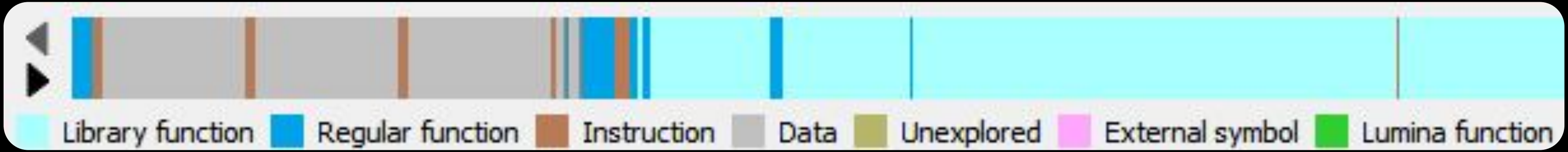
Application vulnérable DLL Side Loading

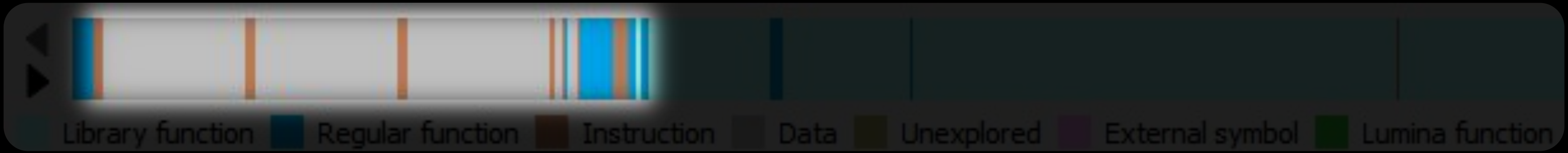
DLL illégitime

Backdoor FlowCloud

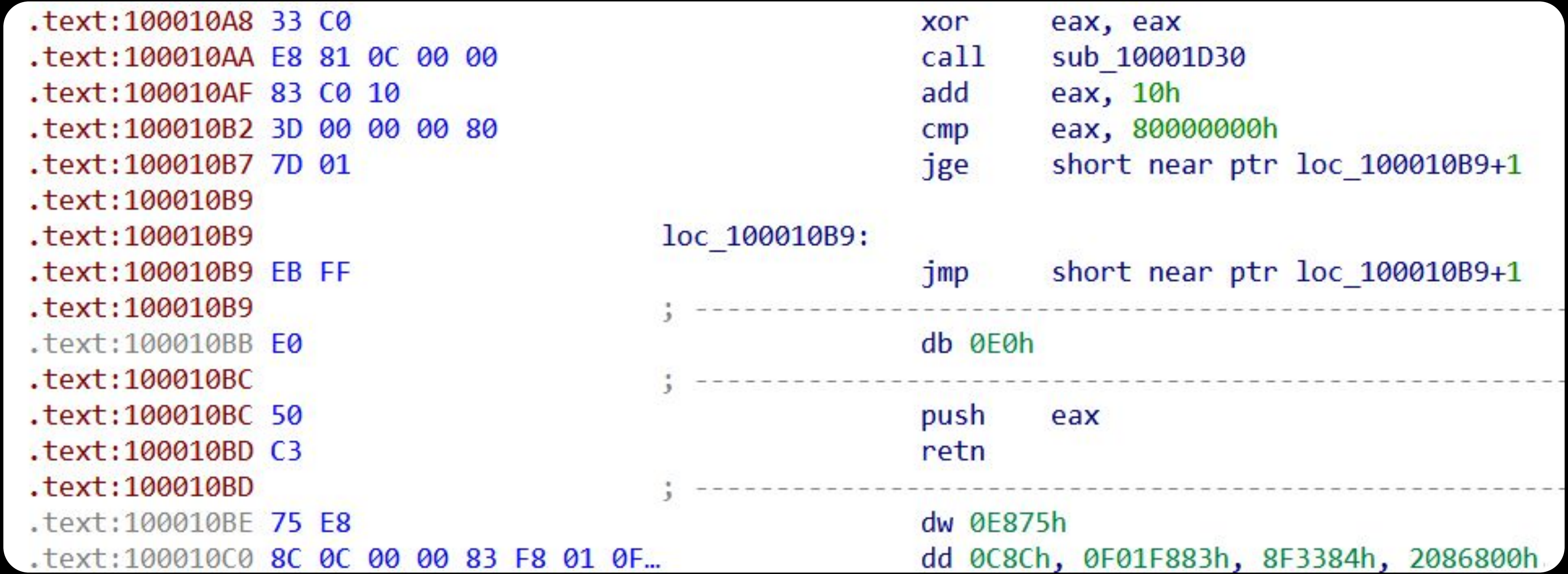
Analyse initiale

TDR by sekoia


Library function
Regular function
Instruction
Data
Unexplored
External symbol
Lumina function

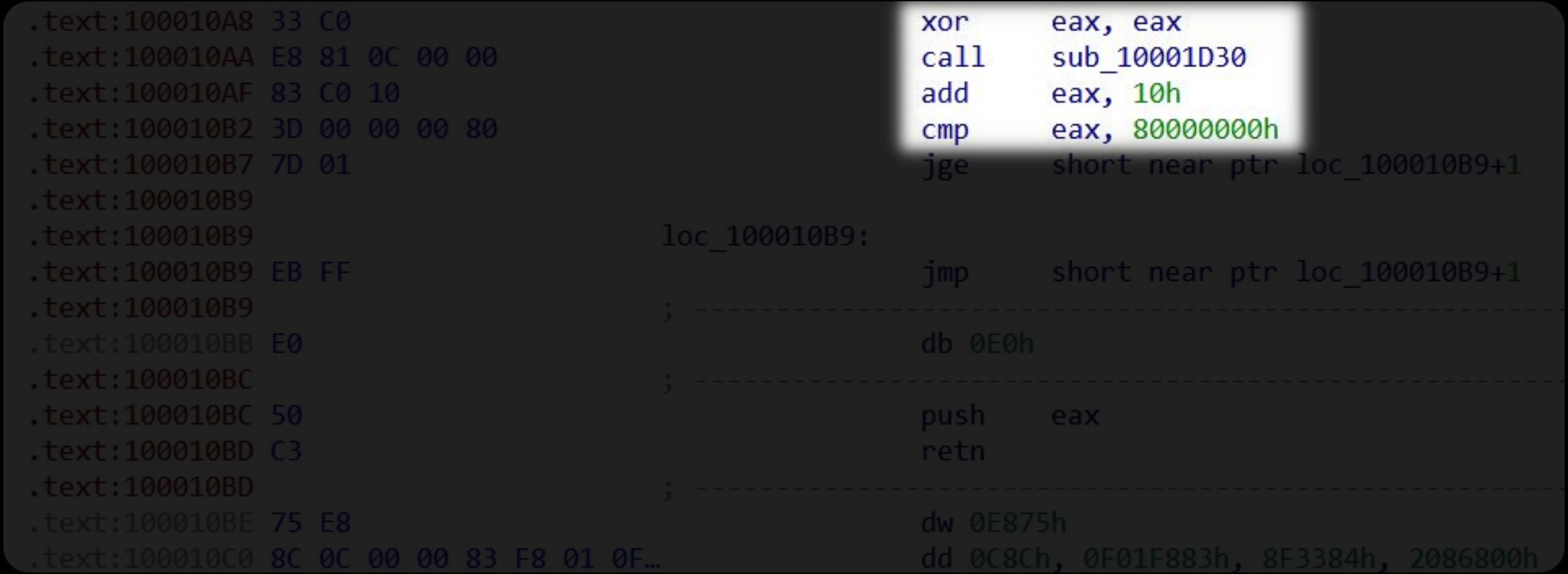


TDR by sekoia

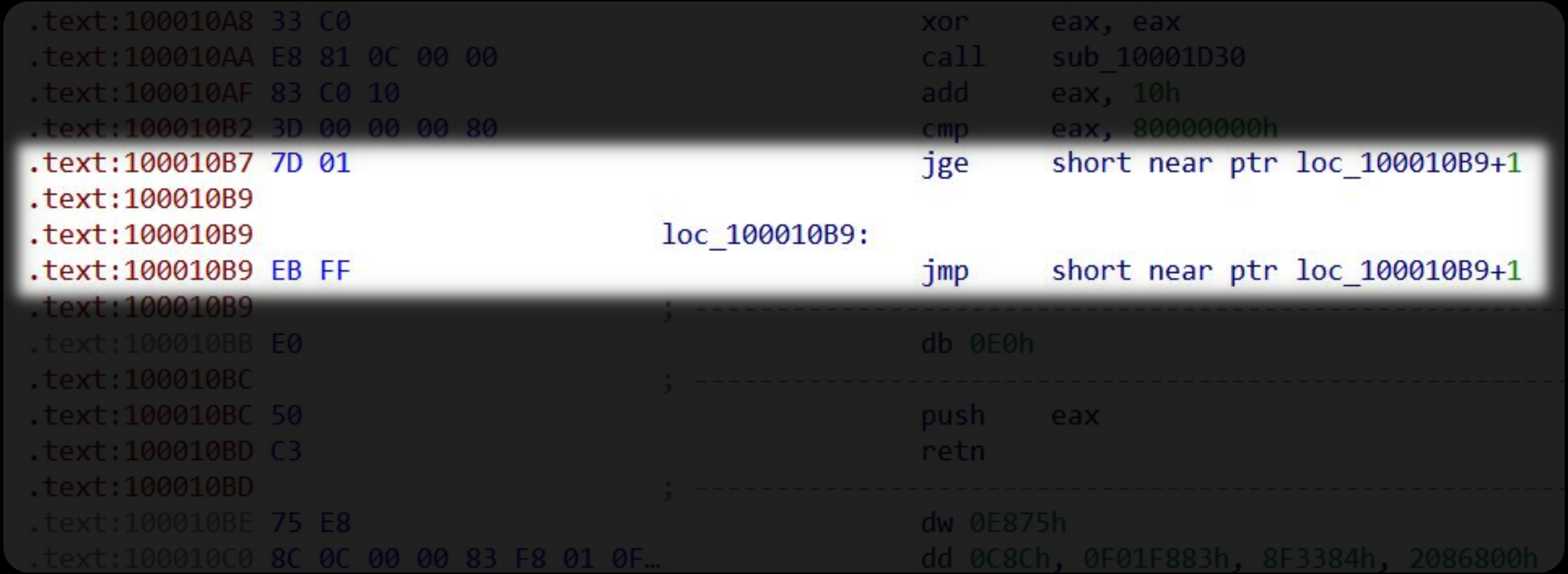


```
.text:100010A8 33 C0                                    xor     eax, eax
.text:100010AA E8 81 0C 00 00                           call    sub_10001D30
.text:100010AF 83 C0 10                                 add     eax, 10h
.text:100010B2 3D 00 00 00 80                           cmp     eax, 80000000h
.text:100010B7 7D 01                                    jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                          loc_100010B9:
.text:100010B9 EB FF                                    jmp     short near ptr loc_100010B9+1
.text:100010B9                          ; ------------------------------------------------
.text:100010BB E0                                       db 0E0h
.text:100010BC                          ; ------------------------------------------------
.text:100010BC 50                                       push    eax
.text:100010BD C3                                       retn
.text:100010BD                          ; ------------------------------------------------
.text:100010BE 75 E8                                    dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                 dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

TDR by sekoia

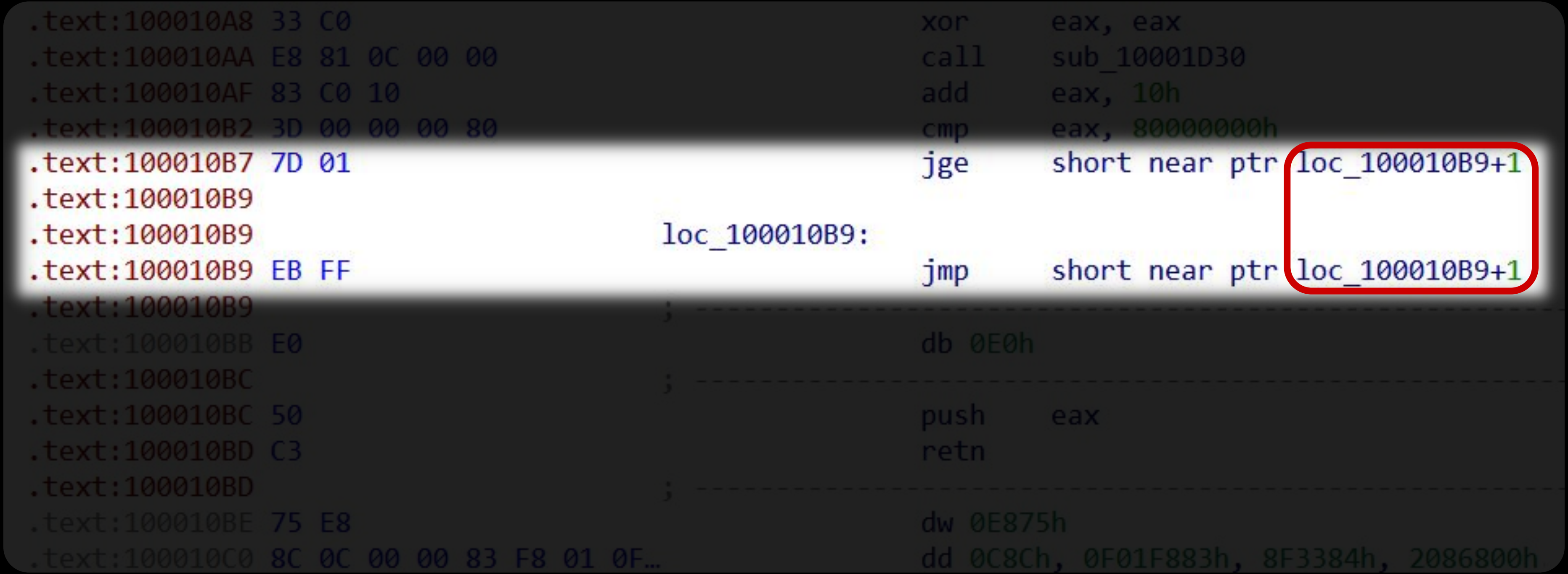


```
.text:100010A8 33 C0                                       xor     eax, eax
.text:100010AA E8 81 0C 00 00                              call    sub_10001D30
.text:100010AF 83 C0 10                                    add     eax, 10h
.text:100010B2 3D 00 00 00 80                              cmp     eax, 80000000h
.text:100010B7 7D 01                                       jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                          loc_100010B9:
.text:100010B9 EB FF                                       jmp     short near ptr loc_100010B9+1
.text:100010B9                          ; ---------------------------------------------
.text:100010BB E0                                          db 0E0h
.text:100010BC                          ; ---------------------------------------------
.text:100010BC 50                                          push    eax
.text:100010BD C3                                          retn
.text:100010BD                          ; ---------------------------------------------
.text:100010BE 75 E8                                       dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                    dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

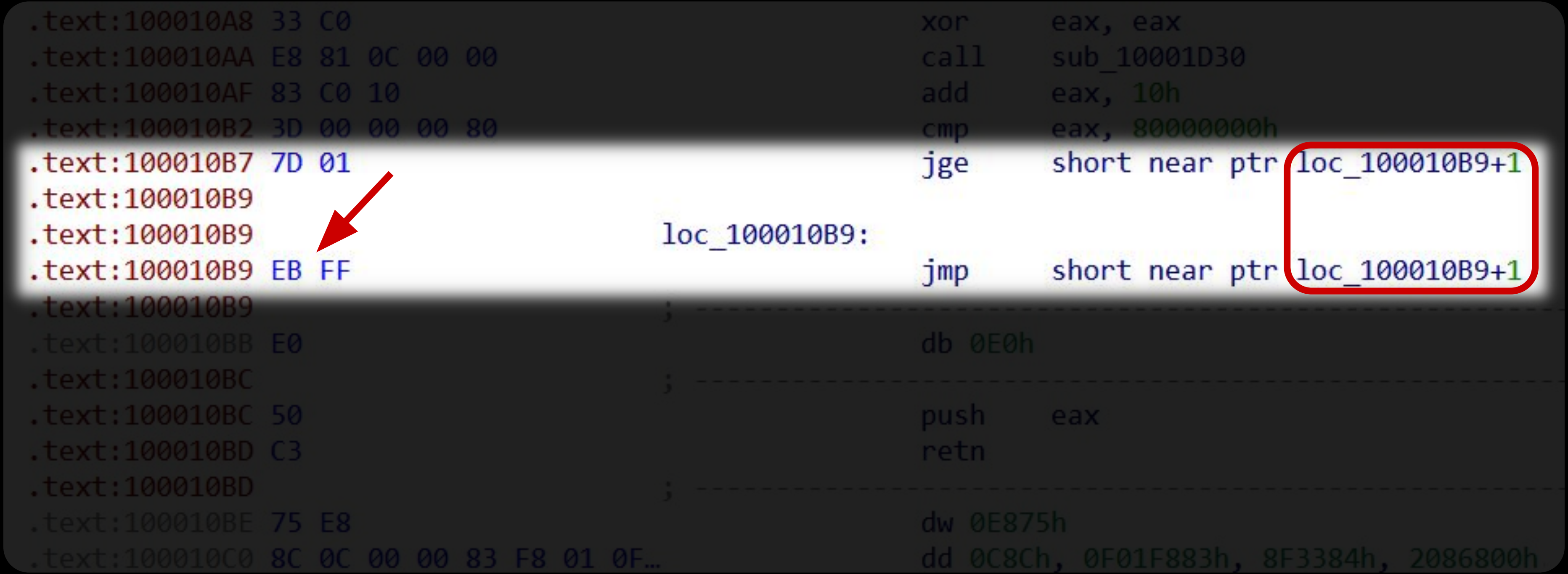


```
.text:100010A8 33 C0                                          xor     eax, eax
.text:100010AA E8 81 0C 00 00                                 call    sub_10001D30
.text:100010AF 83 C0 10                                       add     eax, 10h
.text:100010B2 3D 00 00 00 80                                 cmp     eax, 80000000h
.text:100010B7 7D 01                                          jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                              loc_100010B9:
.text:100010B9 EB FF                                          jmp     short near ptr loc_100010B9+1
.text:100010B9                              ; ---------------------------------------------------
.text:100010BB E0                                             db 0E0h
.text:100010BC                              ; ---------------------------------------------------
.text:100010BC 50                                             push    eax
.text:100010BD C3                                             retn
.text:100010BD                              ; ---------------------------------------------------
.text:100010BE 75 E8                                          dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                       dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

TDR by sekoia

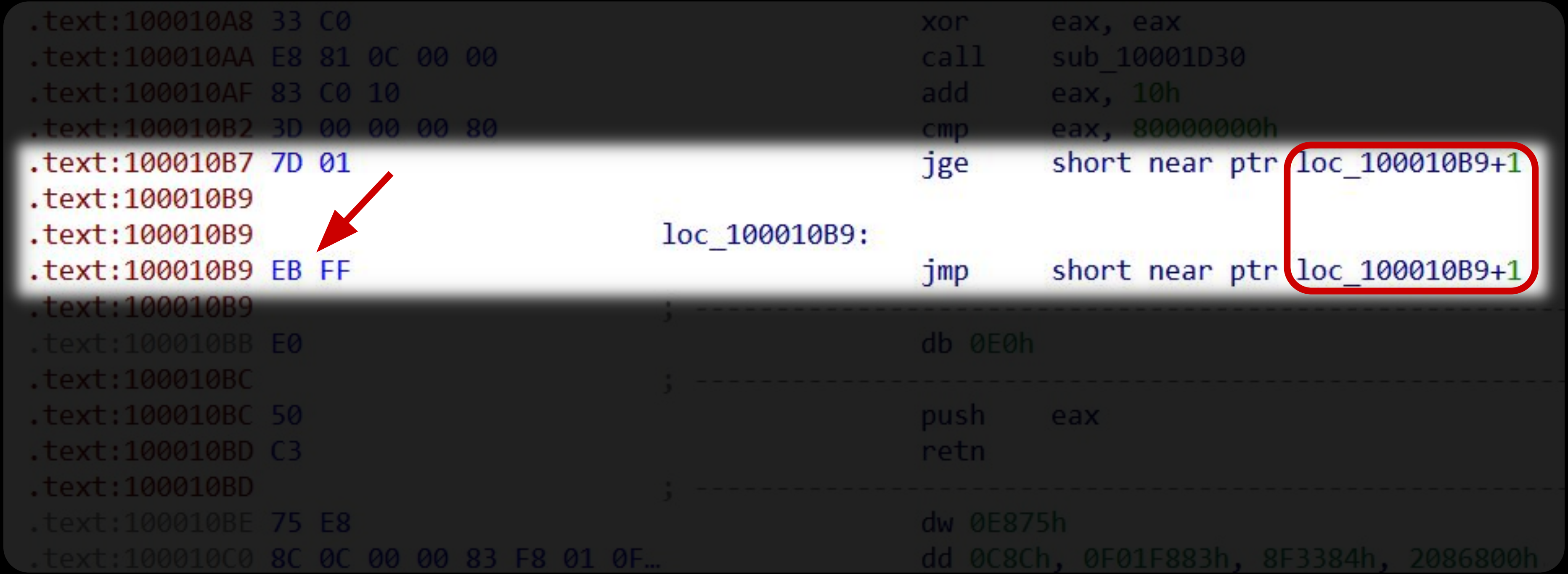


```
.text:100010A8 33 C0                                       xor     eax, eax
.text:100010AA E8 81 0C 00 00                              call    sub_10001D30
.text:100010AF 83 C0 10                                    add     eax, 10h
.text:100010B2 3D 00 00 00 80                              cmp     eax, 80000000h
.text:100010B7 7D 01                                       jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                          loc_100010B9:
.text:100010B9 EB FF                                       jmp     short near ptr loc_100010B9+1
.text:100010B9                          ; ---------------------------------------------
.text:100010BB E0                                          db 0E0h
.text:100010BC                          ; ---------------------------------------------
.text:100010BC 50                                          push    eax
.text:100010BD C3                                          retn
.text:100010BD                          ; ---------------------------------------------
.text:100010BE 75 E8                                       dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                    dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

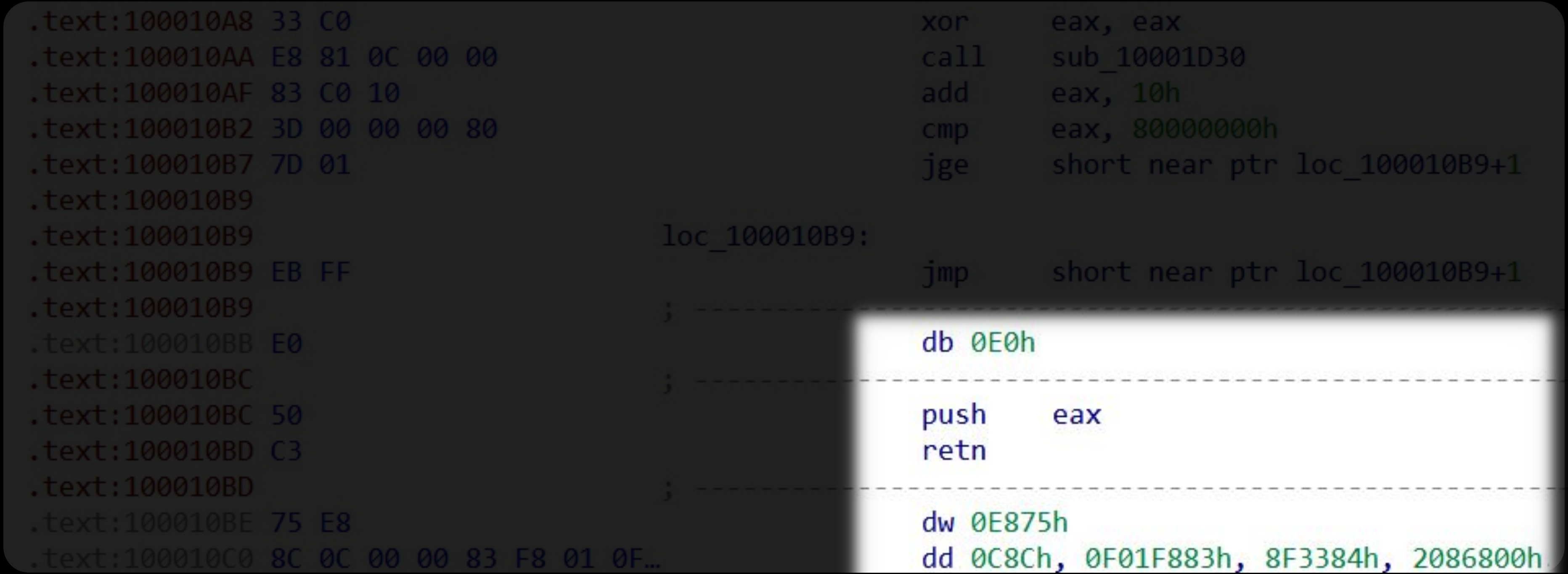


```
.text:100010A8 33 C0                                           xor     eax, eax
.text:100010AA E8 81 0C 00 00                                  call    sub_10001D30
.text:100010AF 83 C0 10                                        add     eax, 10h
.text:100010B2 3D 00 00 00 80                                  cmp     eax, 80000000h
.text:100010B7 7D 01                                           jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                              loc_100010B9:
.text:100010B9 EB FF                                           jmp     short near ptr loc_100010B9+1
.text:100010B9                              ; ---------------------------------------------
.text:100010BB E0                                              db 0E0h
.text:100010BC                              ; ---------------------------------------------
.text:100010BC 50                                              push    eax
.text:100010BD C3                                              retn
.text:100010BD                              ; ---------------------------------------------
.text:100010BE 75 E8                                           dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                        dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

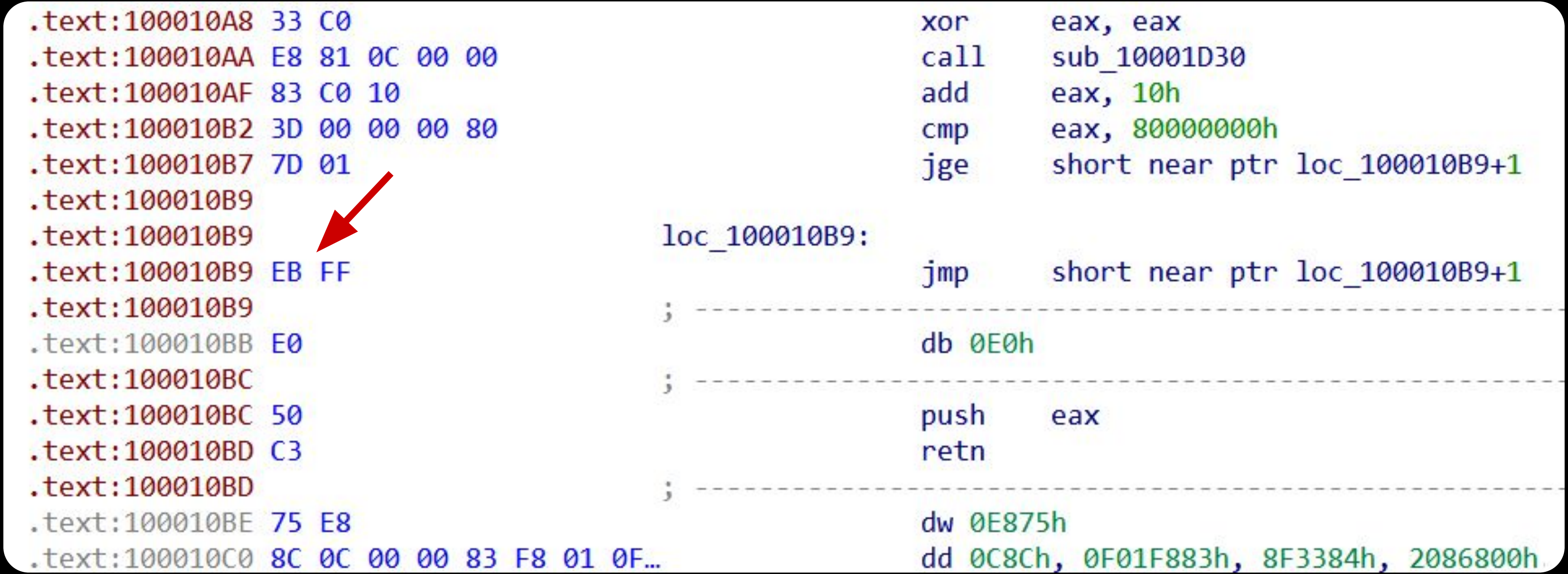
Jump in the middle



```
.text:100010A8 33 C0                                xor     eax, eax
.text:100010AA E8 81 0C 00 00                       call    sub_10001D30
.text:100010AF 83 C0 10                             add     eax, 10h
.text:100010B2 3D 00 00 00 80                       cmp     eax, 80000000h
.text:100010B7 7D 01                                jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                      loc_100010B9:
.text:100010B9 EB FF                                jmp     short near ptr loc_100010B9+1
.text:100010B9                      ; ---------------------------------------------------------
.text:100010BB E0                                   db 0E0h
.text:100010BC                      ; ---------------------------------------------------------
.text:100010BC 50                                   push    eax
.text:100010BD C3                                   retn
.text:100010BD                      ; ---------------------------------------------------------
.text:100010BE 75 E8                                dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…             dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

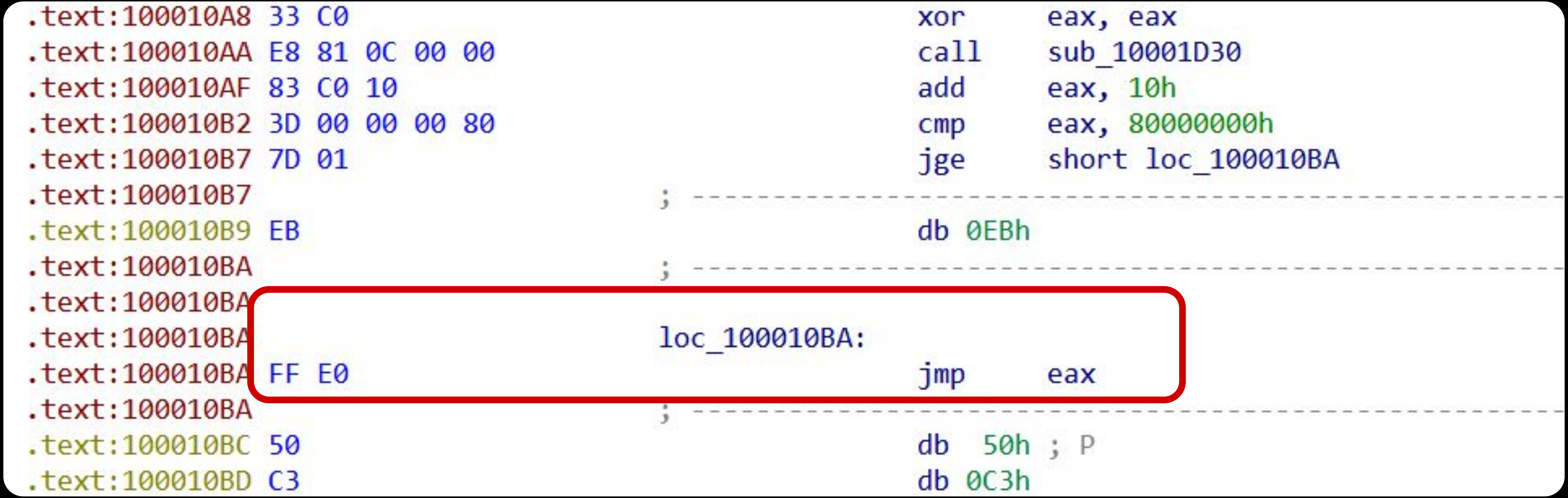


```
.text:100010A8 33 C0                                         xor     eax, eax
.text:100010AA E8 81 0C 00 00                                call    sub_10001D30
.text:100010AF 83 C0 10                                      add     eax, 10h
.text:100010B2 3D 00 00 00 80                                cmp     eax, 80000000h
.text:100010B7 7D 01                                         jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                            loc_100010B9:
.text:100010B9 EB FF                                         jmp     short near ptr loc_100010B9+1
.text:100010B9                            ; -----------------------------------------------
.text:100010BB E0                                            db 0E0h
.text:100010BC                            ; -----------------------------------------------
.text:100010BC 50                                            push    eax
.text:100010BD C3                                            retn
.text:100010BD                            ; -----------------------------------------------
.text:100010BE 75 E8                                         dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                      dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

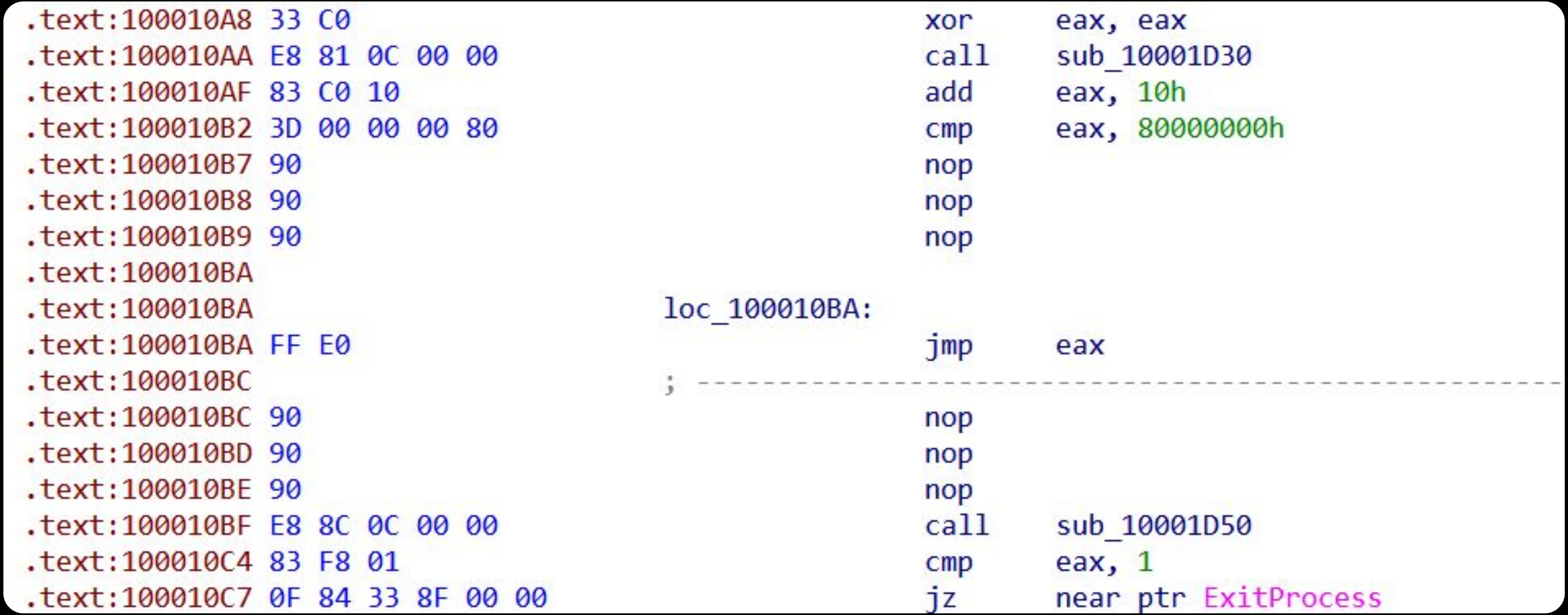


```
.text:100010A8 33 C0                                          xor     eax, eax
.text:100010AA E8 81 0C 00 00                                 call    sub_10001D30
.text:100010AF 83 C0 10                                       add     eax, 10h
.text:100010B2 3D 00 00 00 80                                 cmp     eax, 80000000h
.text:100010B7 7D 01                                          jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                            loc_100010B9:
.text:100010B9 EB FF                                          jmp     short near ptr loc_100010B9+1
.text:100010B9                            ; ---------------------------------------------------
.text:100010BB E0                                             db 0E0h
.text:100010BC                            ; ---------------------------------------------------
.text:100010BC 50                                             push    eax
.text:100010BD C3                                             retn
.text:100010BD                            ; ---------------------------------------------------
.text:100010BE 75 E8                                          dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                      dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

TDR by sekoia



```
.text:100010A8 33 C0                                      xor     eax, eax
.text:100010AA E8 81 0C 00 00                             call    sub_10001D30
.text:100010AF 83 C0 10                                   add     eax, 10h
.text:100010B2 3D 00 00 00 80                             cmp     eax, 80000000h
.text:100010B7 7D 01                                      jge     short loc_100010BA
.text:100010B7                            ; ---------------------------------------------------
.text:100010B9 EB                                         db 0EBh
.text:100010BA                            ; ---------------------------------------------------
.text:100010BA
.text:100010BA                            loc_100010BA:
.text:100010BA FF E0                                      jmp     eax
.text:100010BA                            ; ---------------------------------------------------
.text:100010BC 50                                         db  50h ; P
.text:100010BD C3                                         db 0C3h
```

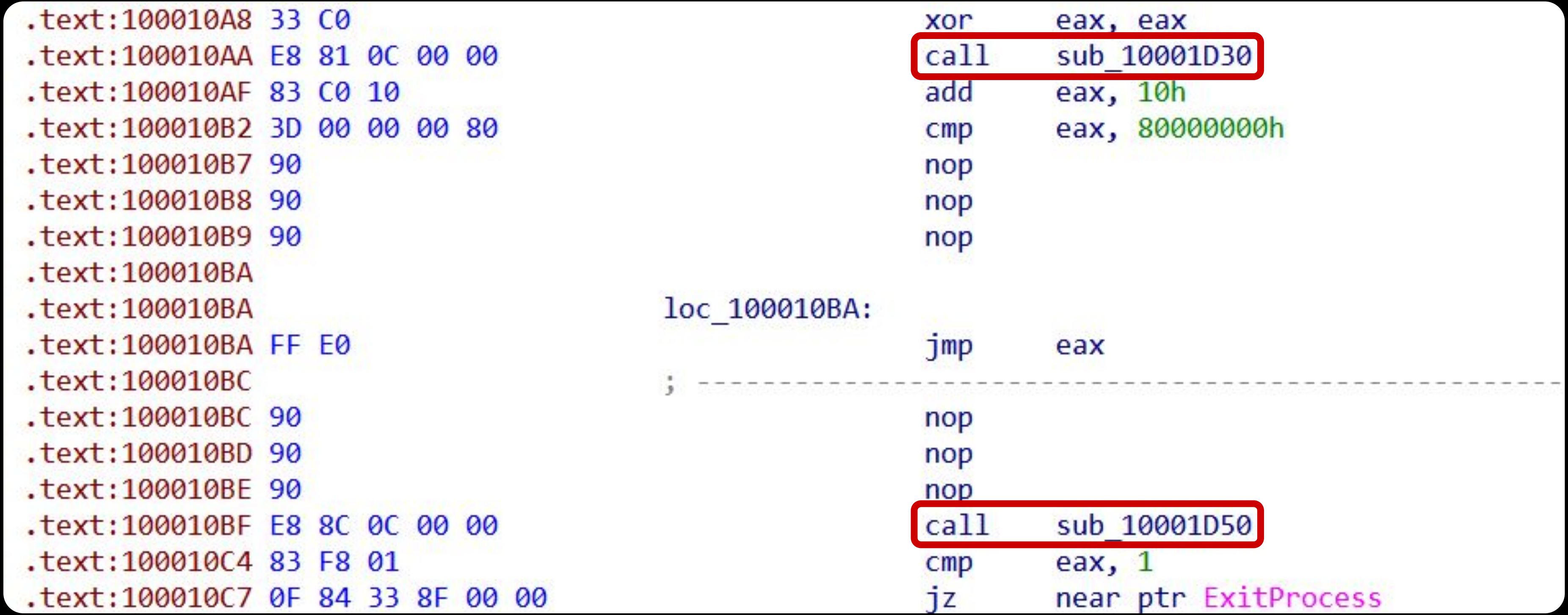


```
.text:100010A8 33 C0                                          xor     eax, eax
.text:100010AA E8 81 0C 00 00                                 call    sub_10001D30
.text:100010AF 83 C0 10                                       add     eax, 10h
.text:100010B2 3D 00 00 00 80                                 cmp     eax, 80000000h
.text:100010B7 90                                             nop
.text:100010B8 90                                             nop
.text:100010B9 90                                             nop
.text:100010BA
.text:100010BA                             loc_100010BA:
.text:100010BA FF E0                                          jmp     eax
.text:100010BC                             ; ------------------------------------
.text:100010BC 90                                             nop
.text:100010BD 90                                             nop
.text:100010BE 90                                             nop
.text:100010BF E8 8C 0C 00 00                                 call    sub_10001D50
.text:100010C4 83 F8 01                                       cmp     eax, 1
.text:100010C7 0F 84 33 8F 00 00                              jz      near ptr ExitProcess
```

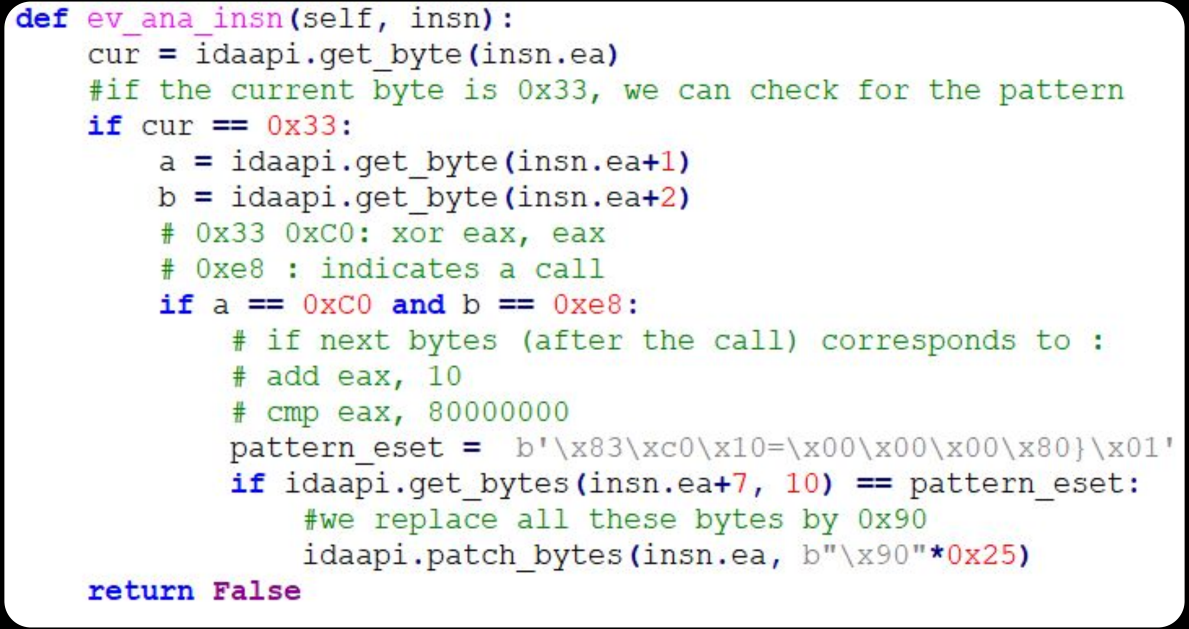
TDR by sekoia

Automatisation de la désobfuscation

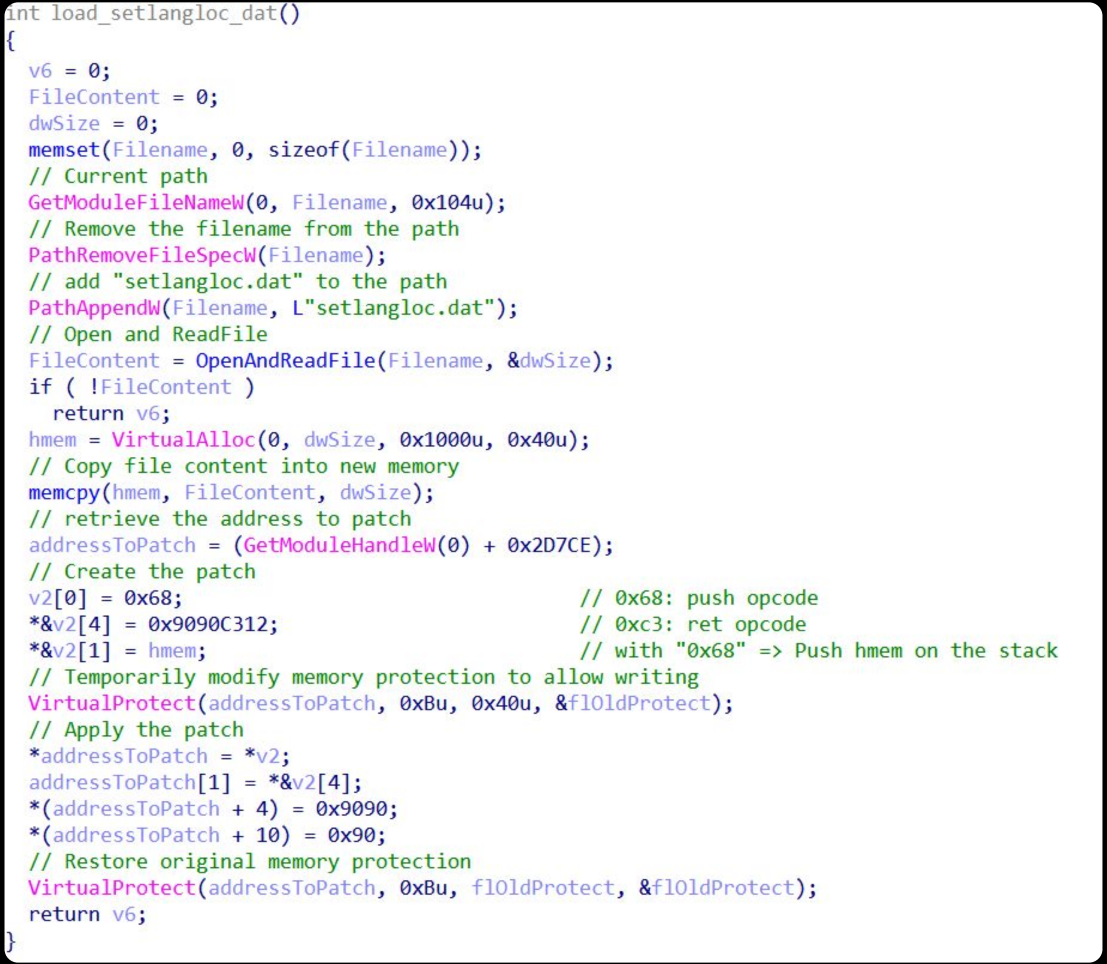
TDR by sekoia



```
.text:100010A8 33 C0                                xor     eax, eax
.text:100010AA E8 81 0C 00 00                       call    sub_10001D30
.text:100010AF 83 C0 10                             add     eax, 10h
.text:100010B2 3D 00 00 00 80                       cmp     eax, 80000000h
.text:100010B7 90                                   nop
.text:100010B8 90                                   nop
.text:100010B9 90                                   nop
.text:100010BA
.text:100010BA                      loc_100010BA:
.text:100010BA FF E0                                jmp     eax
.text:100010BC                      ; ---------------------------------------------
.text:100010BC 90                                   nop
.text:100010BD 90                                   nop
.text:100010BE 90                                   nop
.text:100010BF E8 8C 0C 00 00                       call    sub_10001D50
.text:100010C4 83 F8 01                             cmp     eax, 1
.text:100010C7 0F 84 33 8F 00 00                    jz      near ptr ExitProcess
```

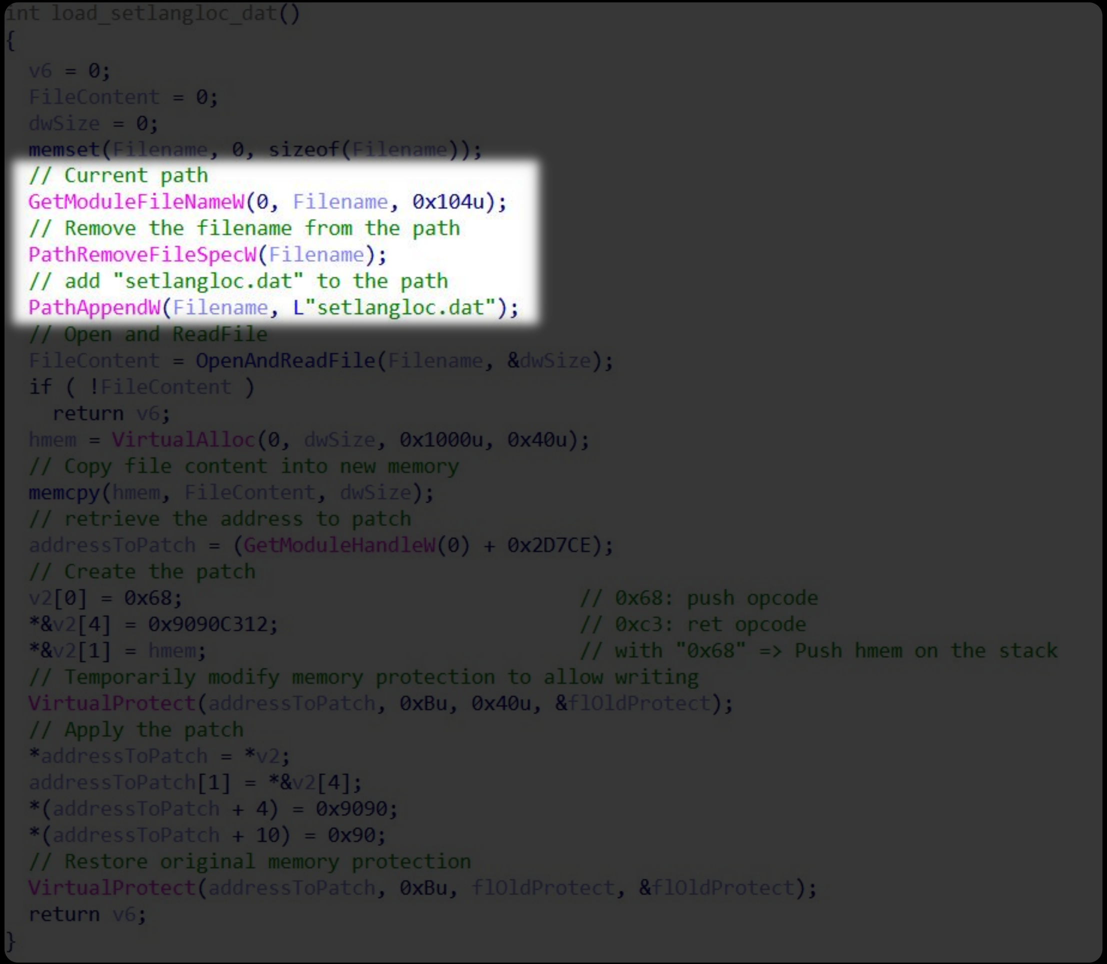


```python
def ev_ana_insn(self, insn):
    cur = idaapi.get_byte(insn.ea)
    #if the current byte is 0x33, we can check for the pattern
    if cur == 0x33:
        a = idaapi.get_byte(insn.ea+1)
        b = idaapi.get_byte(insn.ea+2)
        # 0x33 0xC0: xor eax, eax
        # 0xe8 : indicates a call
        if a == 0xC0 and b == 0xe8:
            # if next bytes (after the call) corresponds to :
            # add eax, 10
            # cmp eax, 80000000
            pattern_eset =  b'\x83\xc0\x10=\x00\x00\x00\x80}\x01'
            if idaapi.get_bytes(insn.ea+7, 10) == pattern_eset:
                #we replace all these bytes by 0x90
                idaapi.patch_bytes(insn.ea, b"\x90"*0x25)
    return False
```

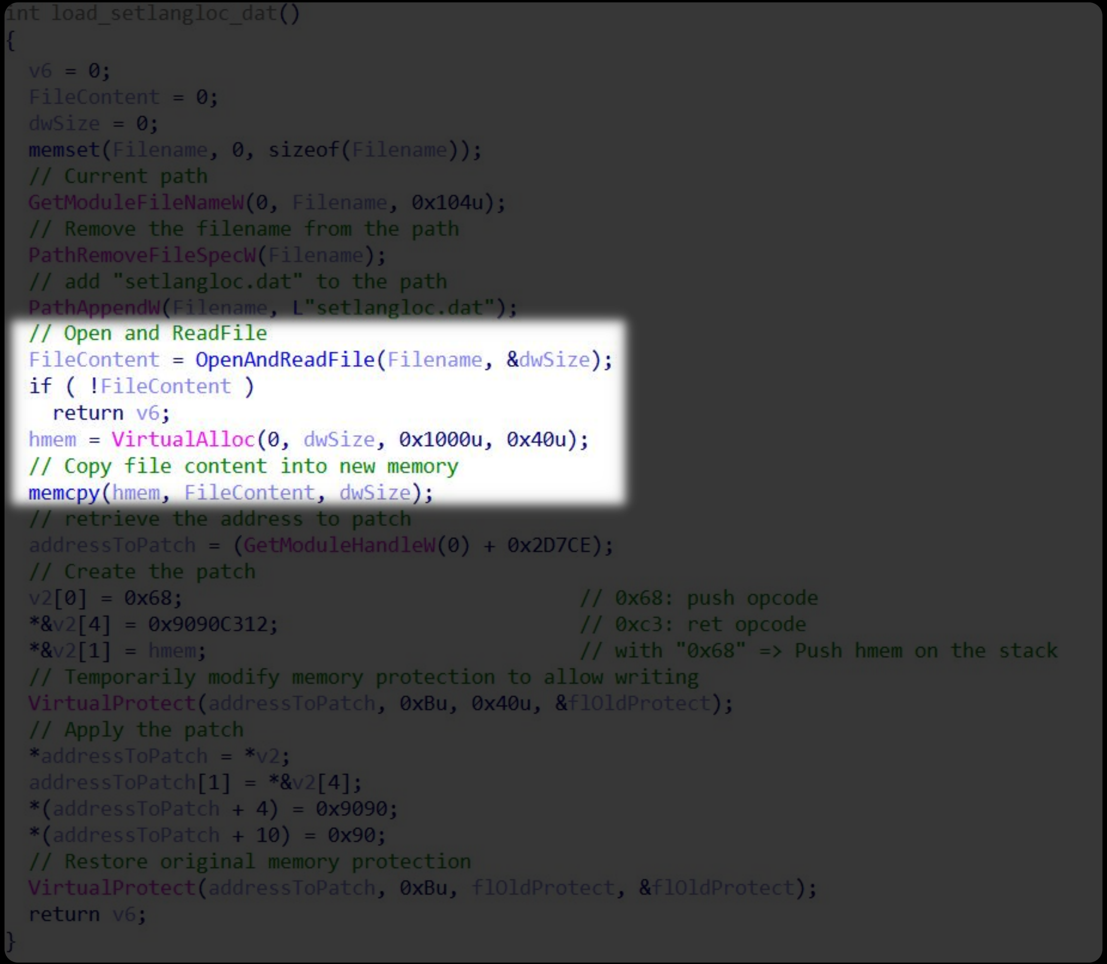


```
int load_setlangloc_dat()
{
  v6 = 0;
  FileContent = 0;
  dwSize = 0;
  memset(Filename, 0, sizeof(Filename));
  // Current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add "setlangloc.dat" to the path
  PathAppendW(Filename, L"setlangloc.dat");
  // Open and ReadFile
  FileContent = OpenAndReadFile(Filename, &dwSize);
  if ( !FileContent )
    return v6;
  hmem = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  // Copy file content into new memory
  memcpy(hmem, FileContent, dwSize);
  // retrieve the address to patch
  addressToPatch = (GetModuleHandleW(0) + 0x2D7CE);
  // Create the patch
  v2[0] = 0x68;                                   // 0x68: push opcode
  *&v2[4] = 0x9090C312;                           // 0xc3: ret opcode
  *&v2[1] = hmem;                                 // with "0x68" => Push hmem on the stack
  // Temporarily modify memory protection to allow writing
  VirtualProtect(addressToPatch, 0xBu, 0x40u, &flOldProtect);
  // Apply the patch
  *addressToPatch = *v2;
  addressToPatch[1] = *&v2[4];
  *(addressToPatch + 4) = 0x9090;
  *(addressToPatch + 10) = 0x90;
  // Restore original memory protection
  VirtualProtect(addressToPatch, 0xBu, flOldProtect, &flOldProtect);
  return v6;
}
```

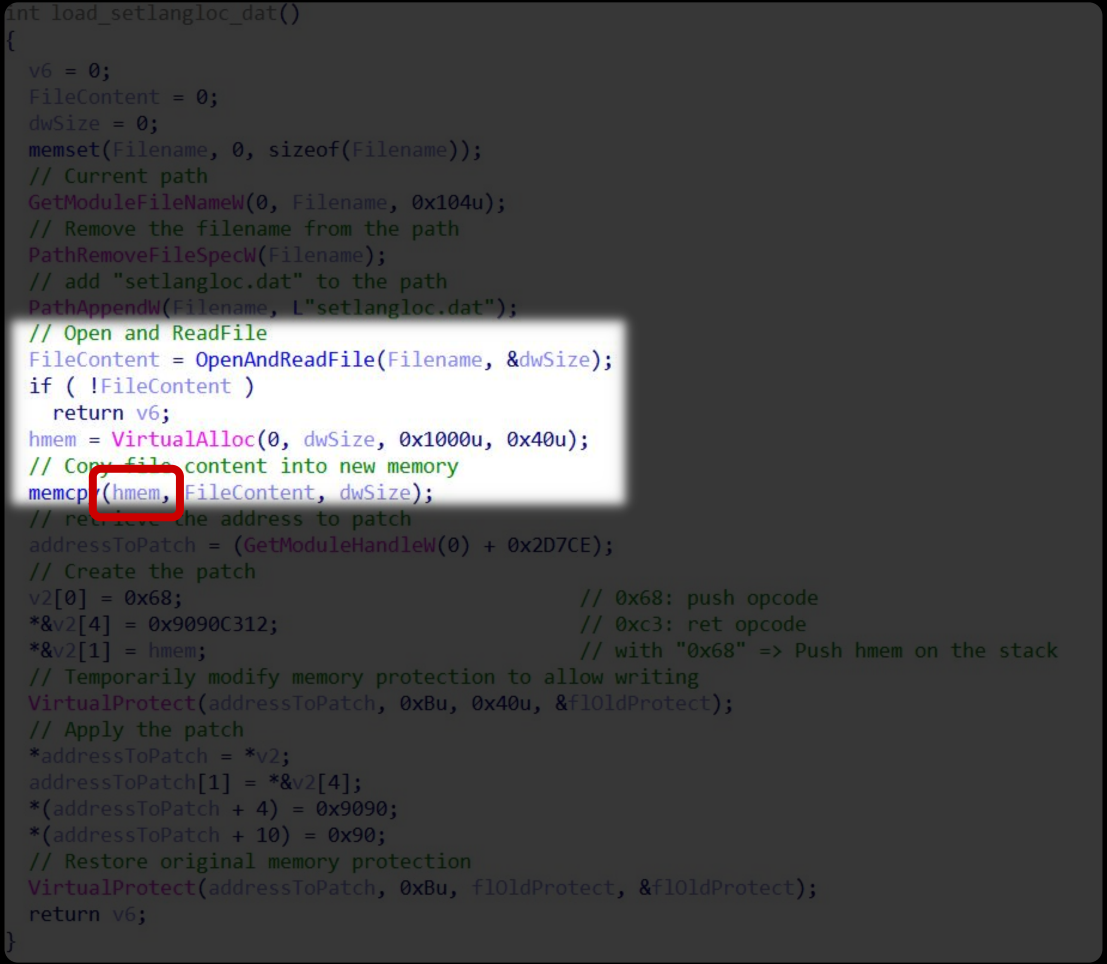
TDR by sekoia



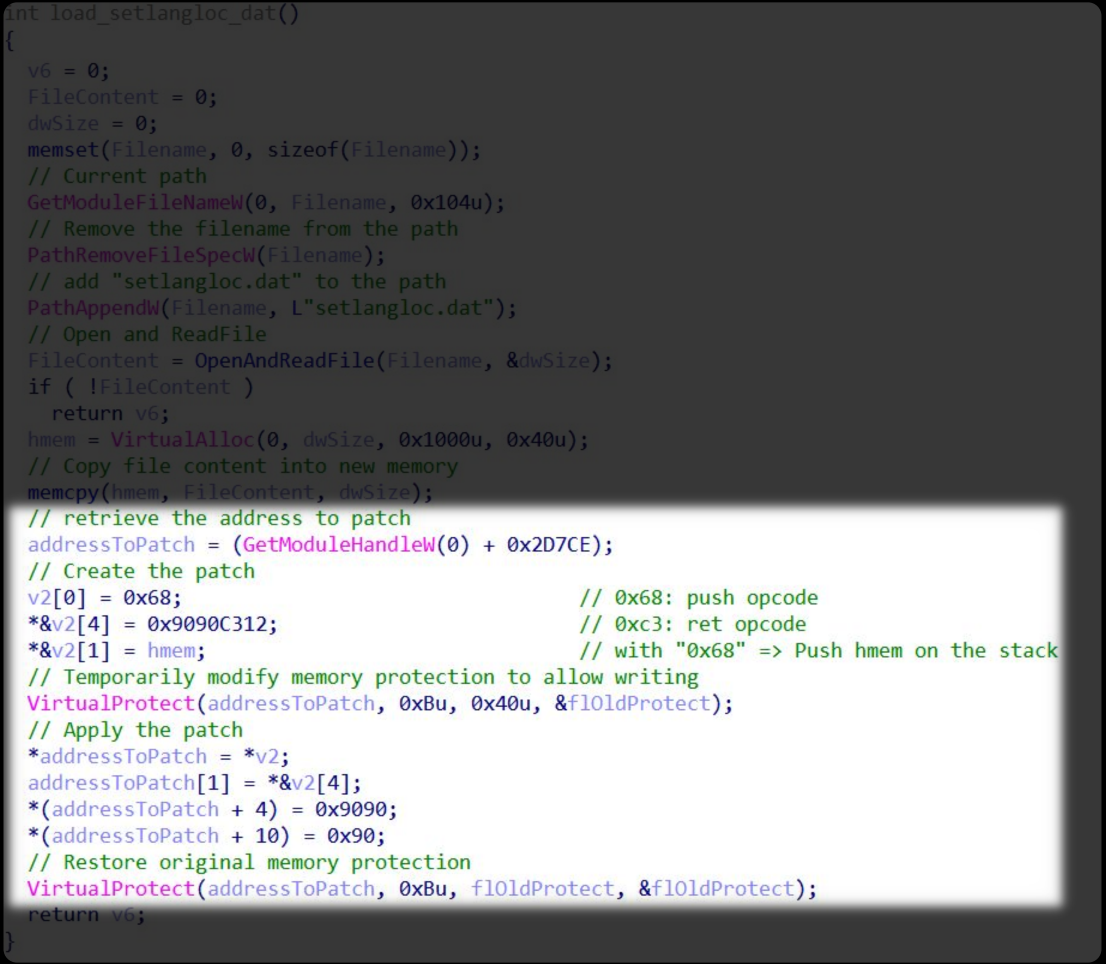
```
int load_setlangloc_dat()
{
  v6 = 0;
  FileContent = 0;
  dwSize = 0;
  memset(Filename, 0, sizeof(Filename));
  // Current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add "setlangloc.dat" to the path
  PathAppendW(Filename, L"setlangloc.dat");
  // Open and ReadFile
  FileContent = OpenAndReadFile(Filename, &dwSize);
  if ( !FileContent )
    return v6;
  hmem = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  // Copy file content into new memory
  memcpy(hmem, FileContent, dwSize);
  // retrieve the address to patch
  addressToPatch = (GetModuleHandleW(0) + 0x2D7CE);
  // Create the patch
  v2[0] = 0x68;                                     // 0x68: push opcode
  *&v2[4] = 0x9090C312;                             // 0xc3: ret opcode
  *&v2[1] = hmem;                                   // with "0x68" => Push hmem on the stack
  // Temporarily modify memory protection to allow writing
  VirtualProtect(addressToPatch, 0xBu, 0x40u, &flOldProtect);
  // Apply the patch
  *addressToPatch = *v2;
  addressToPatch[1] = *&v2[4];
  *(addressToPatch + 4) = 0x9090;
  *(addressToPatch + 10) = 0x90;
  // Restore original memory protection
  VirtualProtect(addressToPatch, 0xBu, flOldProtect, &flOldProtect);
  return v6;
}
```



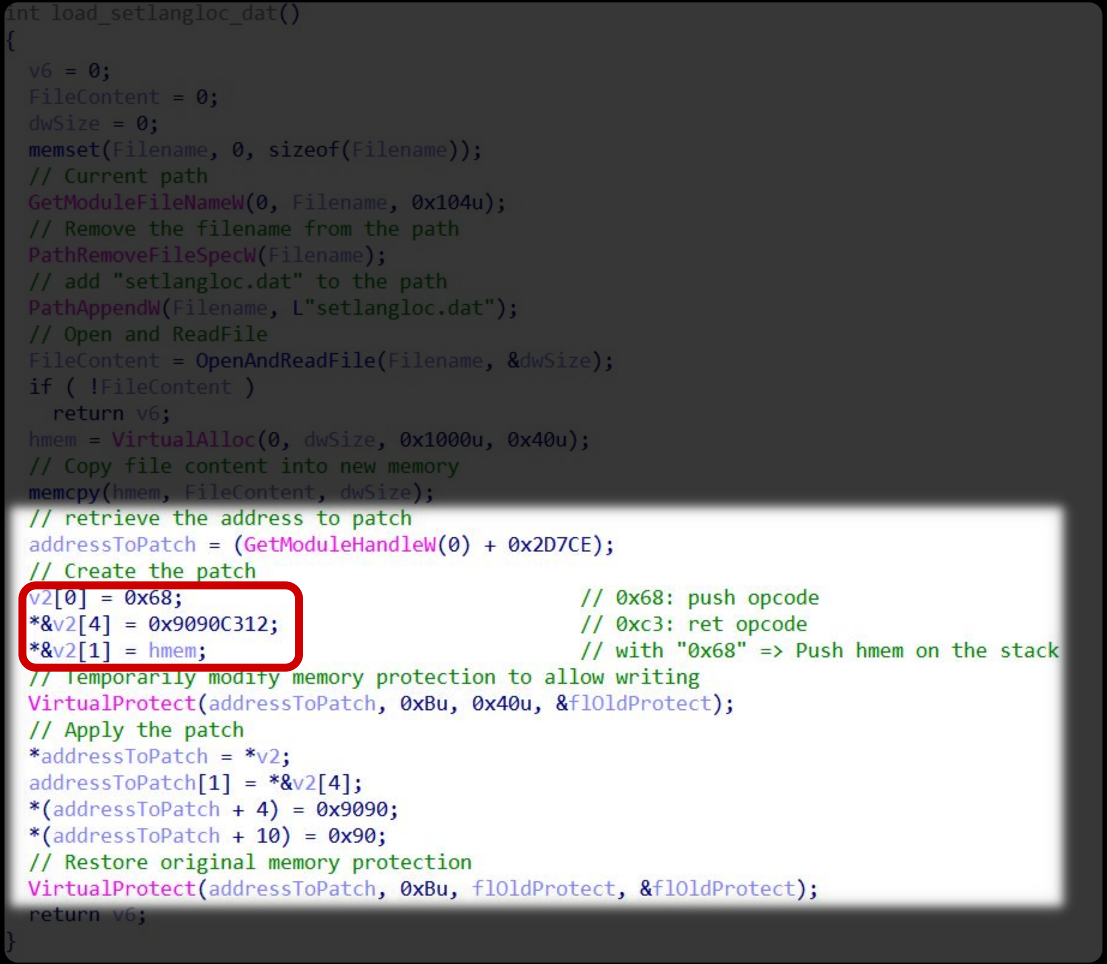
```
int load_setlangloc_dat()
{
  v6 = 0;
  FileContent = 0;
  dwSize = 0;
  memset(Filename, 0, sizeof(Filename));
  // Current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add "setlangloc.dat" to the path
  PathAppendW(Filename, L"setlangloc.dat");
  // Open and ReadFile
  FileContent = OpenAndReadFile(Filename, &dwSize);
  if ( !FileContent )
    return v6;
  hmem = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  // Copy file content into new memory
  memcpy(hmem, FileContent, dwSize);
  // retrieve the address to patch
  addressToPatch = (GetModuleHandleW(0) + 0x2D7CE);
  // Create the patch
  v2[0] = 0x68;                                   // 0x68: push opcode
  *&v2[4] = 0x9090C312;                           // 0xc3: ret opcode
  *&v2[1] = hmem;                                 // with "0x68" => Push hmem on the stack
  // Temporarily modify memory protection to allow writing
  VirtualProtect(addressToPatch, 0xBu, 0x40u, &flOldProtect);
  // Apply the patch
  *addressToPatch = *v2;
  addressToPatch[1] = *&v2[4];
  *(addressToPatch + 4) = 0x9090;
  *(addressToPatch + 10) = 0x90;
  // Restore original memory protection
  VirtualProtect(addressToPatch, 0xBu, flOldProtect, &flOldProtect);
  return v6;
}
```



```
int load_setlangloc_dat()
{
  v6 = 0;
  FileContent = 0;
  dwSize = 0;
  memset(Filename, 0, sizeof(Filename));
  // Current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add "setlangloc.dat" to the path
  PathAppendW(Filename, L"setlangloc.dat");
  // Open and ReadFile
  FileContent = OpenAndReadFile(Filename, &dwSize);
  if ( !FileContent )
    return v6;
  hmem = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  // Copy file content into new memory
  memcpy(hmem, FileContent, dwSize);
  // retrieve the address to patch
  addressToPatch = (GetModuleHandleW(0) + 0x2D7CE);
  // Create the patch
  v2[0] = 0x68;                                    // 0x68: push opcode
  *&v2[4] = 0x9090C312;                            // 0xc3: ret opcode
  *&v2[1] = hmem;                                  // with "0x68" => Push hmem on the stack
  // Temporarily modify memory protection to allow writing
  VirtualProtect(addressToPatch, 0xBu, 0x40u, &flOldProtect);
  // Apply the patch
  *addressToPatch = *v2;
  addressToPatch[1] = *&v2[4];
  *(addressToPatch + 4) = 0x9090;
  *(addressToPatch + 10) = 0x90;
  // Restore original memory protection
  VirtualProtect(addressToPatch, 0xBu, flOldProtect, &flOldProtect);
  return v6;
}
```



```
int load_setlangloc_dat()
{
  v6 = 0;
  FileContent = 0;
  dwSize = 0;
  memset(Filename, 0, sizeof(Filename));
  // Current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add "setlangloc.dat" to the path
  PathAppendW(Filename, L"setlangloc.dat");
  // Open and ReadFile
  FileContent = OpenAndReadFile(Filename, &dwSize);
  if ( !FileContent )
    return v6;
  hmem = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  // Copy file content into new memory
  memcpy(hmem, FileContent, dwSize);
  // retrieve the address to patch
  addressToPatch = (GetModuleHandleW(0) + 0x2D7CE);
  // Create the patch
  v2[0] = 0x68;                                   // 0x68: push opcode
  *&v2[4] = 0x9090C312;                           // 0xc3: ret opcode
  *&v2[1] = hmem;                                 // with "0x68" => Push hmem on the stack
  // Temporarily modify memory protection to allow writing
  VirtualProtect(addressToPatch, 0xBu, 0x40u, &flOldProtect);
  // Apply the patch
  *addressToPatch = *v2;
  addressToPatch[1] = *&v2[4];
  *(addressToPatch + 4) = 0x9090;
  *(addressToPatch + 10) = 0x90;
  // Restore original memory protection
  VirtualProtect(addressToPatch, 0xBu, flOldProtect, &flOldProtect);
  return v6;
}
```



```
int load_setlangloc_dat()
{
  v6 = 0;
  FileContent = 0;
  dwSize = 0;
  memset(Filename, 0, sizeof(Filename));
  // Current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add "setlangloc.dat" to the path
  PathAppendW(Filename, L"setlangloc.dat");
  // Open and ReadFile
  FileContent = OpenAndReadFile(Filename, &dwSize);
  if ( !FileContent )
    return v6;
  hmem = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  // Copy file content into new memory
  memcpy(hmem, FileContent, dwSize);
  // retrieve the address to patch
  addressToPatch = (GetModuleHandleW(0) + 0x2D7CE);
  // Create the patch
  v2[0] = 0x68;                                  // 0x68: push opcode
  *&v2[4] = 0x9090C312;                          // 0xc3: ret opcode
  *&v2[1] = hmem;                                // with "0x68" => Push hmem on the stack
  // Temporarily modify memory protection to allow writing
  VirtualProtect(addressToPatch, 0xBu, 0x40u, &flOldProtect);
  // Apply the patch
  *addressToPatch = *v2;
  addressToPatch[1] = *&v2[4];
  *(addressToPatch + 4) = 0x9090;
  *(addressToPatch + 10) = 0x90;
  // Restore original memory protection
  VirtualProtect(addressToPatch, 0xBu, flOldProtect, &flOldProtect);
  return v6;
}
```

Bilan

> Identification du mécanisme anti-analyse
> La DLL charge puis exécute “setlangloc.dat”

Problème :

> “setlangloc.dat” introuvable

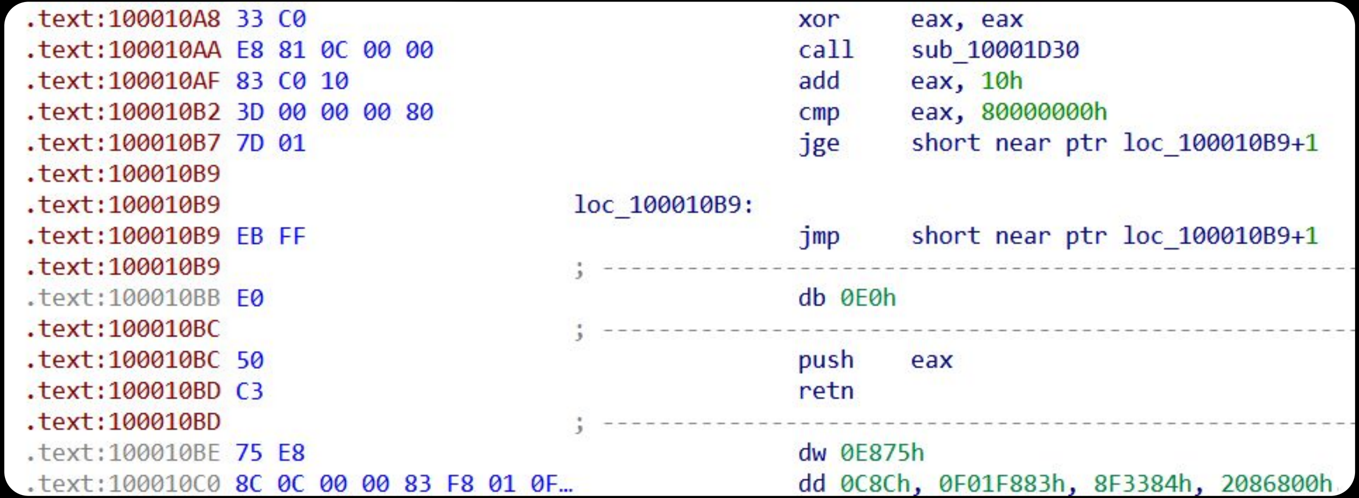
Signature du code

TDR by sekoia

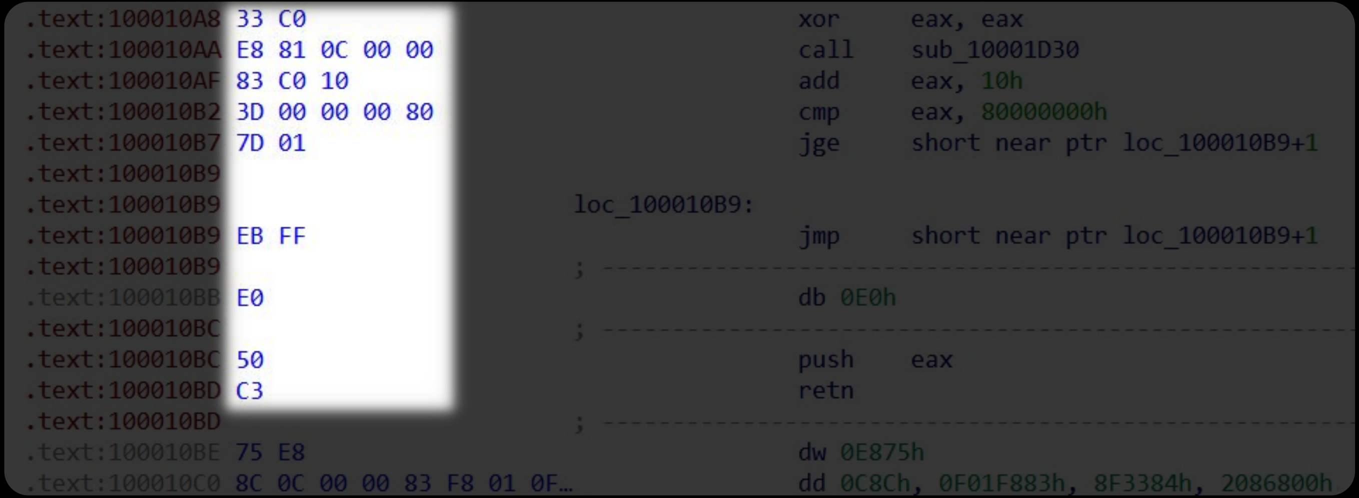
Création d’une règle YARA

> Langage de création de règles / signatures
> Identification des patterns dans des fichiers
> Utilisé pour la détection de malwares

Création d’une règle YARA

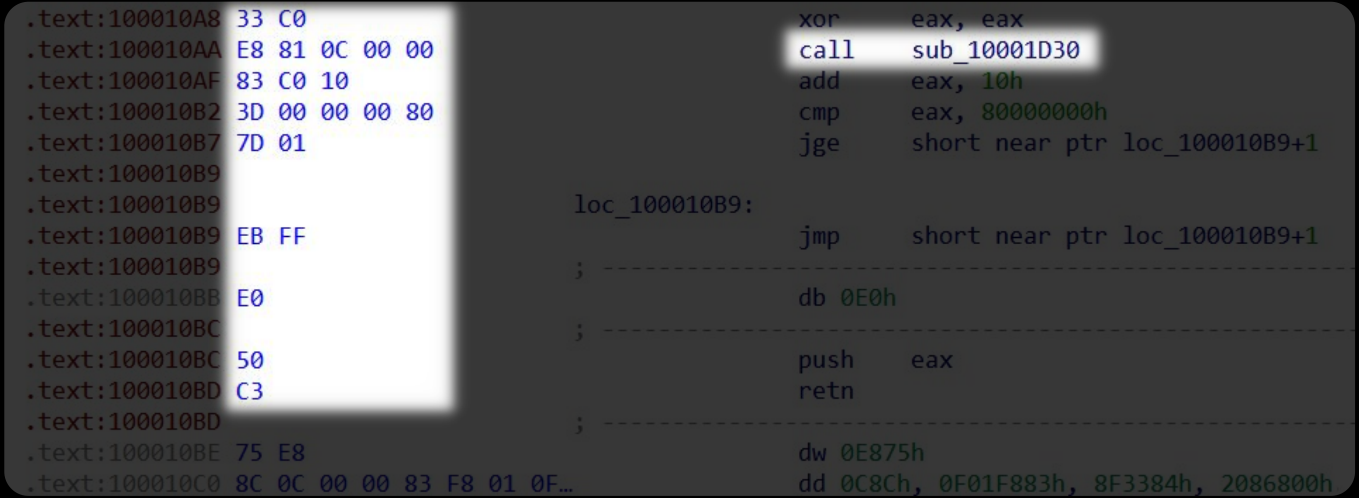

```
.text:100010A8 33 C0                                     xor     eax, eax
.text:100010AA E8 81 0C 00 00                            call    sub_10001D30
.text:100010AF 83 C0 10                                  add     eax, 10h
.text:100010B2 3D 00 00 00 80                            cmp     eax, 80000000h
.text:100010B7 7D 01                                     jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                           loc_100010B9:
.text:100010B9 EB FF                                     jmp     short near ptr loc_100010B9+1
.text:100010B9                           ; ---------------------------------------------------------
.text:100010BB E0                                        db 0E0h
.text:100010BC                           ; ---------------------------------------------------------
.text:100010BC 50                                        push    eax
.text:100010BD C3                                        retn
.text:100010BD                           ; ---------------------------------------------------------
.text:100010BE 75 E8                                     dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                  dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

Création d’une règle YARA



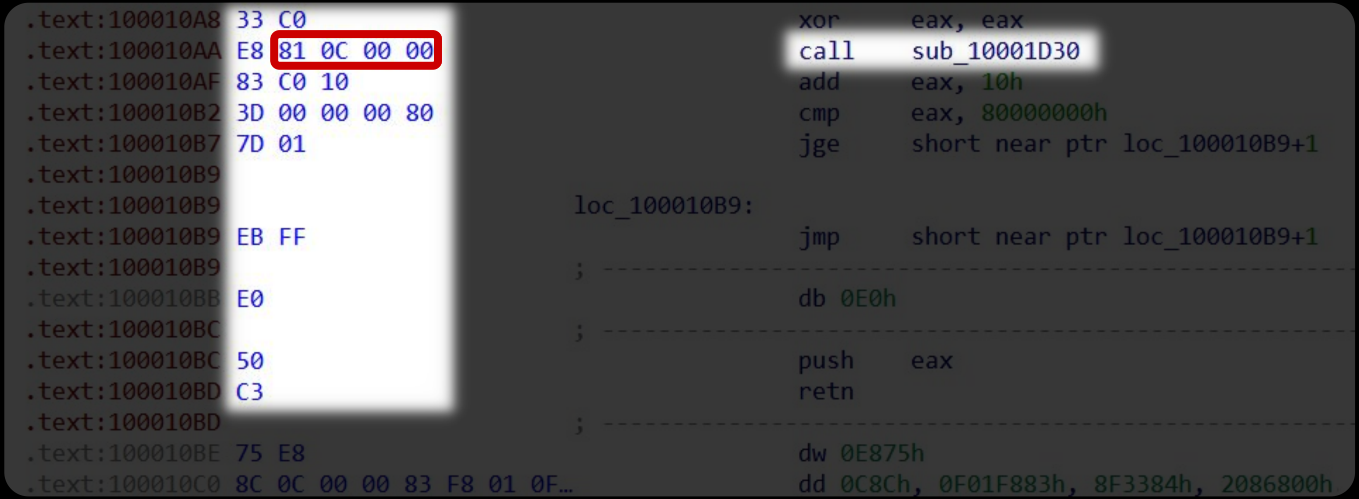
```
.text:100010A8 33 C0                                     xor     eax, eax
.text:100010AA E8 81 0C 00 00                            call    sub_10001D30
.text:100010AF 83 C0 10                                  add     eax, 10h
.text:100010B2 3D 00 00 00 80                            cmp     eax, 80000000h
.text:100010B7 7D 01                                     jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                               loc_100010B9:
.text:100010B9 EB FF                                     jmp     short near ptr loc_100010B9+1
.text:100010B9                               ; ---------------------------------------------------
.text:100010BB E0                                        db 0E0h
.text:100010BC                               ; ---------------------------------------------------
.text:100010BC 50                                        push    eax
.text:100010BD C3                                        retn
.text:100010BD                               ; ---------------------------------------------------
.text:100010BE 75 E8                                     dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                  dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

Création d'une règle YARA



```
.text:100010A8 33 C0                                        xor     eax, eax
.text:100010AA E8 81 0C 00 00                               call    sub_10001D30
.text:100010AF 83 C0 10                                     add     eax, 10h
.text:100010B2 3D 00 00 00 80                               cmp     eax, 80000000h
.text:100010B7 7D 01                                        jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                              loc_100010B9:
.text:100010B9 EB FF                                        jmp     short near ptr loc_100010B9+1
.text:100010B9                              ; ---------------------------------------------------
.text:100010BB E0                                           db 0E0h
.text:100010BC                              ; ---------------------------------------------------
.text:100010BC 50                                           push    eax
.text:100010BD C3                                           retn
.text:100010BD                              ; ---------------------------------------------------
.text:100010BE 75 E8                                        dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                     dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

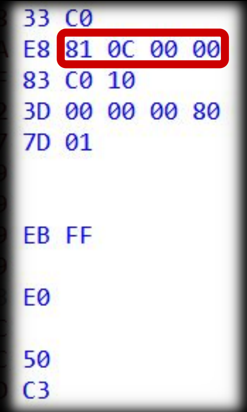
Création d’une règle YARA

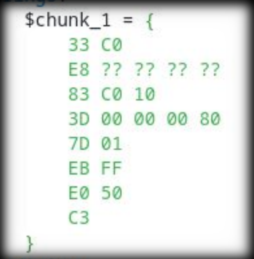


```
.text:100010A8 33 C0                                         xor     eax, eax
.text:100010AA E8 81 0C 00 00                                call    sub_10001D30
.text:100010AF 83 C0 10                                      add     eax, 10h
.text:100010B2 3D 00 00 00 80                                cmp     eax, 80000000h
.text:100010B7 7D 01                                         jge     short near ptr loc_100010B9+1
.text:100010B9
.text:100010B9                               loc_100010B9:
.text:100010B9 EB FF                                         jmp     short near ptr loc_100010B9+1
.text:100010B9                               ; ---------------------------------------------------
.text:100010BB E0                                            db 0E0h
.text:100010BC                               ; ---------------------------------------------------
.text:100010BC 50                                            push    eax
.text:100010BD C3                                            retn
.text:100010BD                               ; ---------------------------------------------------
.text:100010BE 75 E8                                         dw 0E875h
.text:100010C0 8C 0C 00 00 83 F8 01 0F…                      dd 0C8Ch, 0F01F883h, 8F3384h, 2086800h
```

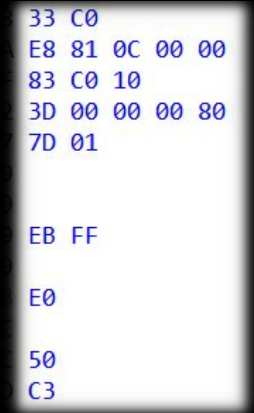
TDR by sekoia

Création d'une règle YARA

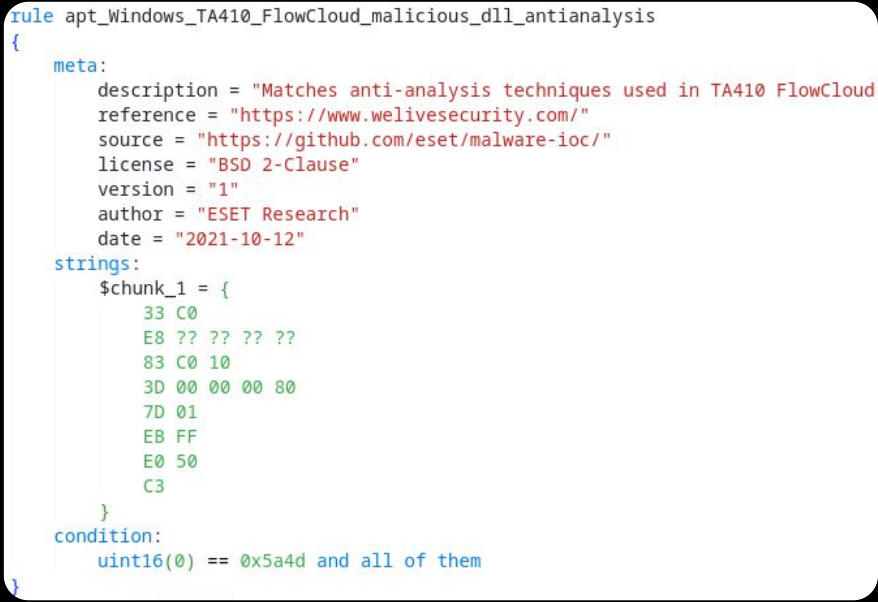

```
33 C0
E8 81 0C 00 00
83 C0 10
3D 00 00 00 80
7D 01


EB FF

E0

50
C3
```


```
$chunk_1 = {
    33 C0
    E8 ?? ?? ?? ??
    83 C0 10
    3D 00 00 00 80
    7D 01
    EB FF
    E0 50
    C3
}
```

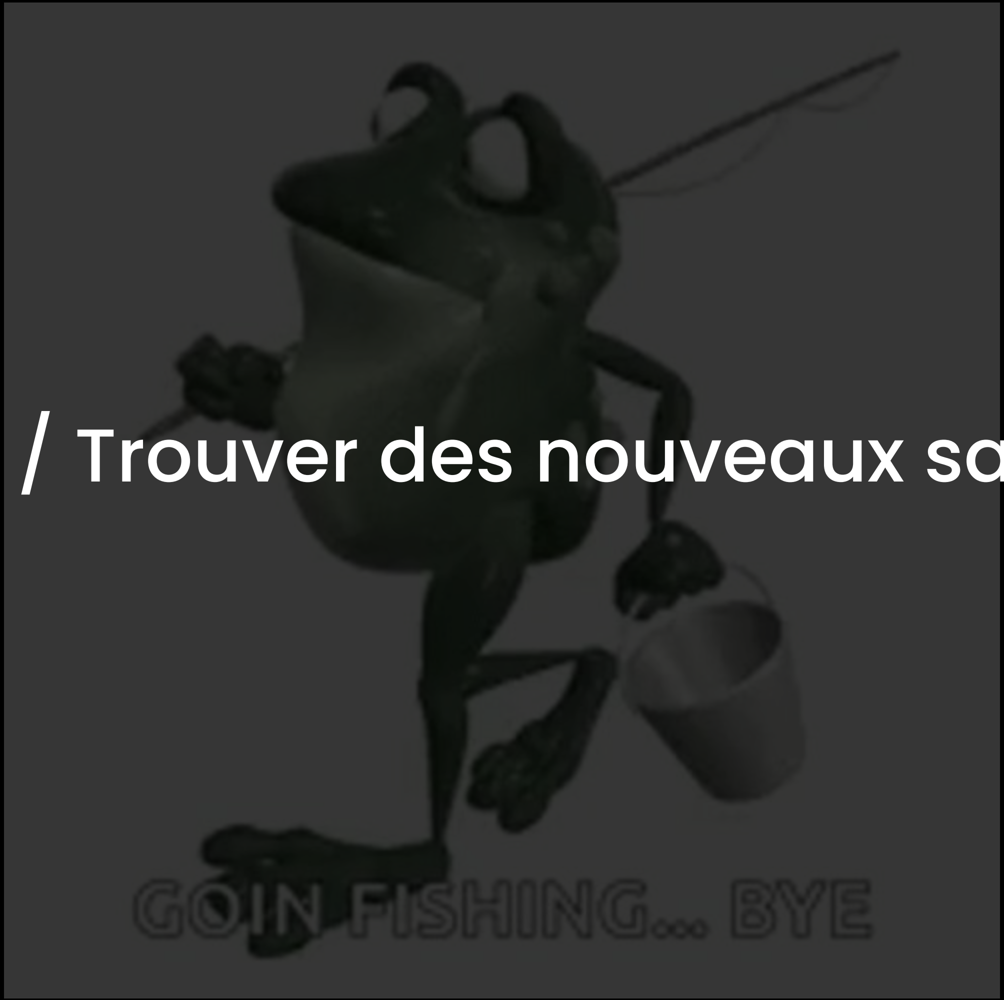
Création d’une règle YARA



```
33 C0
E8 81 0C 00 00
83 C0 10
3D 00 00 00 80
7D 01


EB FF


E0


50
C3
```

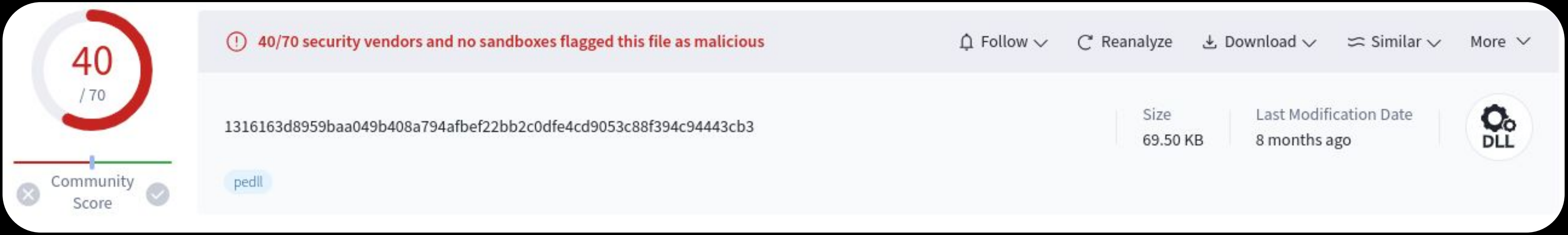


```
rule apt_Windows_TA410_FlowCloud_malicious_dll_antianalysis
{
    meta:
        description = "Matches anti-analysis techniques used in TA410 FlowCloud
        reference = "https://www.welivesecurity.com/"
        source = "https://github.com/eset/malware-ioc/"
        license = "BSD 2-Clause"
        version = "1"
        author = "ESET Research"
        date = "2021-10-12"
    strings:
        $chunk_1 = {
            33 C0
            E8 ?? ?? ?? ??
            83 C0 10
            3D 00 00 00 80
            7D 01
            EB FF
            E0 50
            C3
        }
    condition:
        uint16(0) == 0x5a4d and all of them
}
```



Pivoter / Trouver des nouveaux samples

Deux variants connus sur VirusTotal

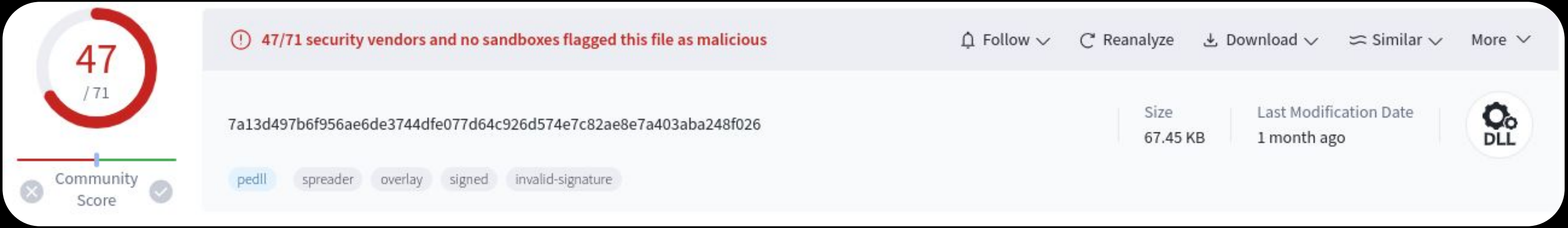


40/70 security vendors and no sandboxes flagged this file as malicious

1316163d8959baa049b408a794afbef22bb2c0dfe4cd9053c88f394c94443cb3

pedll

Size	Last Modification Date
69.50 KB	8 months ago



47/71 security vendors and no sandboxes flagged this file as malicious

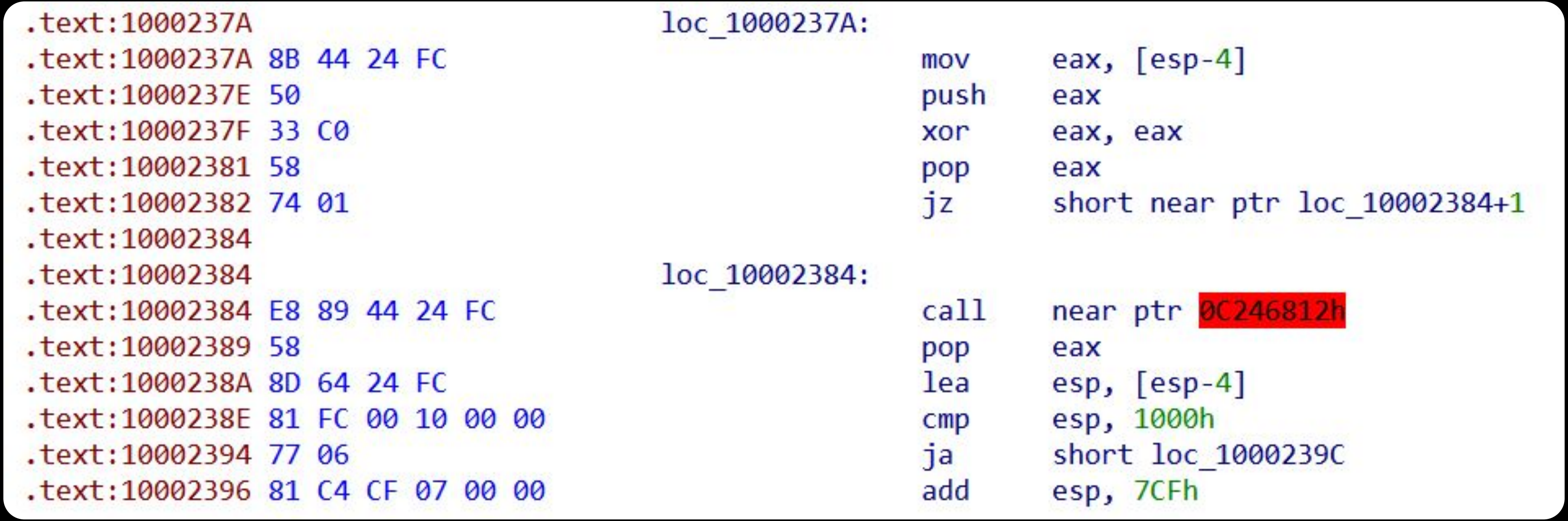
7a13d497b6f956ae6de3744dfe077d64c926d574e7c82ae8e7a403aba248f026

pedll spreader overlay signed invalid-signature

Size	Last Modification Date
67.45 KB	1 month ago

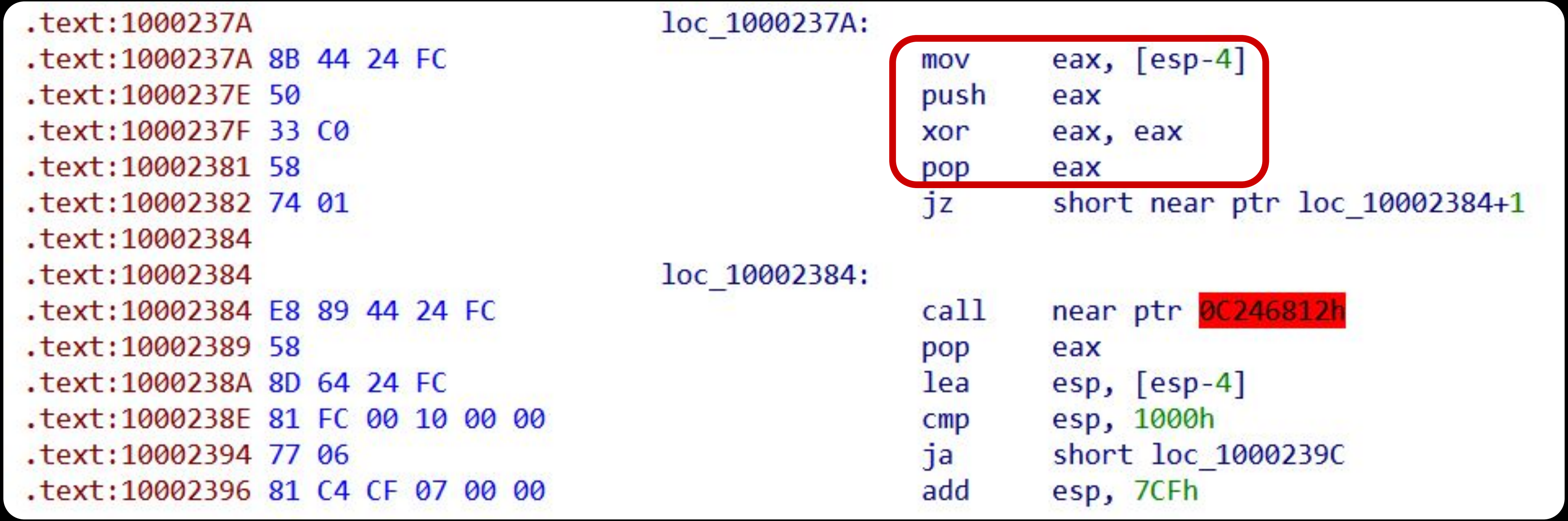
Des nouveaux mécanismes d’obfuscation

Jump in the middle



```
.text:1000237A                         loc_1000237A:
.text:1000237A 8B 44 24 FC                                  mov     eax, [esp-4]
.text:1000237E 50                                           push    eax
.text:1000237F 33 C0                                        xor     eax, eax
.text:10002381 58                                           pop     eax
.text:10002382 74 01                                        jz      short near ptr loc_10002384+1
.text:10002384
.text:10002384                         loc_10002384:
.text:10002384 E8 89 44 24 FC                               call    near ptr 0C246812h
.text:10002389 58                                           pop     eax
.text:1000238A 8D 64 24 FC                                  lea     esp, [esp-4]
.text:1000238E 81 FC 00 10 00 00                            cmp     esp, 1000h
.text:10002394 77 06                                        ja      short loc_1000239C
.text:10002396 81 C4 CF 07 00 00                            add     esp, 7CFh
```

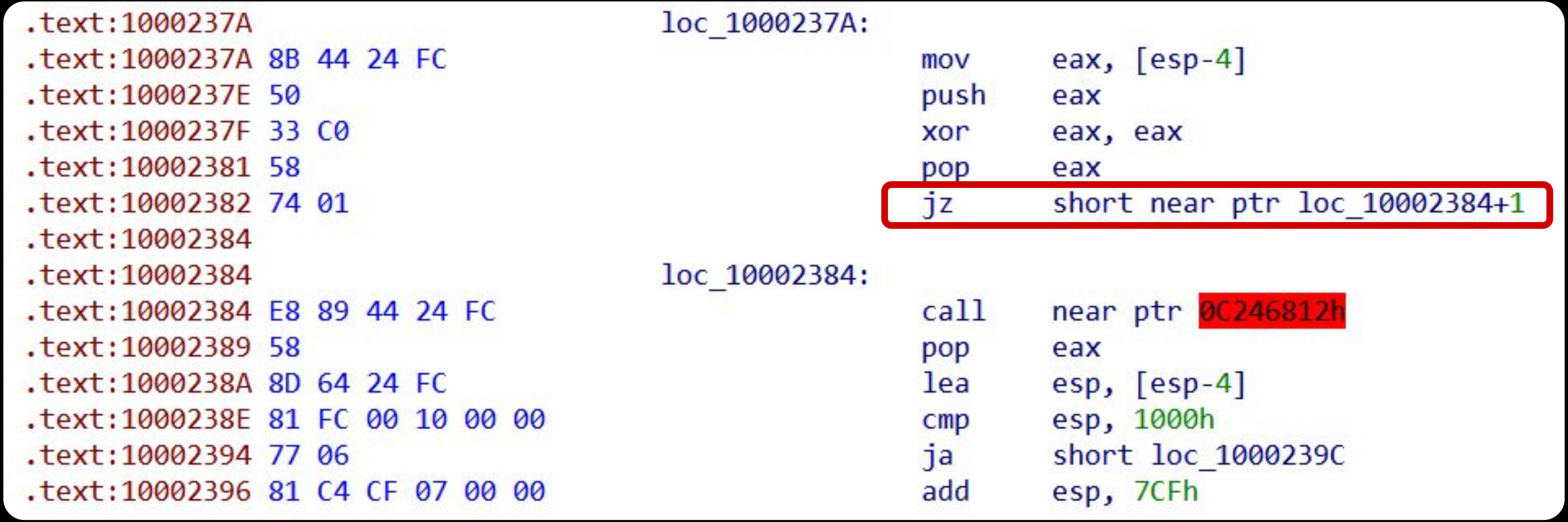
Jump in the middle



```
.text:1000237A                          loc_1000237A:
.text:1000237A 8B 44 24 FC                           mov     eax, [esp-4]
.text:1000237E 50                                    push    eax
.text:1000237F 33 C0                                 xor     eax, eax
.text:10002381 58                                    pop     eax
.text:10002382 74 01                                 jz      short near ptr loc_10002384+1
.text:10002384
.text:10002384                          loc_10002384:
.text:10002384 E8 89 44 24 FC                        call    near ptr 0C246812h
.text:10002389 58                                    pop     eax
.text:1000238A 8D 64 24 FC                           lea     esp, [esp-4]
.text:1000238E 81 FC 00 10 00 00                     cmp     esp, 1000h
.text:10002394 77 06                                 ja      short loc_1000239C
.text:10002396 81 C4 CF 07 00 00                     add     esp, 7CFh
```

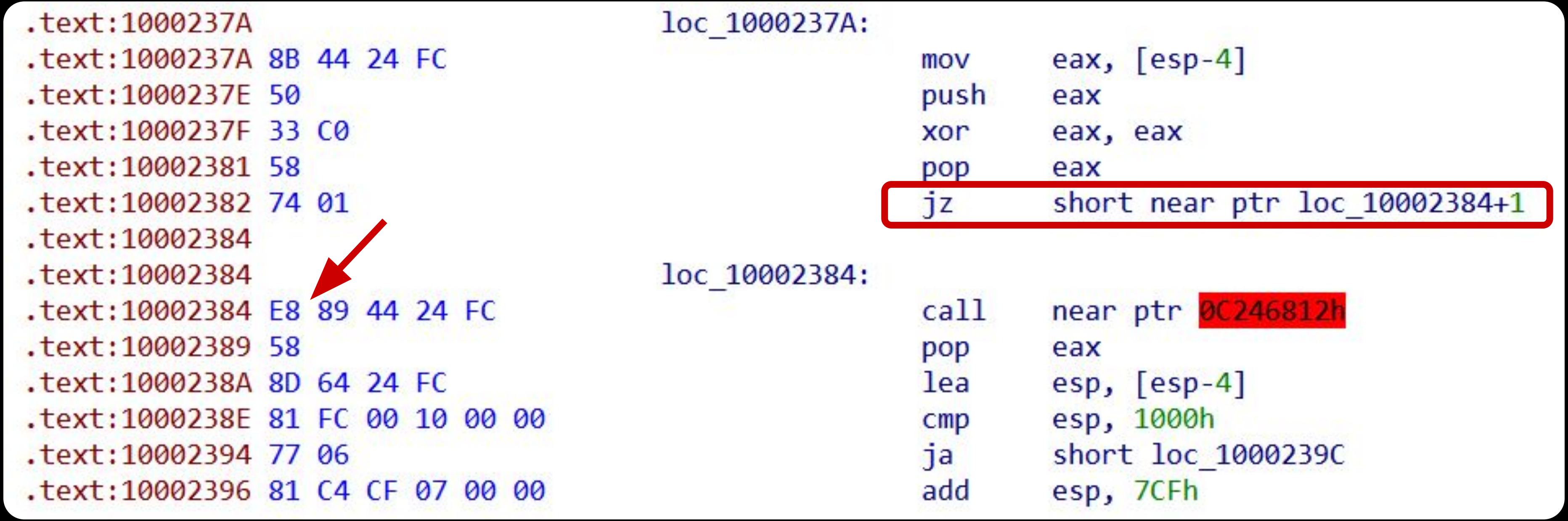
TDR by sekoia

Jump in the middle



```
.text:1000237A                           loc_1000237A:
.text:1000237A 8B 44 24 FC                                mov     eax, [esp-4]
.text:1000237E 50                                         push    eax
.text:1000237F 33 C0                                      xor     eax, eax
.text:10002381 58                                         pop     eax
.text:10002382 74 01                                      jz      short near ptr loc_10002384+1
.text:10002384
.text:10002384                           loc_10002384:
.text:10002384 E8 89 44 24 FC                             call    near ptr 0C246812h
.text:10002389 58                                         pop     eax
.text:1000238A 8D 64 24 FC                                lea     esp, [esp-4]
.text:1000238E 81 FC 00 10 00 00                          cmp     esp, 1000h
.text:10002394 77 06                                      ja      short loc_1000239C
.text:10002396 81 C4 CF 07 00 00                          add     esp, 7CFh
```

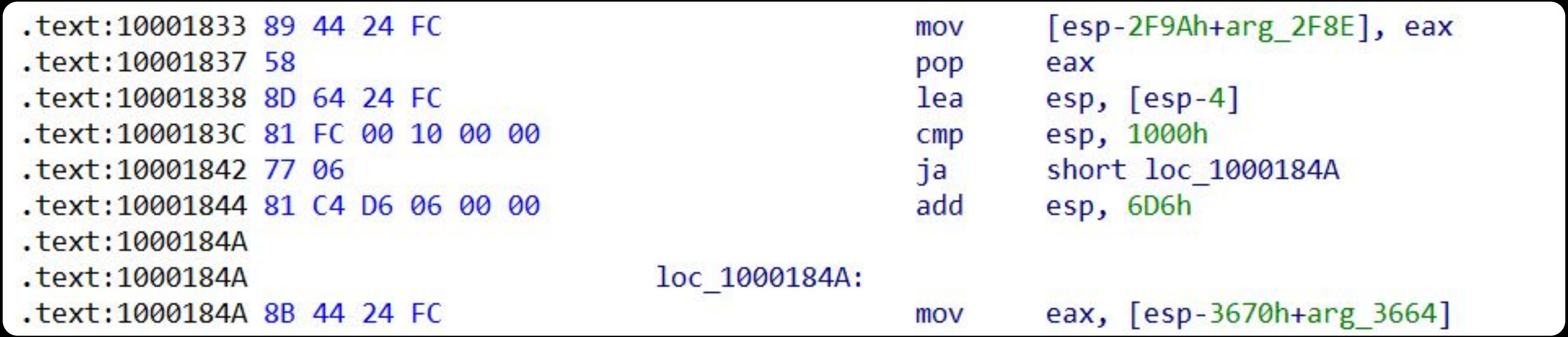
Jump in the middle



```
.text:1000237A                          loc_1000237A:
.text:1000237A 8B 44 24 FC                              mov     eax, [esp-4]
.text:1000237E 50                                       push    eax
.text:1000237F 33 C0                                    xor     eax, eax
.text:10002381 58                                       pop     eax
.text:10002382 74 01                                    jz      short near ptr loc_10002384+1
.text:10002384
.text:10002384                          loc_10002384:
.text:10002384 E8 89 44 24 FC                           call    near ptr 0C246812h
.text:10002389 58                                       pop     eax
.text:1000238A 8D 64 24 FC                              lea     esp, [esp-4]
.text:1000238E 81 FC 00 10 00 00                        cmp     esp, 1000h
.text:10002394 77 06                                    ja      short loc_1000239C
.text:10002396 81 C4 CF 07 00 00                        add     esp, 7CFh
```

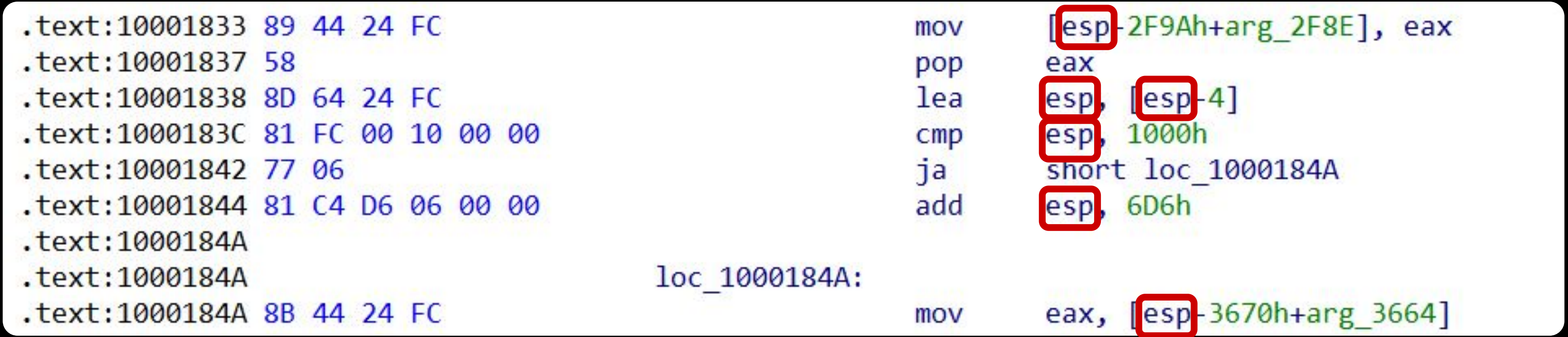
TDR by sekoia

Création de multiples variables



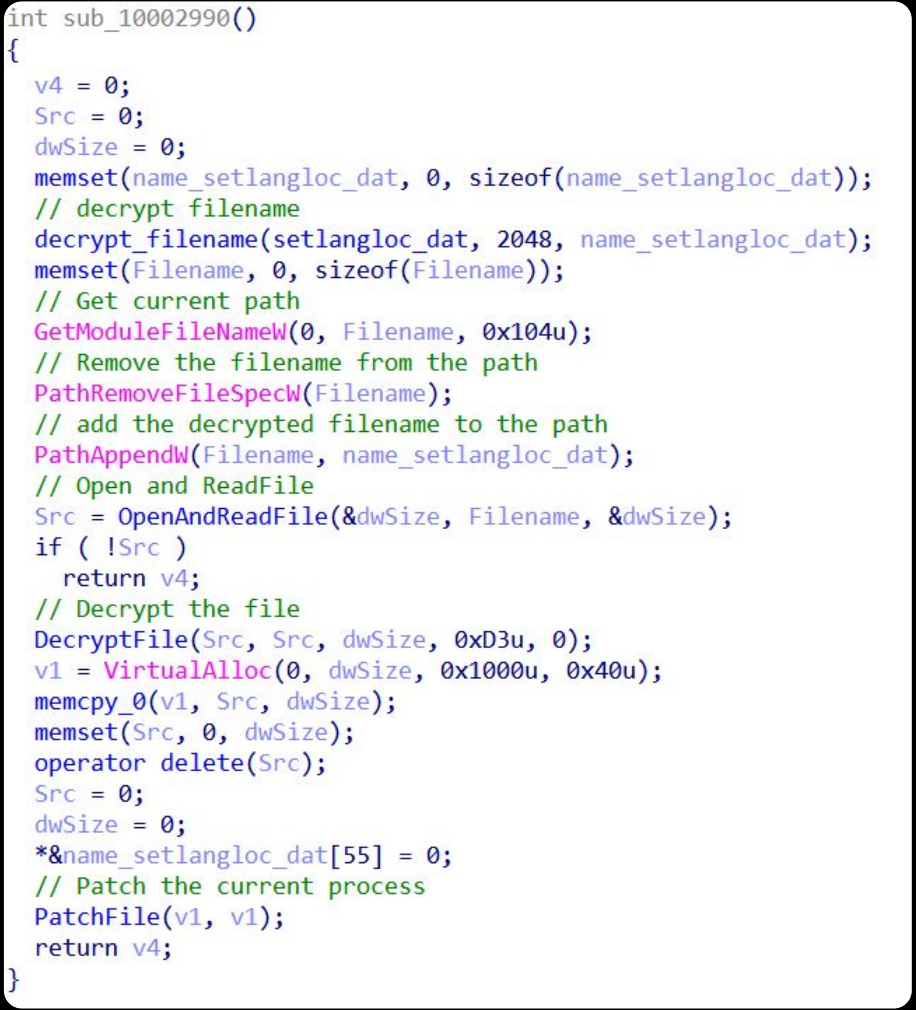
```
.text:10001833 89 44 24 FC                          mov     [esp-2F9Ah+arg_2F8E], eax
.text:10001837 58                                   pop     eax
.text:10001838 8D 64 24 FC                          lea     esp, [esp-4]
.text:1000183C 81 FC 00 10 00 00                    cmp     esp, 1000h
.text:10001842 77 06                                ja      short loc_1000184A
.text:10001844 81 C4 D6 06 00 00                    add     esp, 6D6h
.text:1000184A
.text:1000184A                       loc_1000184A:
.text:1000184A 8B 44 24 FC                          mov     eax, [esp-3670h+arg_3664]
```

Création de multiples variables



```
.text:10001833 89 44 24 FC                          mov     [esp-2F9Ah+arg_2F8E], eax
.text:10001837 58                                   pop     eax
.text:10001838 8D 64 24 FC                          lea     esp, [esp-4]
.text:1000183C 81 FC 00 10 00 00                    cmp     esp, 1000h
.text:10001842 77 06                                ja      short loc_1000184A
.text:10001844 81 C4 D6 06 00 00                    add     esp, 6D6h
.text:1000184A
.text:1000184A                      loc_1000184A:
.text:1000184A 8B 44 24 FC                          mov     eax, [esp-3670h+arg_3664]
```

TDR by sekoia



```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

TDR by sekoia

Fichier	Motif 1	Motif 2	Motif 3
Initial	48	0	0
Variant 1	69	136	136
Variant 2	1	64	128

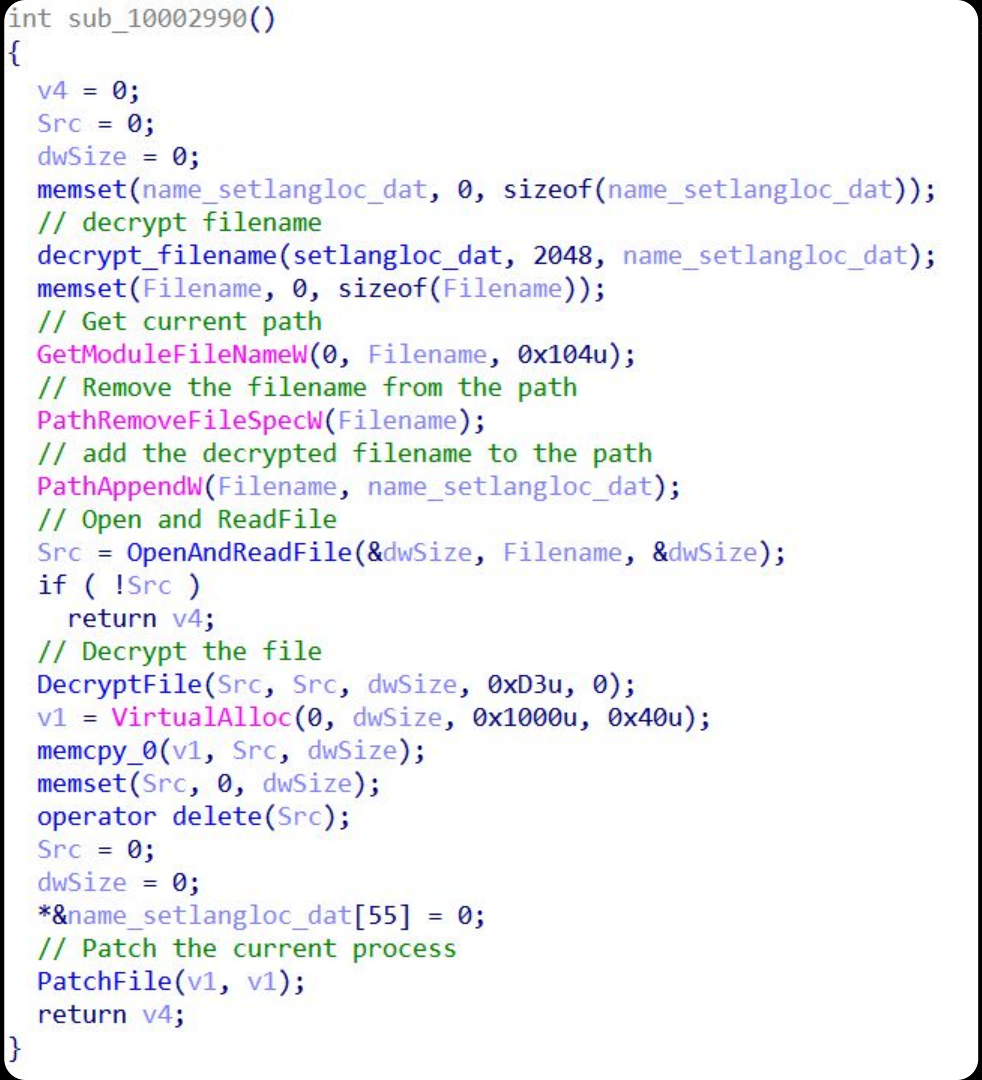
Fichier	Motif 1	Motif 2	Motif 3
Initial	48	0	0
Variant 1	69	136	136
Variant 2	1	64	128

Fichier	Motif 1	Motif 2	Motif 3
Initial	48	0	0
Variant 1	69	136	136
Variant 2	1	64	128

Fichier	Motif 1	Motif 2	Motif 3
Initial	48	0	0
Variant 1	69	136	136
Variant 2	1	64	128

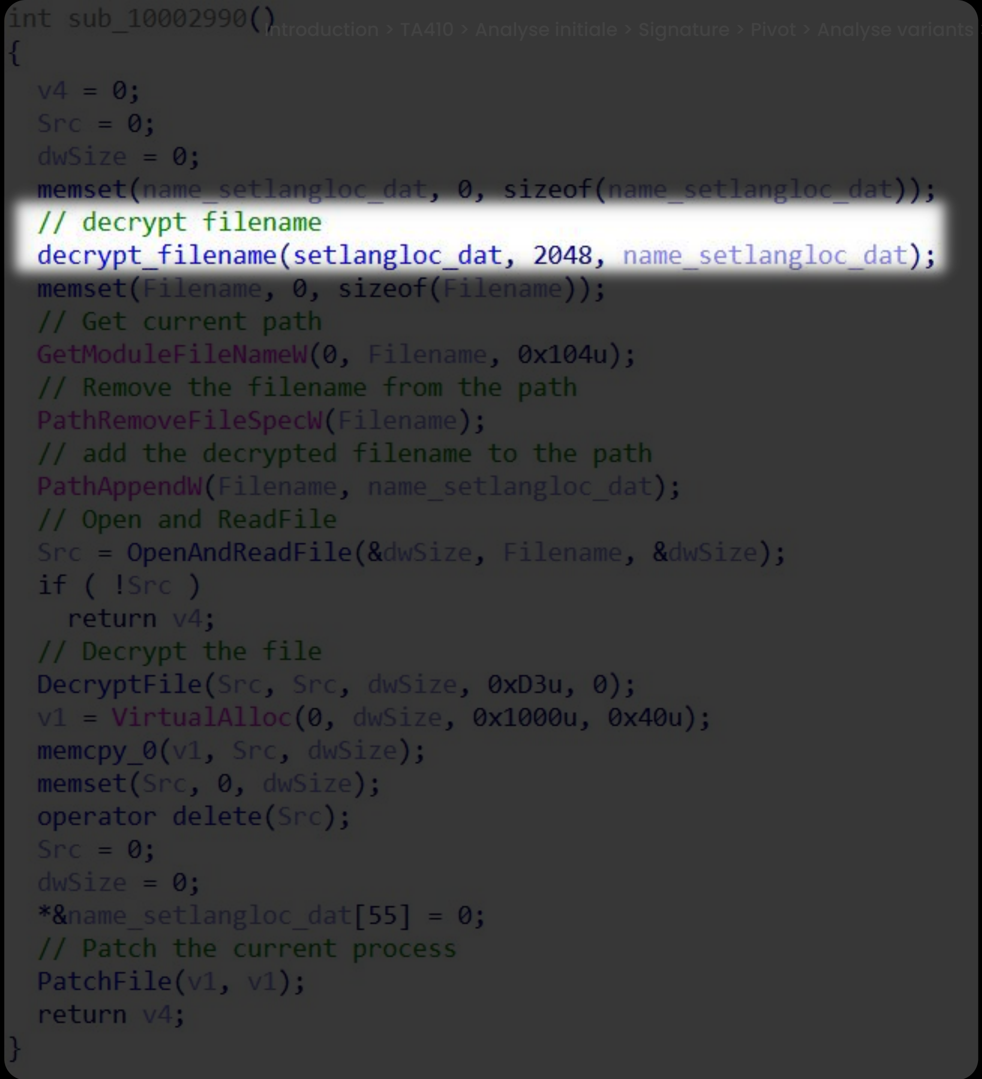
Création d’une nouvelle signature

TDR by sekoia



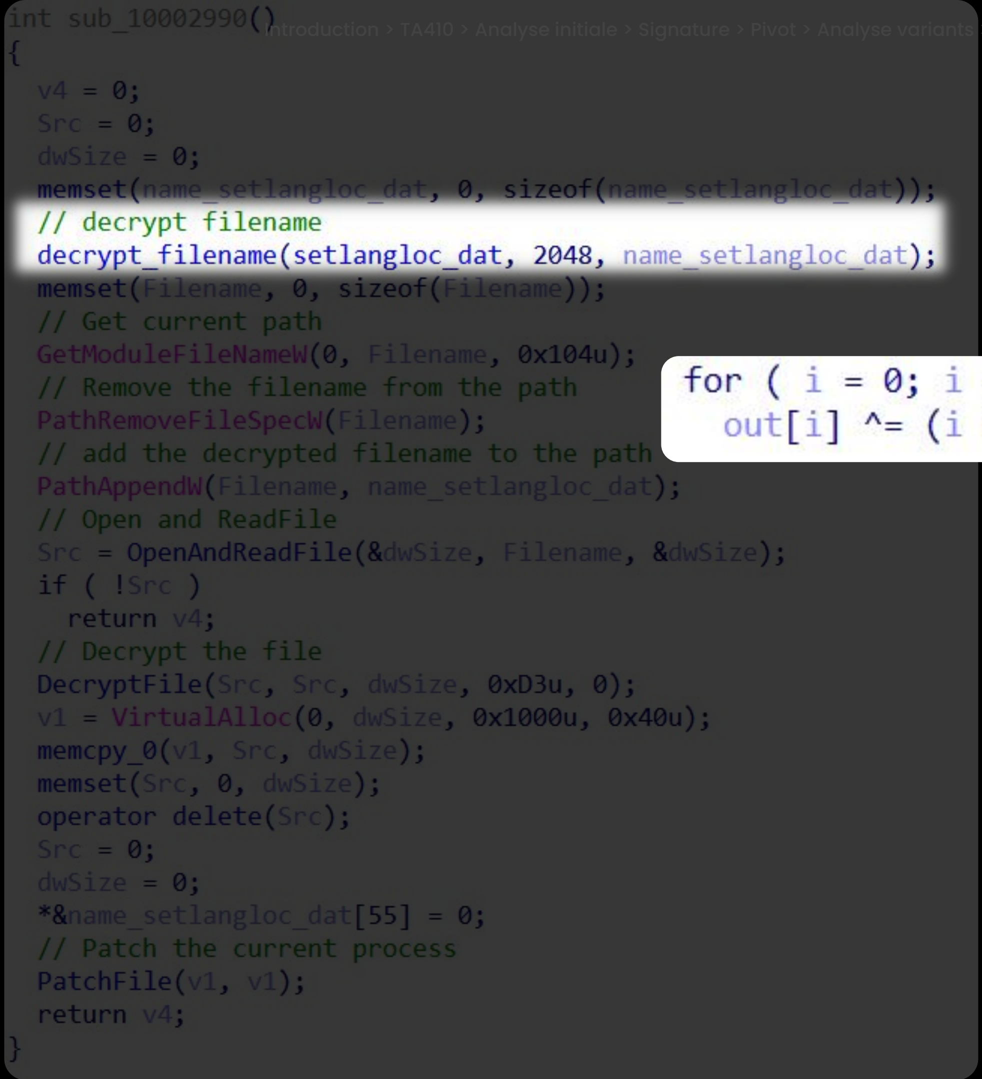
```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

TDR by sekoia

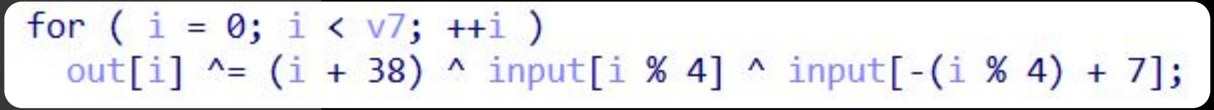


```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

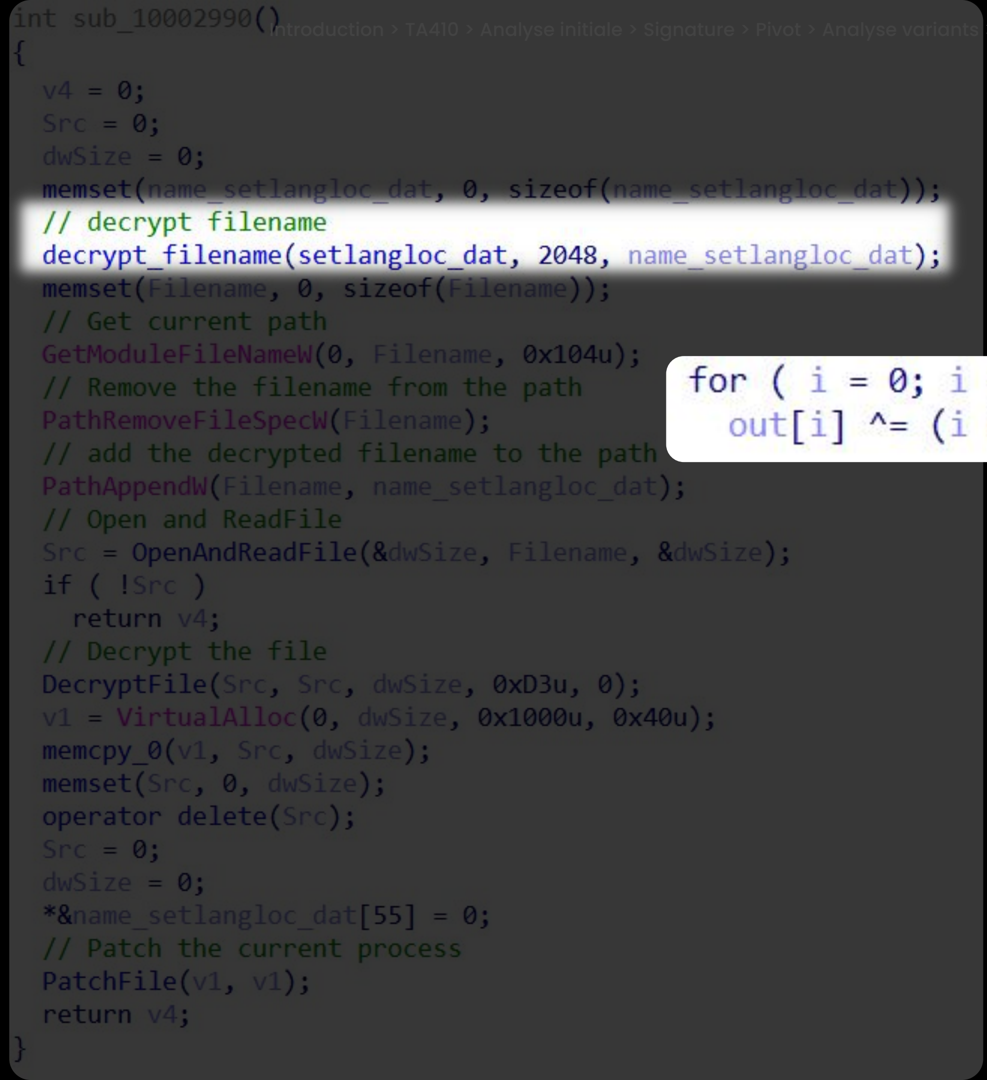
TDR by sekoia



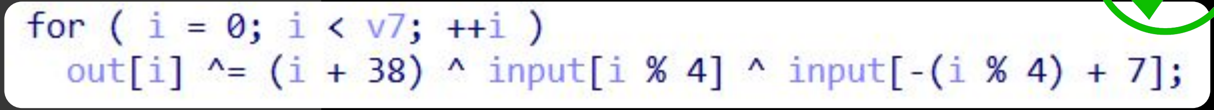
```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```



```
for ( i = 0; i < v7; ++i )
  out[i] ^= (i + 38) ^ input[i % 4] ^ input[-(i % 4) + 7];
```

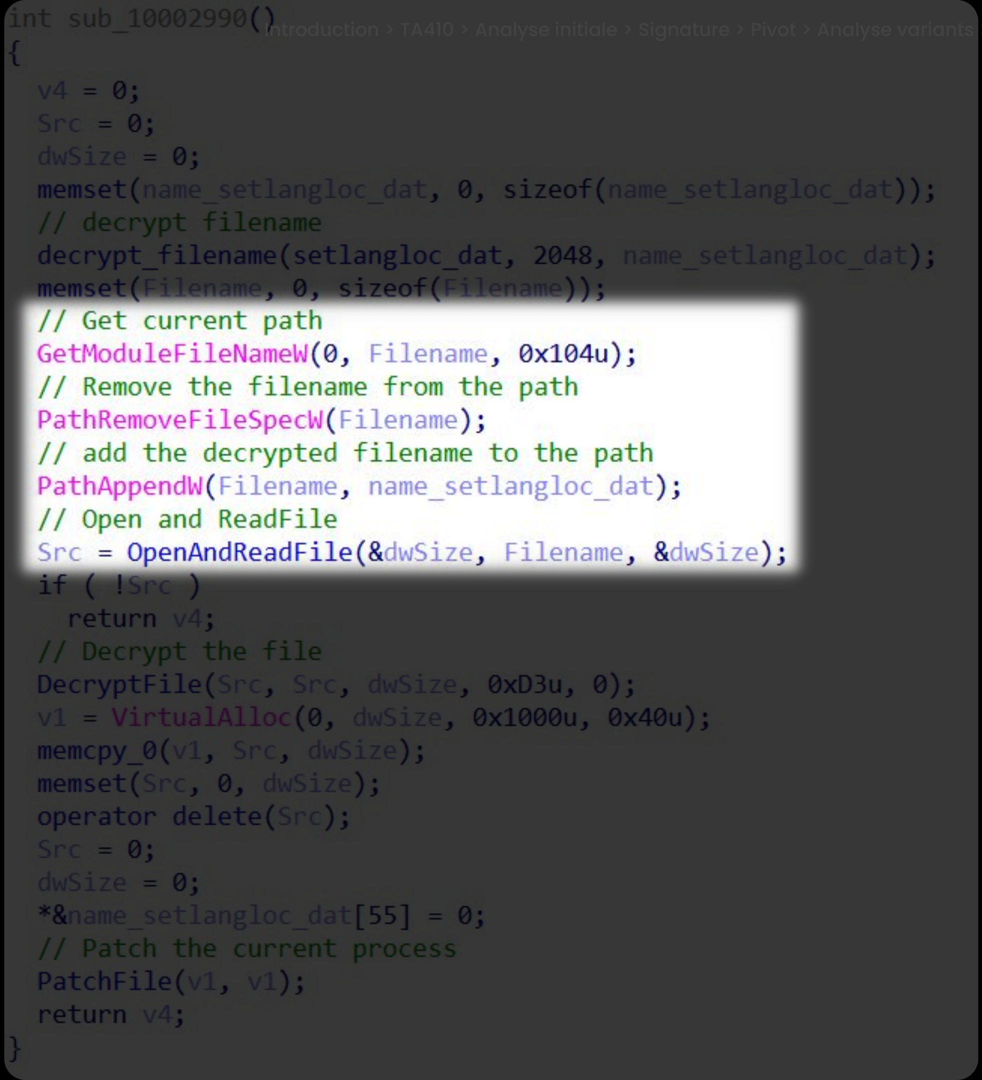


```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

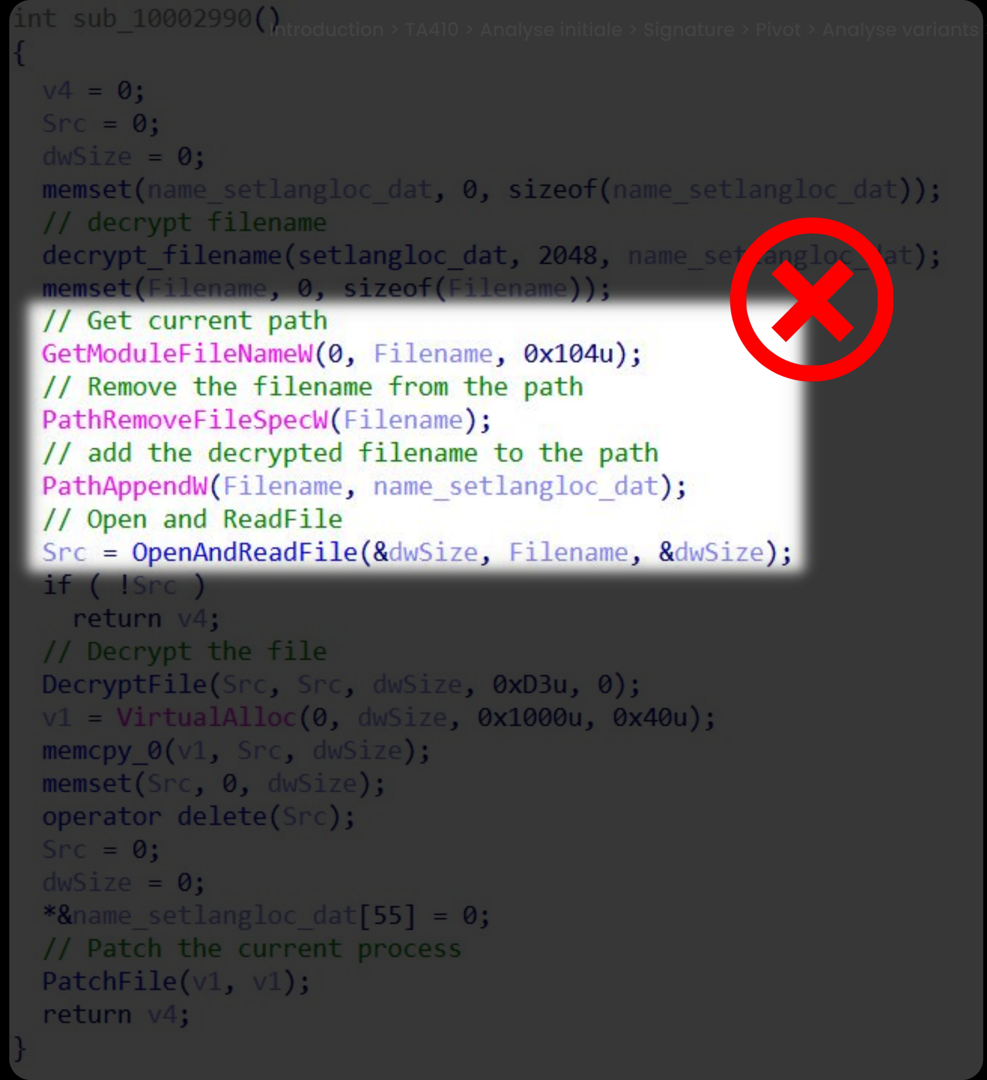


```
for ( i = 0; i < v7; ++i )
  out[i] ^= (i + 38) ^ input[i % 4] ^ input[-(i % 4) + 7];
```

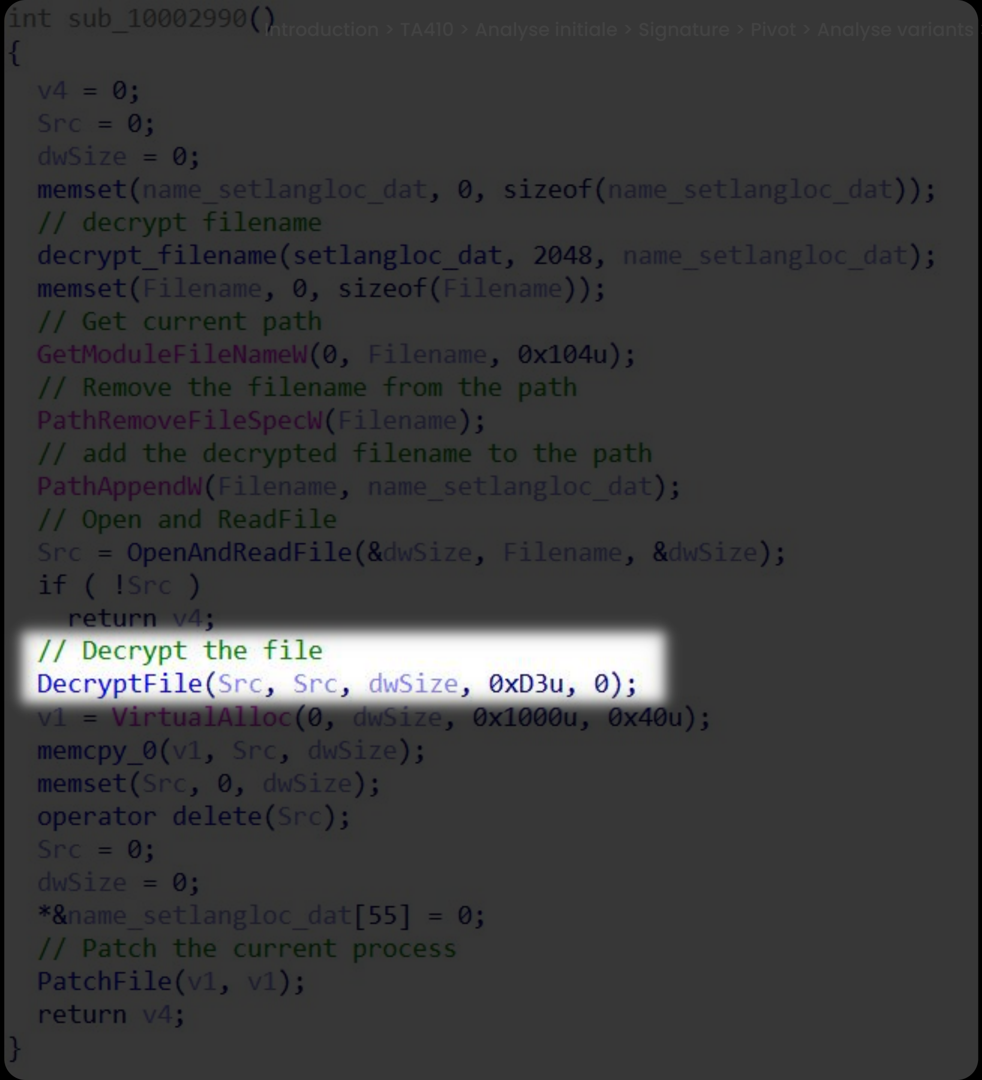
TDR by sekoia



```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

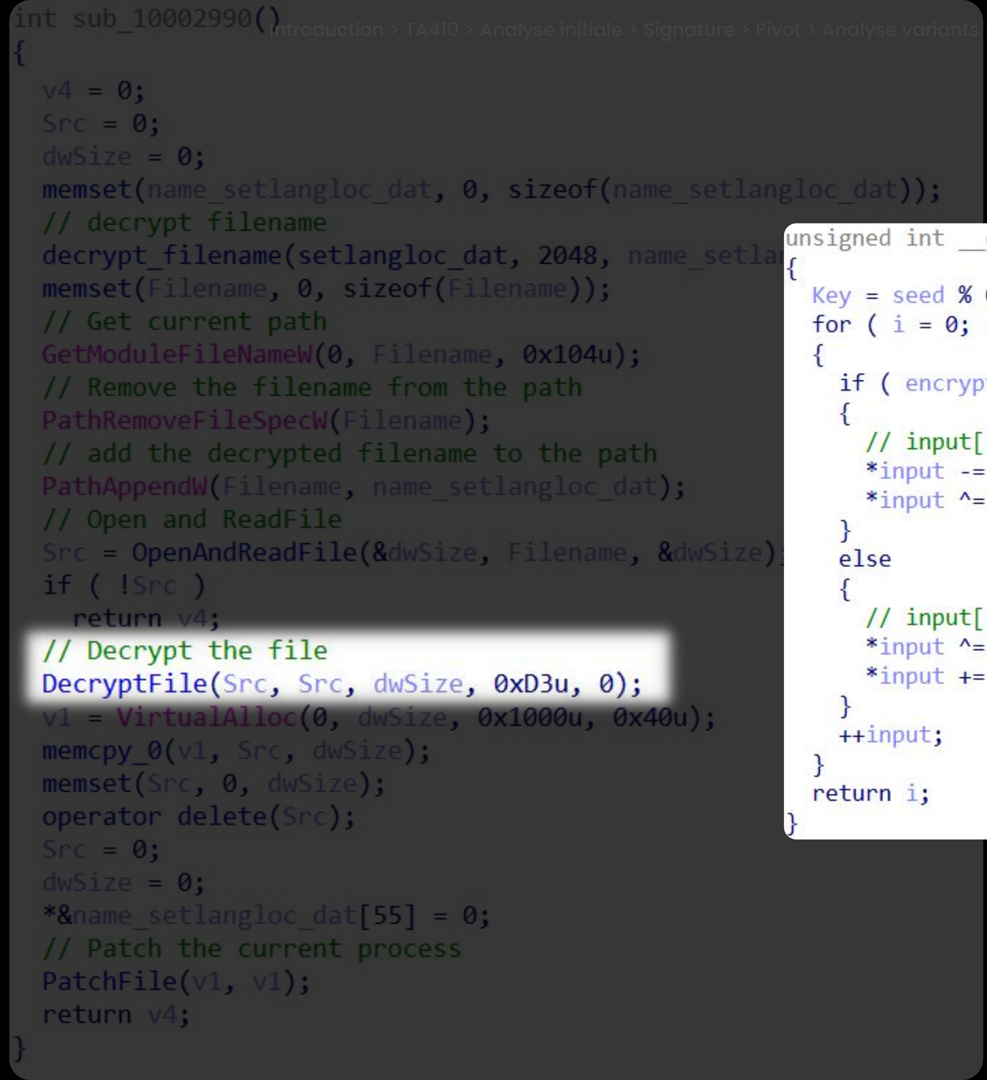


```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

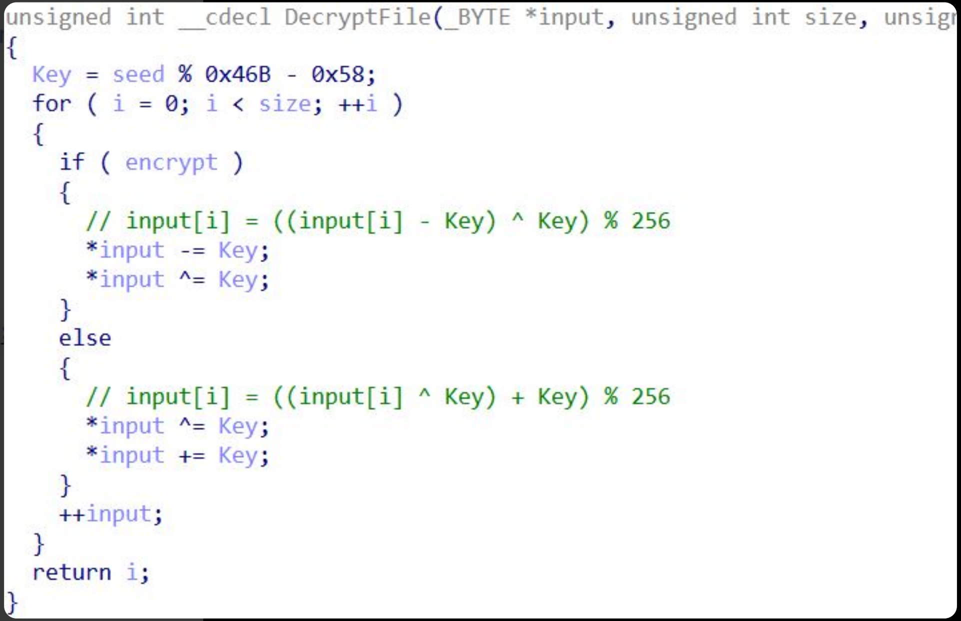


```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

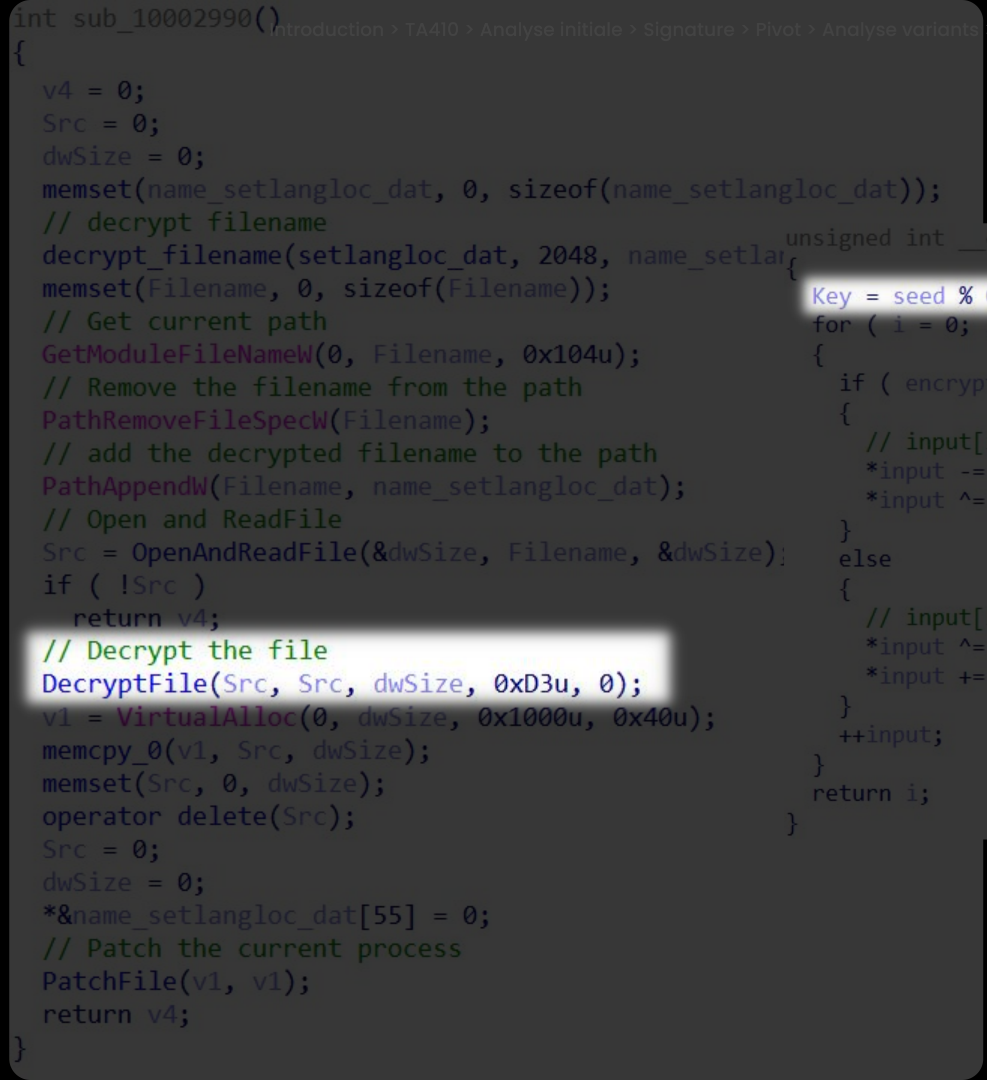
TDR by sekoia



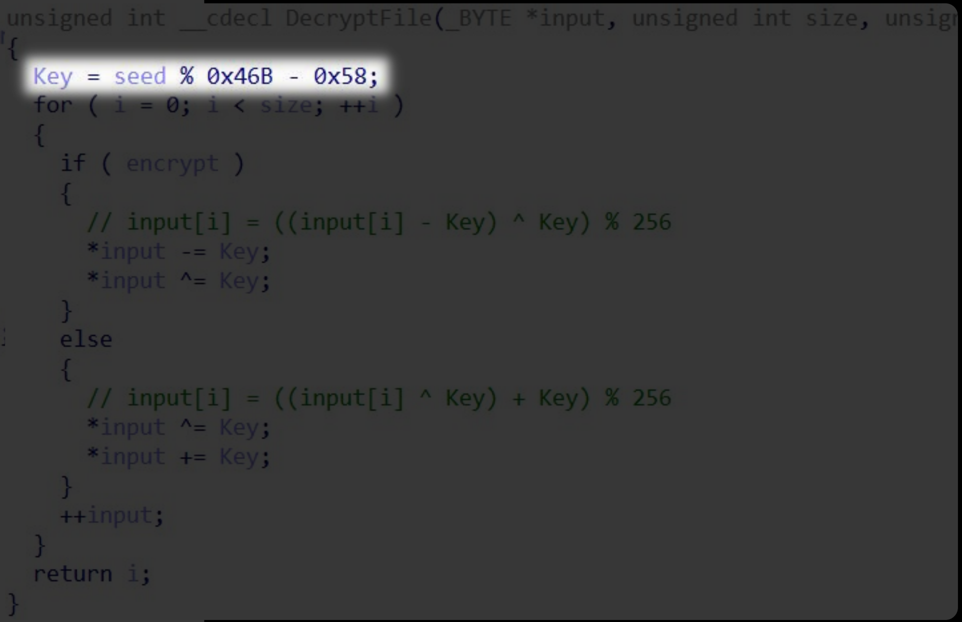
```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setla
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize)
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```



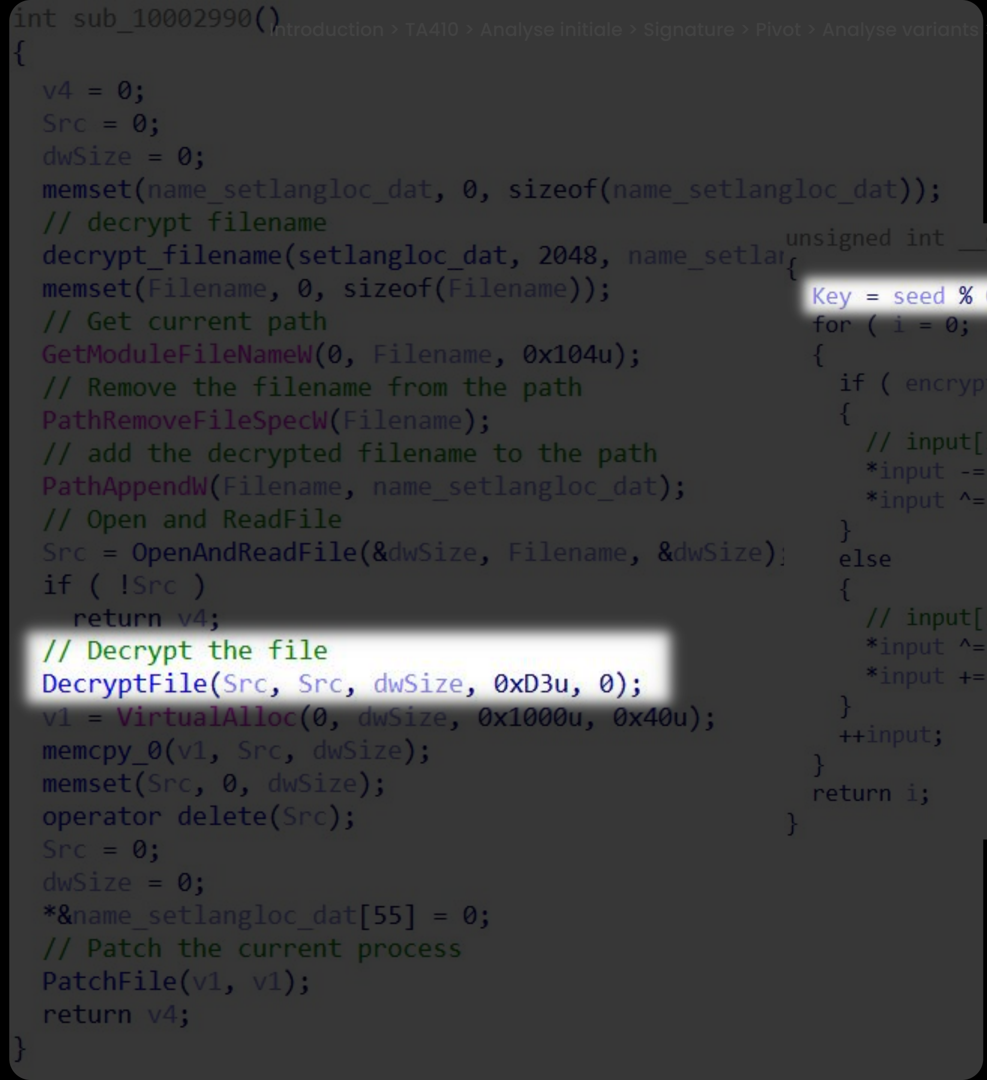
```
unsigned int __cdecl DecryptFile(_BYTE *input, unsigned int size, unsig
{
  Key = seed % 0x46B - 0x58;
  for ( i = 0; i < size; ++i )
  {
    if ( encrypt )
    {
      // input[i] = ((input[i] - Key) ^ Key) % 256
      *input -= Key;
      *input ^= Key;
    }
    else
    {
      // input[i] = ((input[i] ^ Key) + Key) % 256
      *input ^= Key;
      *input += Key;
    }
    ++input;
  }
  return i;
}
```



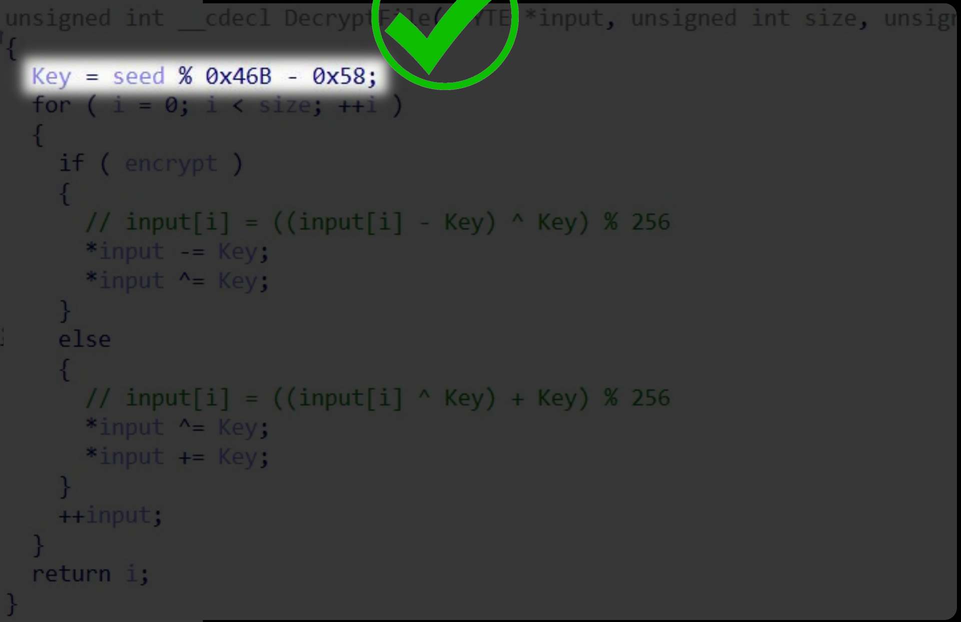
```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setla
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```



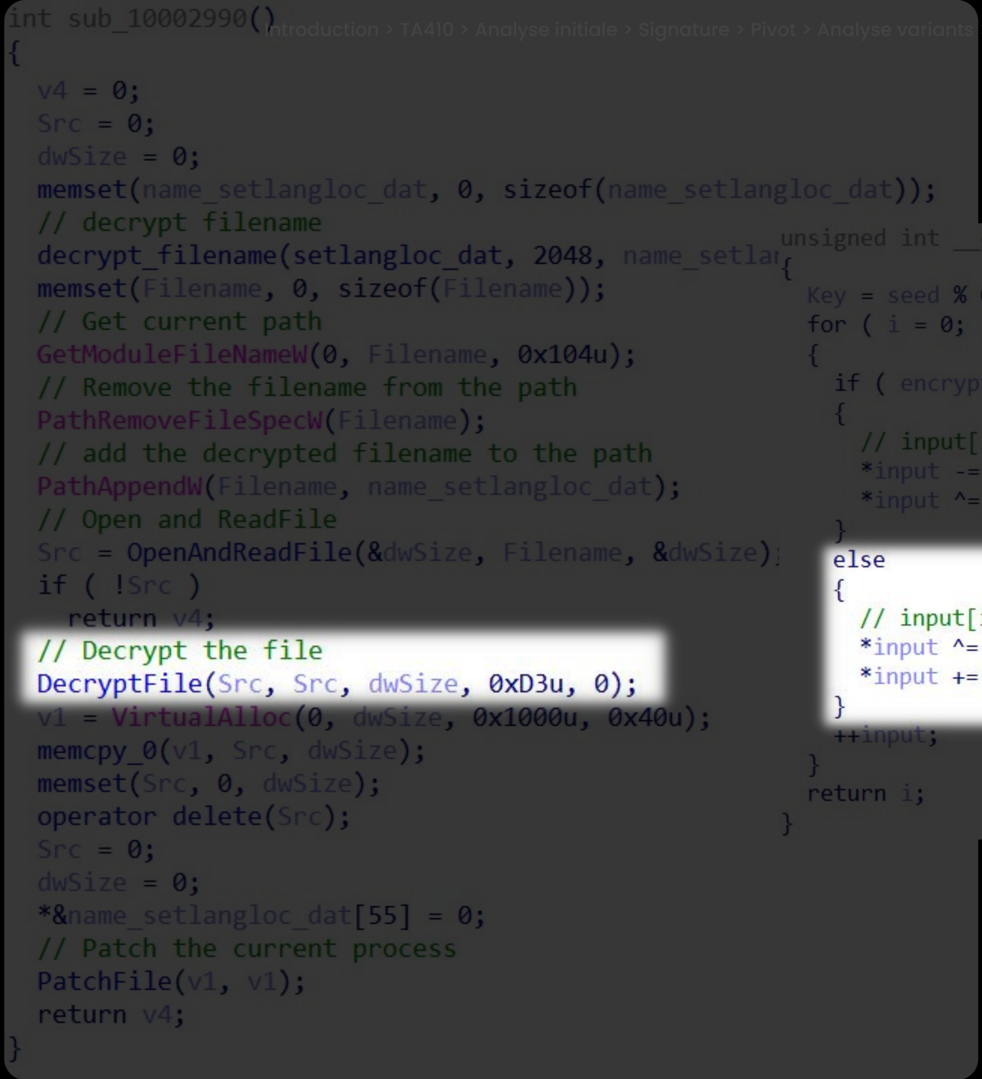
```
unsigned int __cdecl DecryptFile(_BYTE *input, unsigned int size, unsign
{
  Key = seed % 0x46B - 0x58;
  for ( i = 0; i < size; ++i )
  {
    if ( encrypt )
    {
      // input[i] = ((input[i] - Key) ^ Key) % 256
      *input -= Key;
      *input ^= Key;
    }
    else
    {
      // input[i] = ((input[i] ^ Key) + Key) % 256
      *input ^= Key;
      *input += Key;
    }
    ++input;
  }
  return i;
}
```



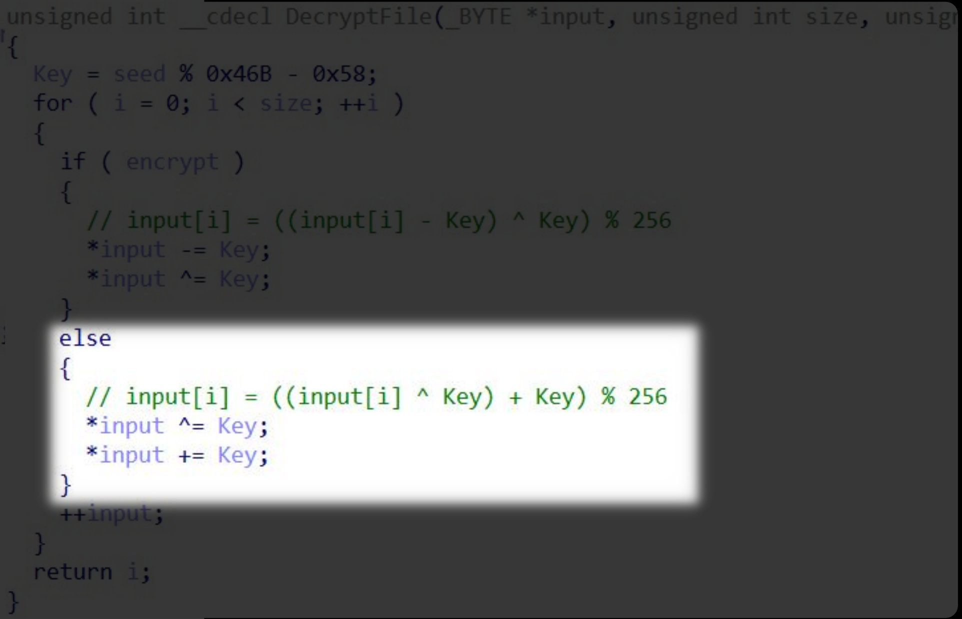
```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setla
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```



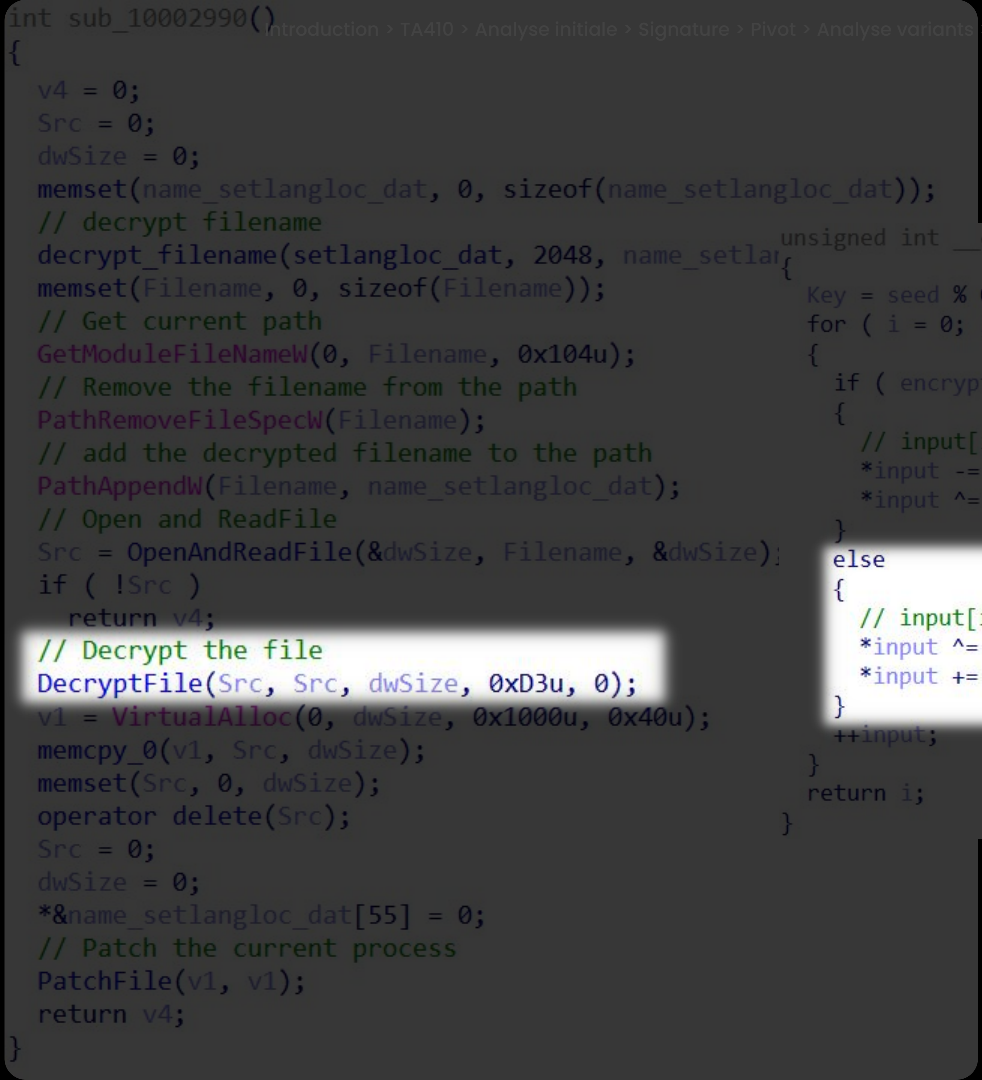
```
unsigned int __cdecl DecryptFile(BYTE *input, unsigned int size, unsig
{
  Key = seed % 0x46B - 0x58;
  for ( i = 0; i < size; ++i )
  {
    if ( encrypt )
    {
      // input[i] = ((input[i] - Key) ^ Key) % 256
      *input -= Key;
      *input ^= Key;
    }
    else
    {
      // input[i] = ((input[i] ^ Key) + Key) % 256
      *input ^= Key;
      *input += Key;
    }
    ++input;
  }
  return i;
}
```



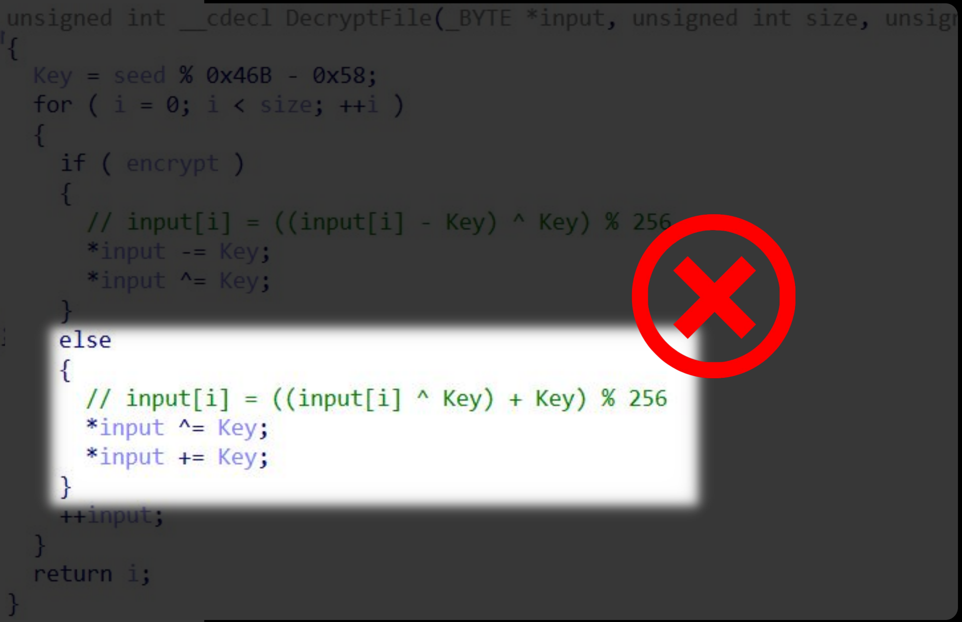
```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setla
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```



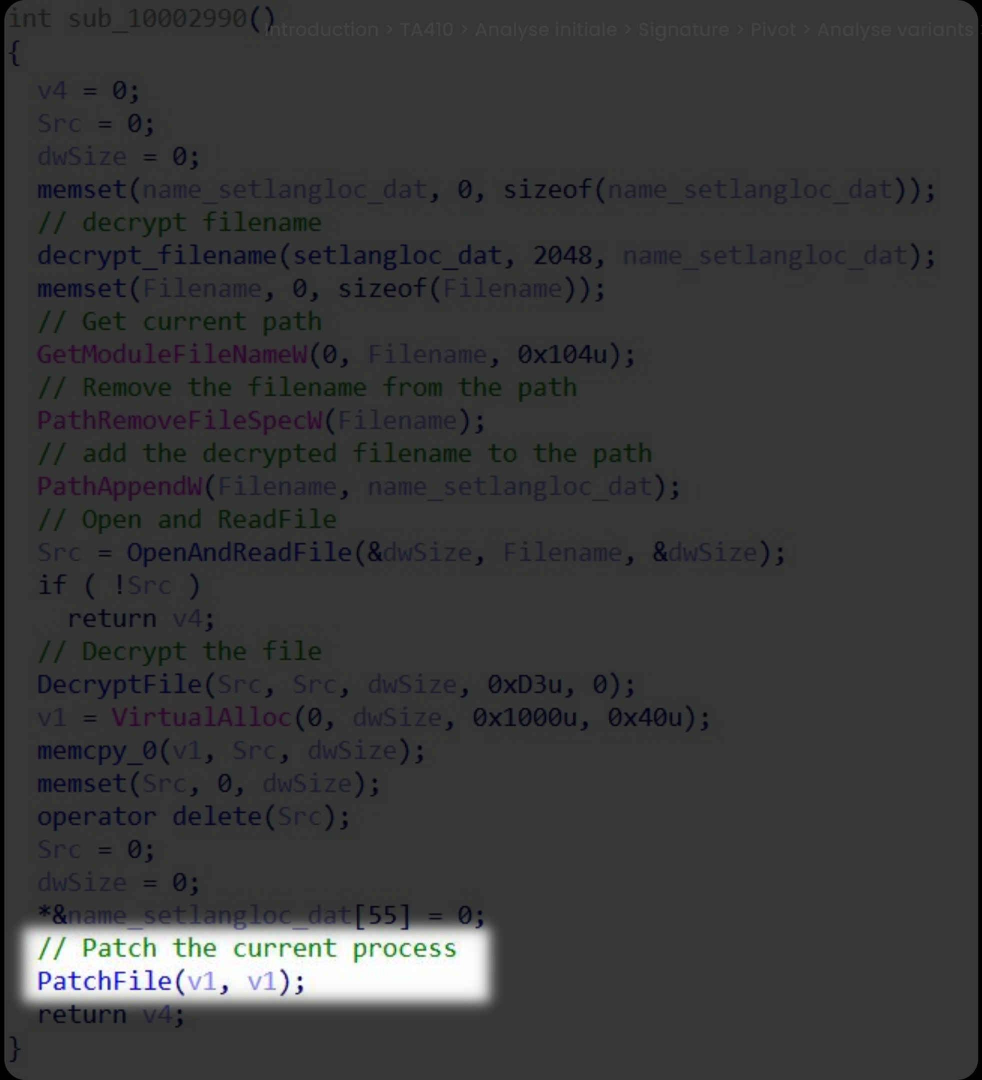
```
unsigned int __cdecl DecryptFile(_BYTE *input, unsigned int size, unsig
{
  Key = seed % 0x46B - 0x58;
  for ( i = 0; i < size; ++i )
  {
    if ( encrypt )
    {
      // input[i] = ((input[i] - Key) ^ Key) % 256
      *input -= Key;
      *input ^= Key;
    }
    else
    {
      // input[i] = ((input[i] ^ Key) + Key) % 256
      *input ^= Key;
      *input += Key;
    }
    ++input;
  }
  return i;
}
```



```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlan
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

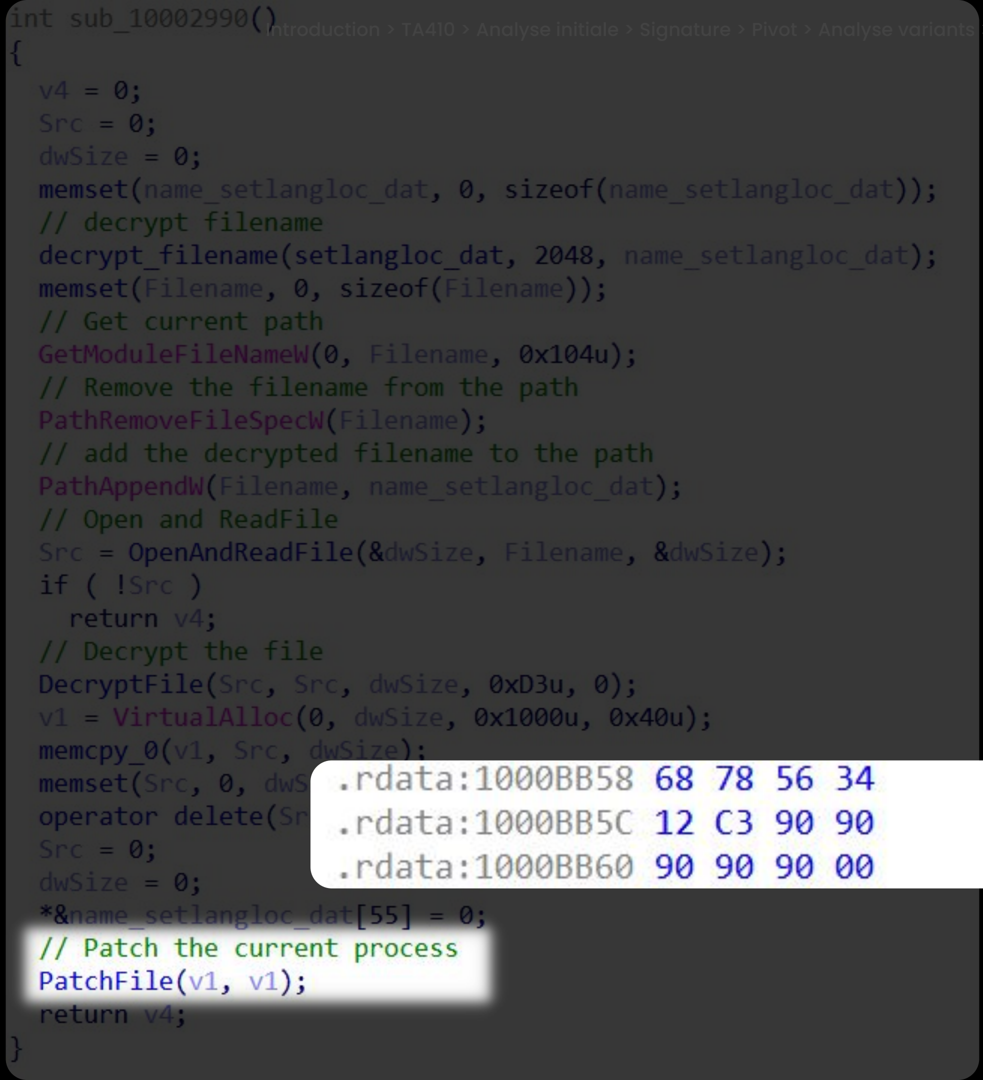


```
unsigned int __cdecl DecryptFile(_BYTE *input, unsigned int size, unsig
{
  Key = seed % 0x46B - 0x58;
  for ( i = 0; i < size; ++i )
  {
    if ( encrypt )
    {
      // input[i] = ((input[i] - Key) ^ Key) % 256
      *input -= Key;
      *input ^= Key;
    }
    else
    {
      // input[i] = ((input[i] ^ Key) + Key) % 256
      *input ^= Key;
      *input += Key;
    }
    ++input;
  }
  return i;
}
```

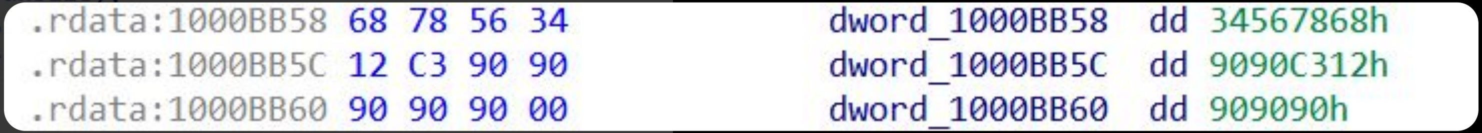


```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwSize);
  operator delete(Src);
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```

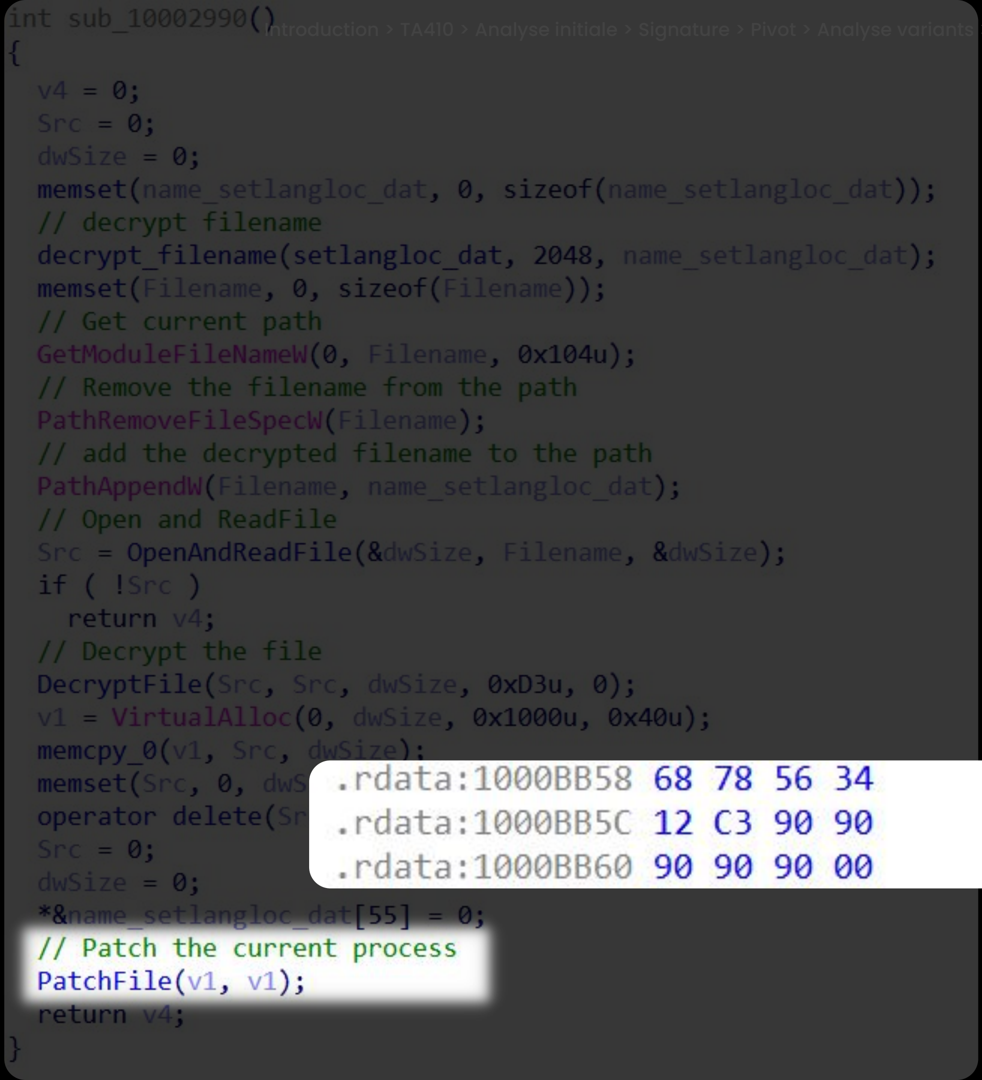
TDR by sekoia



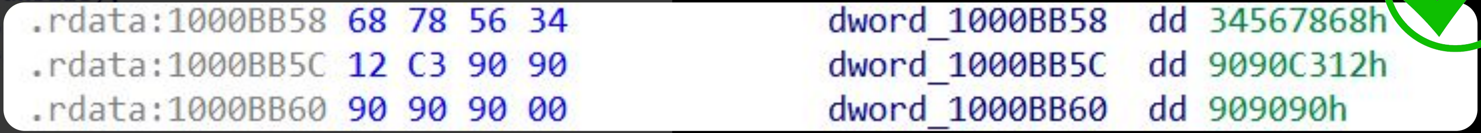
```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwS
  operator delete(Sr
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```



```
.rdata:1000BB58 68 78 56 34             dword_1000BB58  dd 34567868h
.rdata:1000BB5C 12 C3 90 90             dword_1000BB5C  dd 9090C312h
.rdata:1000BB60 90 90 90 00             dword_1000BB60  dd 909090h
```



```
int sub_10002990()
{
  v4 = 0;
  Src = 0;
  dwSize = 0;
  memset(name_setlangloc_dat, 0, sizeof(name_setlangloc_dat));
  // decrypt filename
  decrypt_filename(setlangloc_dat, 2048, name_setlangloc_dat);
  memset(Filename, 0, sizeof(Filename));
  // Get current path
  GetModuleFileNameW(0, Filename, 0x104u);
  // Remove the filename from the path
  PathRemoveFileSpecW(Filename);
  // add the decrypted filename to the path
  PathAppendW(Filename, name_setlangloc_dat);
  // Open and ReadFile
  Src = OpenAndReadFile(&dwSize, Filename, &dwSize);
  if ( !Src )
    return v4;
  // Decrypt the file
  DecryptFile(Src, Src, dwSize, 0xD3u, 0);
  v1 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy_0(v1, Src, dwSize);
  memset(Src, 0, dwS
  operator delete(Sr
  Src = 0;
  dwSize = 0;
  *&name_setlangloc_dat[55] = 0;
  // Patch the current process
  PatchFile(v1, v1);
  return v4;
}
```



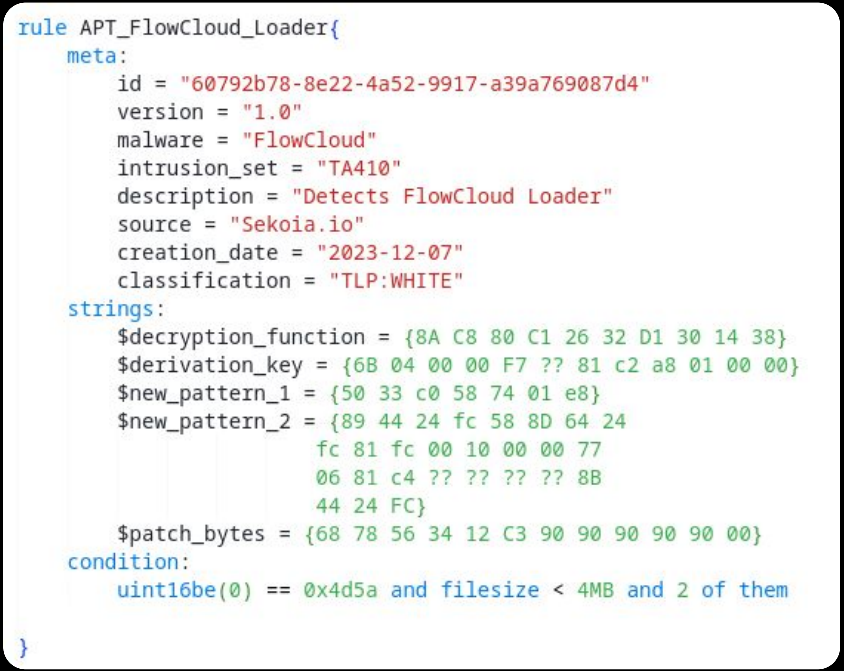
```
.rdata:1000BB58 68 78 56 34        dword_1000BB58  dd 34567868h
.rdata:1000BB5C 12 C3 90 90        dword_1000BB5C  dd 9090C312h
.rdata:1000BB60 90 90 90 00        dword_1000BB60  dd 909090h
```

Nouvelle signature YARA

Basée sur :

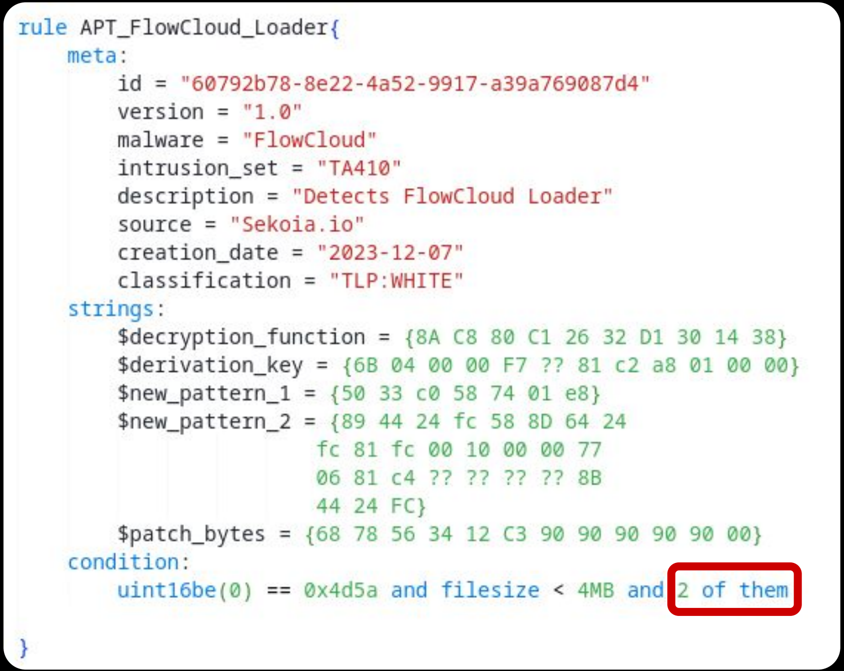
- L’algorithme de déchiffrement du nom de fichier
- L’algorithme de dérivation de clé
- Les mécanismes d’obfuscation
- Le patch

Nouvelle signature YARA

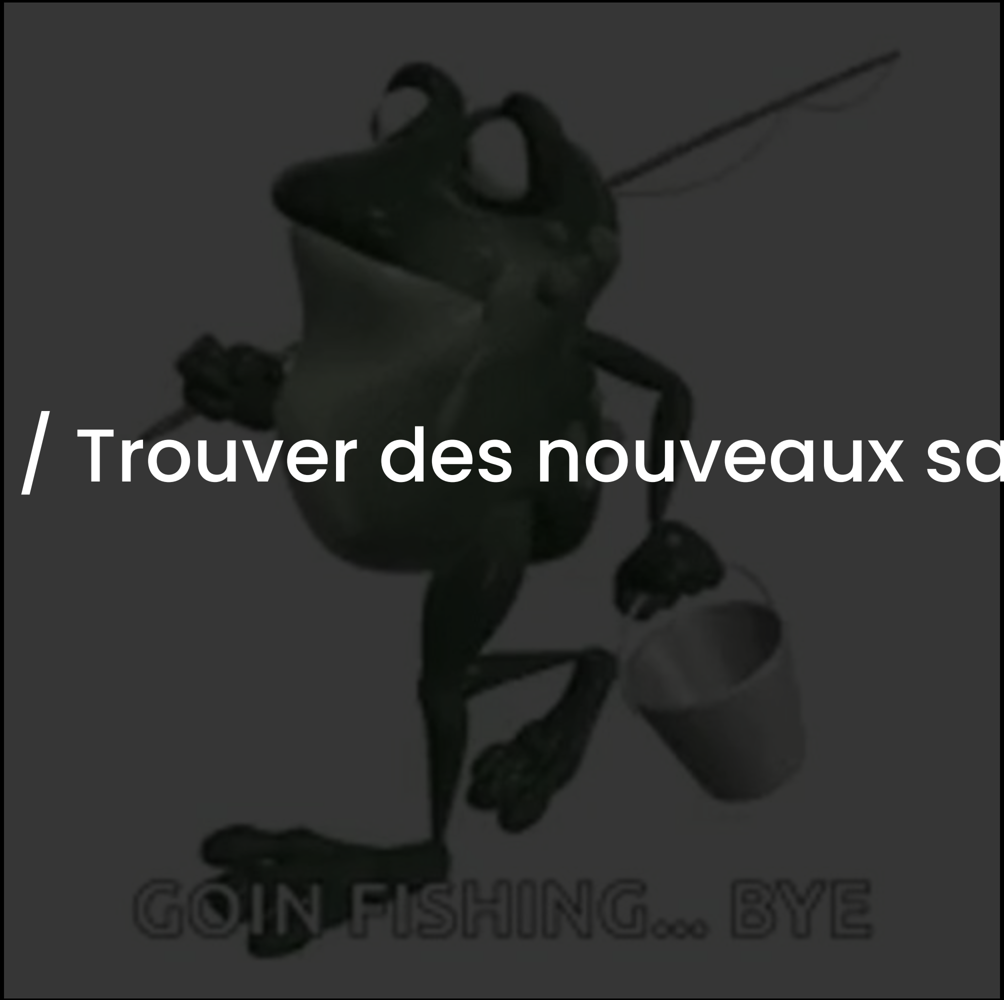


```
rule APT_FlowCloud_Loader{
    meta:
        id = "60792b78-8e22-4a52-9917-a39a769087d4"
        version = "1.0"
        malware = "FlowCloud"
        intrusion_set = "TA410"
        description = "Detects FlowCloud Loader"
        source = "Sekoia.io"
        creation_date = "2023-12-07"
        classification = "TLP:WHITE"
    strings:
        $decryption_function = {8A C8 80 C1 26 32 D1 30 14 38}
        $derivation_key = {6B 04 00 00 F7 ?? 81 c2 a8 01 00 00}
        $new_pattern_1 = {50 33 c0 58 74 01 e8}
        $new_pattern_2 = {89 44 24 fc 58 8D 64 24
                         fc 81 fc 00 10 00 00 77
                         06 81 c4 ?? ?? ?? ?? 8B
                         44 24 FC}
        $patch_bytes = {68 78 56 34 12 C3 90 90 90 90 90 00}
    condition:
        uint16be(0) == 0x4d5a and filesize < 4MB and 2 of them

}
```

Nouvelle signature YARA

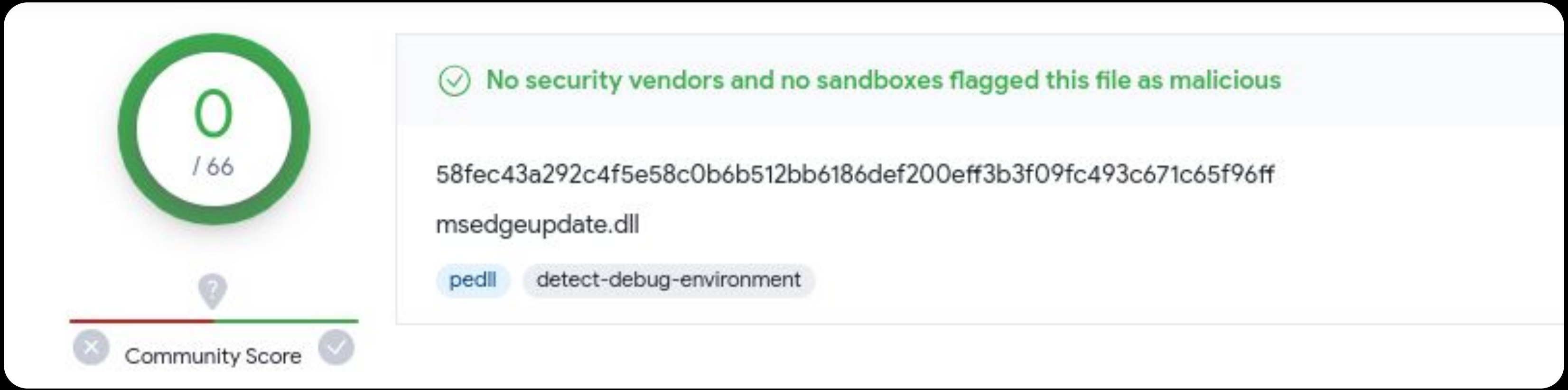


```
rule APT_FlowCloud_Loader{
    meta:
        id = "60792b78-8e22-4a52-9917-a39a769087d4"
        version = "1.0"
        malware = "FlowCloud"
        intrusion_set = "TA410"
        description = "Detects FlowCloud Loader"
        source = "Sekoia.io"
        creation_date = "2023-12-07"
        classification = "TLP:WHITE"
    strings:
        $decryption_function = {8A C8 80 C1 26 32 D1 30 14 38}
        $derivation_key = {6B 04 00 00 F7 ?? 81 c2 a8 01 00 00}
        $new_pattern_1 = {50 33 c0 58 74 01 e8}
        $new_pattern_2 = {89 44 24 fc 58 8D 64 24
                         fc 81 fc 00 10 00 00 77
                         06 81 c4 ?? ?? ?? ?? 8B
                         44 24 FC}
        $patch_bytes = {68 78 56 34 12 C3 90 90 90 90 90 00}
    condition:
        uint16be(0) == 0x4d5a and filesize < 4MB and 2 of them

}
```



Pivoter / Trouver des nouveaux samples

Un nouveau fichier sur VirusTotal



No security vendors and no sandboxes flagged this file as malicious

58fec43a292c4f5e58c0b6b512bb6186def200eff3b3f09fc493c671c65f96ff

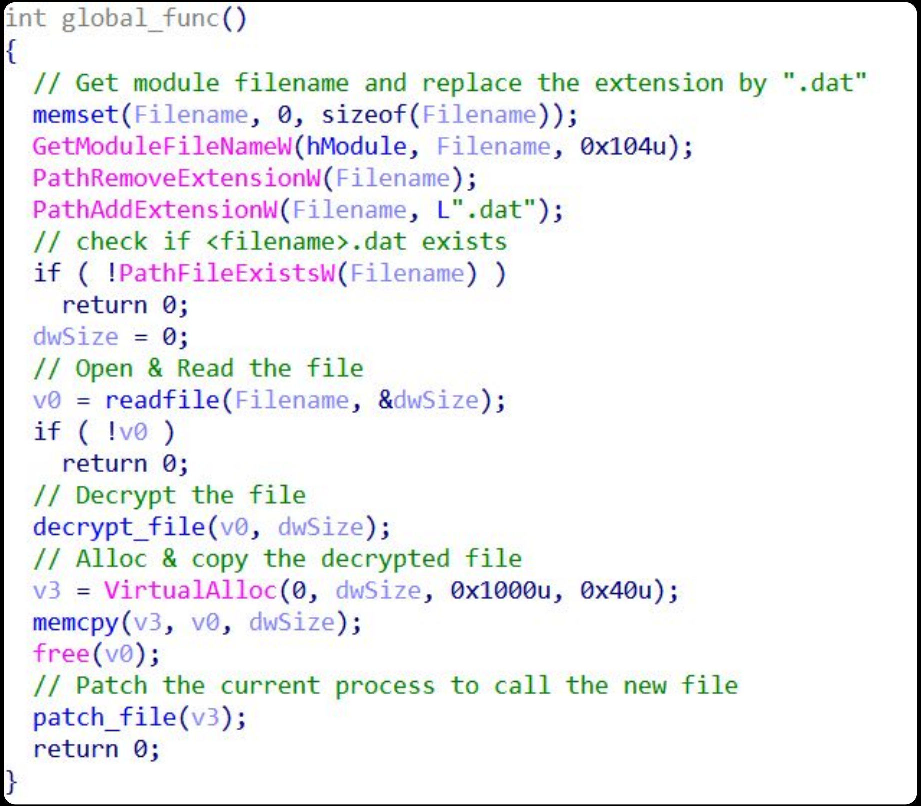
msedgeupdate.dll

pedll detect-debug-environment

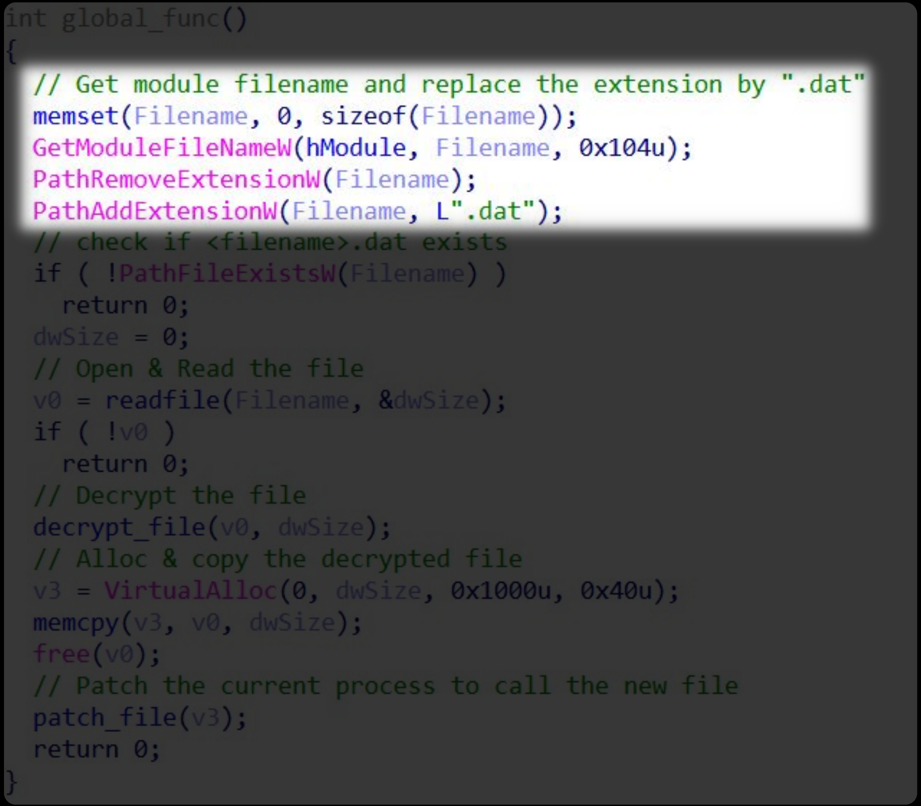
Fichier	Motif 1	Motif 2	Motif 3
Initial	48	0	0
Variant 1	69	136	136
Variant 2	1	64	128
Variant 3	0	280	280

Fichier	Motif 1	Motif 2	Motif 3
Initial	48	0	0
Variant 1	69	136	136
Variant 2	1	64	128
Variant 3	0	280	280

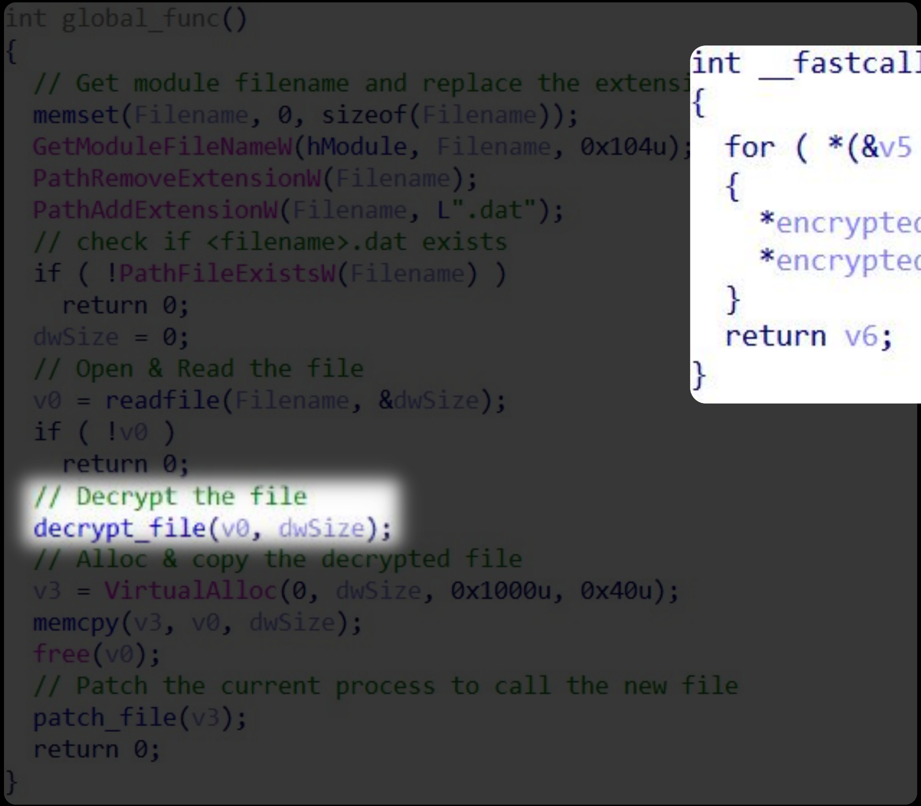
TDR by sekoia



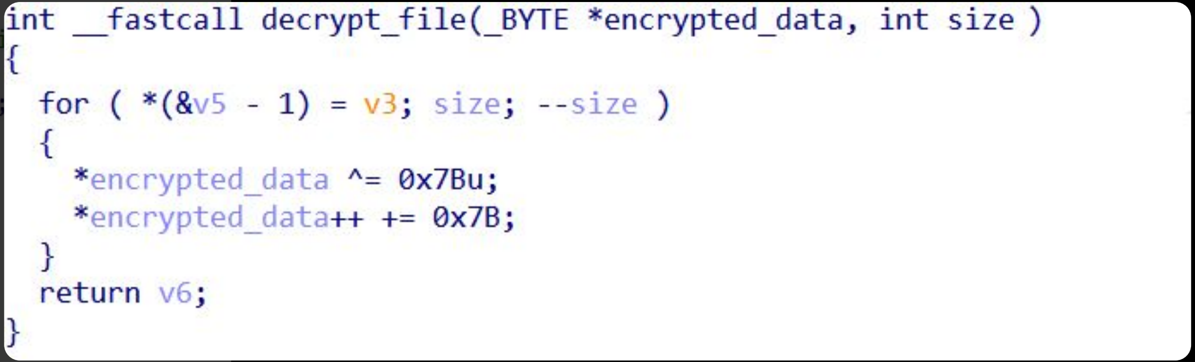
```
int global_func()
{
  // Get module filename and replace the extension by ".dat"
  memset(Filename, 0, sizeof(Filename));
  GetModuleFileNameW(hModule, Filename, 0x104u);
  PathRemoveExtensionW(Filename);
  PathAddExtensionW(Filename, L".dat");
  // check if <filename>.dat exists
  if ( !PathFileExistsW(Filename) )
    return 0;
  dwSize = 0;
  // Open & Read the file
  v0 = readfile(Filename, &dwSize);
  if ( !v0 )
    return 0;
  // Decrypt the file
  decrypt_file(v0, dwSize);
  // Alloc & copy the decrypted file
  v3 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy(v3, v0, dwSize);
  free(v0);
  // Patch the current process to call the new file
  patch_file(v3);
  return 0;
}
```



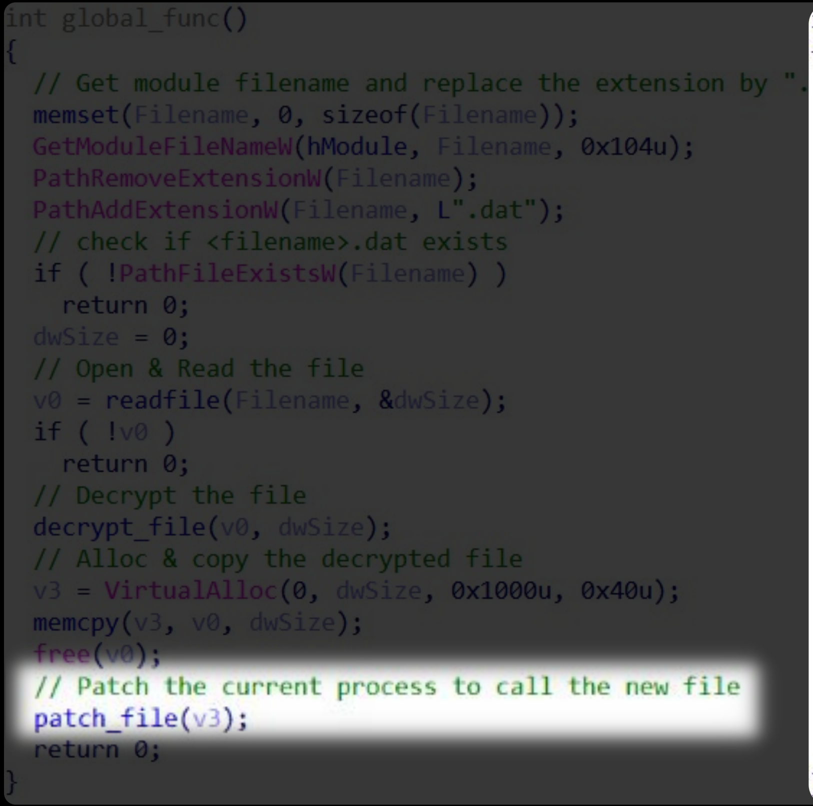
```
int global_func()
{
  // Get module filename and replace the extension by ".dat"
  memset(Filename, 0, sizeof(Filename));
  GetModuleFileNameW(hModule, Filename, 0x104u);
  PathRemoveExtensionW(Filename);
  PathAddExtensionW(Filename, L".dat");
  // check if <filename>.dat exists
  if ( !PathFileExistsW(Filename) )
    return 0;
  dwSize = 0;
  // Open & Read the file
  v0 = readfile(Filename, &dwSize);
  if ( !v0 )
    return 0;
  // Decrypt the file
  decrypt_file(v0, dwSize);
  // Alloc & copy the decrypted file
  v3 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy(v3, v0, dwSize);
  free(v0);
  // Patch the current process to call the new file
  patch_file(v3);
  return 0;
}
```



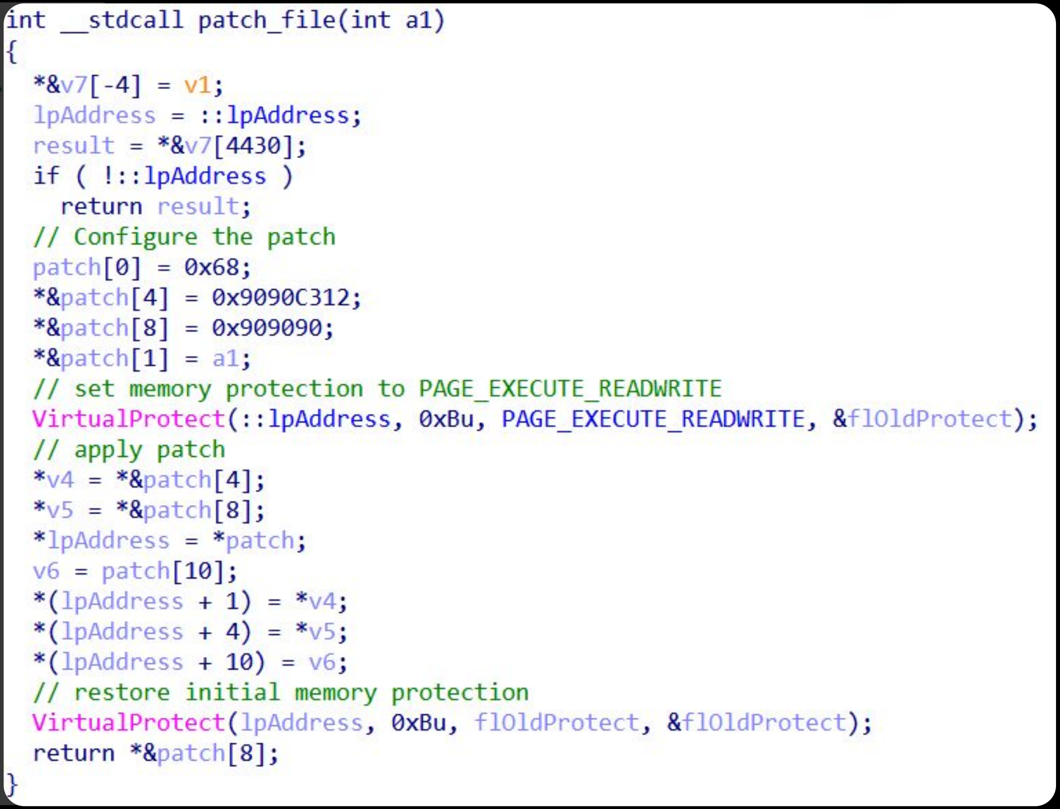
```
int global_func()
{
  // Get module filename and replace the extensi
  memset(Filename, 0, sizeof(Filename));
  GetModuleFileNameW(hModule, Filename, 0x104u);
  PathRemoveExtensionW(Filename);
  PathAddExtensionW(Filename, L".dat");
  // check if <filename>.dat exists
  if ( !PathFileExistsW(Filename) )
    return 0;
  dwSize = 0;
  // Open & Read the file
  v0 = readfile(Filename, &dwSize);
  if ( !v0 )
    return 0;
  // Decrypt the file
  decrypt_file(v0, dwSize);
  // Alloc & copy the decrypted file
  v3 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy(v3, v0, dwSize);
  free(v0);
  // Patch the current process to call the new file
  patch_file(v3);
  return 0;
}
```



```
int __fastcall decrypt_file(_BYTE *encrypted_data, int size )
{
  for ( *(&v5 - 1) = v3; size; --size )
  {
    *encrypted_data ^= 0x7Bu;
    *encrypted_data++ += 0x7B;
  }
  return v6;
}
```



```
int global_func()
{
  // Get module filename and replace the extension by ".
  memset(Filename, 0, sizeof(Filename));
  GetModuleFileNameW(hModule, Filename, 0x104u);
  PathRemoveExtensionW(Filename);
  PathAddExtensionW(Filename, L".dat");
  // check if <filename>.dat exists
  if ( !PathFileExistsW(Filename) )
    return 0;
  dwSize = 0;
  // Open & Read the file
  v0 = readfile(Filename, &dwSize);
  if ( !v0 )
    return 0;
  // Decrypt the file
  decrypt_file(v0, dwSize);
  // Alloc & copy the decrypted file
  v3 = VirtualAlloc(0, dwSize, 0x1000u, 0x40u);
  memcpy(v3, v0, dwSize);
  free(v0);
  // Patch the current process to call the new file
  patch_file(v3);
  return 0;
}
```



```
int __stdcall patch_file(int a1)
{
  *&v7[-4] = v1;
  lpAddress = ::lpAddress;
  result = *&v7[4430];
  if ( !::lpAddress )
    return result;
  // Configure the patch
  patch[0] = 0x68;
  *&patch[4] = 0x9090C312;
  *&patch[8] = 0x909090;
  *&patch[1] = a1;
  // set memory protection to PAGE_EXECUTE_READWRITE
  VirtualProtect(::lpAddress, 0xBu, PAGE_EXECUTE_READWRITE, &flOldProtect);
  // apply patch
  *v4 = *&patch[4];
  *v5 = *&patch[8];
  *lpAddress = *patch;
  v6 = patch[10];
  *(lpAddress + 1) = *v4;
  *(lpAddress + 4) = *v5;
  *(lpAddress + 10) = v6;
  // restore initial memory protection
  VirtualProtect(lpAddress, 0xBu, flOldProtect, &flOldProtect);
  return *&patch[8];
}
```

TDR by sekoia

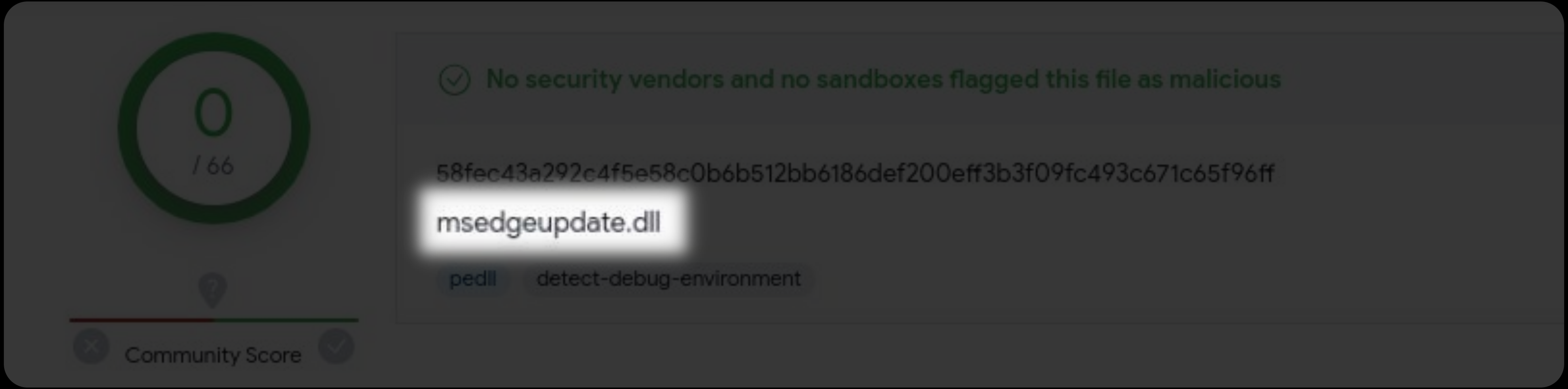
Nouveau fichier

Nombreuses similarités :

- 4 motifs sur 5 sont présents
- seul l'algorithme de dérivation de clé absent

Confiance assez élevée sur l'association avec TA410/FlowCloud

Possibilité d’aller plus loin ?



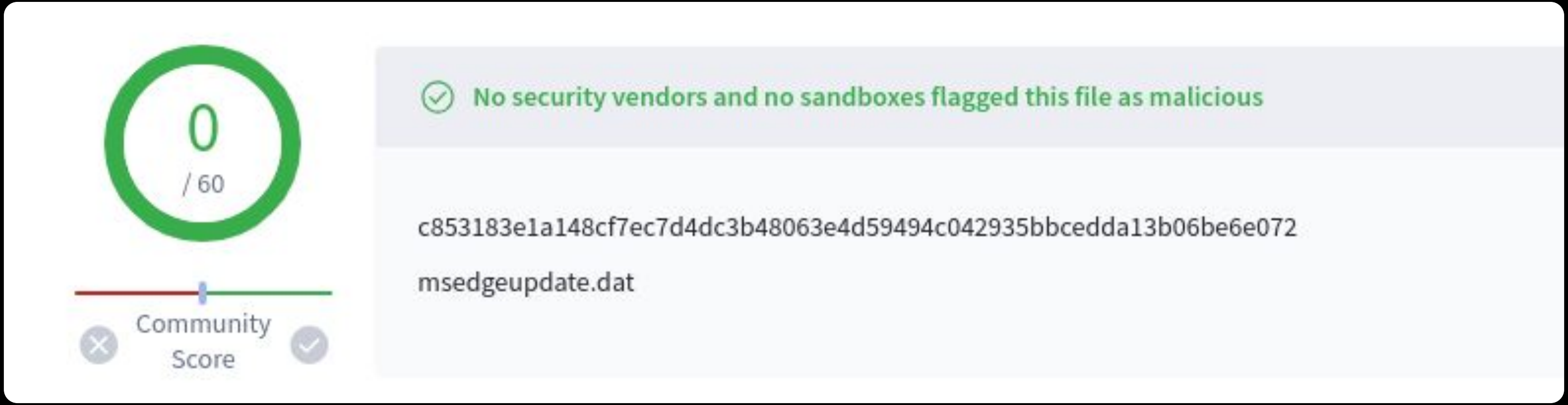
No security vendors and no sandboxes flagged this file as malicious

58fec43a292c4f5e58c0b6b512bb6186def200eff3b3f09fc493c671c65f96ff

msedgeupdate.dll

pedll detect-debug-environment

Est-ce que msedgeupdate.dat existe ?



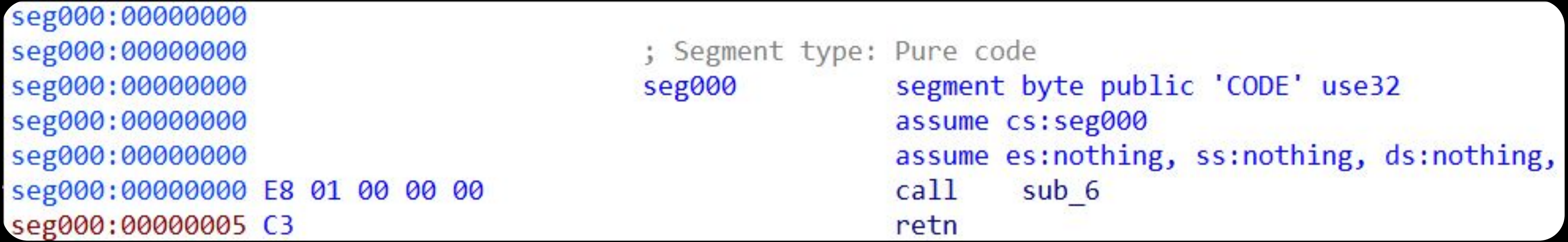
No security vendors and no sandboxes flagged this file as malicious

c853183e1a148cf7ec7d4dc3b48063e4d59494c042935bbcedda13b06be6e072

msedgeupdate.dat

Déchiffrement et chargement dans IDA Pro

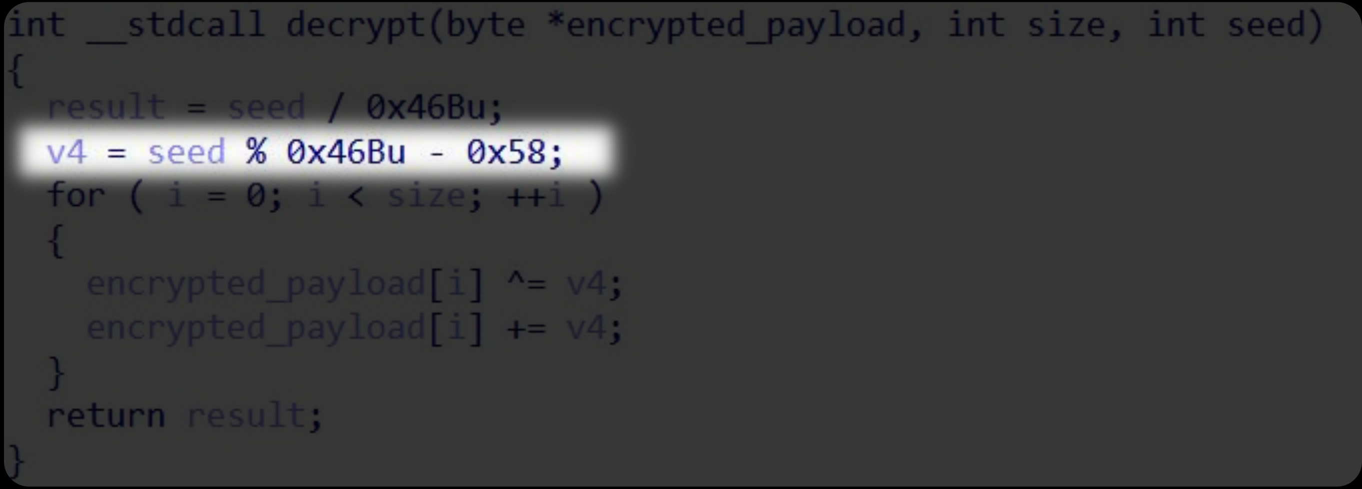
Shellcode



```
seg000:00000000
seg000:00000000                                ; Segment type: Pure code
seg000:00000000                                seg000          segment byte public 'CODE' use32
seg000:00000000                                                assume cs:seg000
seg000:00000000                                                assume es:nothing, ss:nothing, ds:nothing,
seg000:00000000 E8 01 00 00 00                                 call    sub_6
seg000:00000005 C3                                             retn
```

- Algorithme de dérivation de clé
- Déchiffre et charge l’étape suivante (FlowCloud)
- Protégée par VMProtect
- Extraction de l’adresse IP du C2

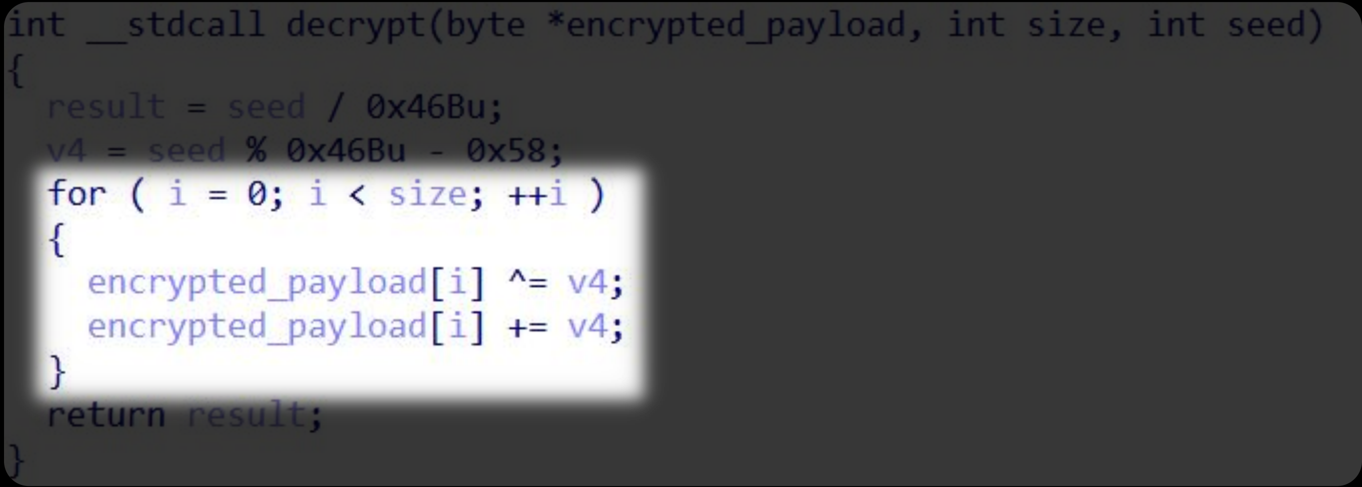
Shellcode



```
int __stdcall decrypt(byte *encrypted_payload, int size, int seed)
{
  result = seed / 0x46Bu;
  v4 = seed % 0x46Bu - 0x58;
  for ( i = 0; i < size; ++i )
  {
    encrypted_payload[i] ^= v4;
    encrypted_payload[i] += v4;
  }
  return result;
}
```

- Algorithme de dérivation de clé
- Déchiffre et charge l’étape suivante (FlowCloud)
- Protégée par VMProtect
- Extraction de l’adresse IP du C2

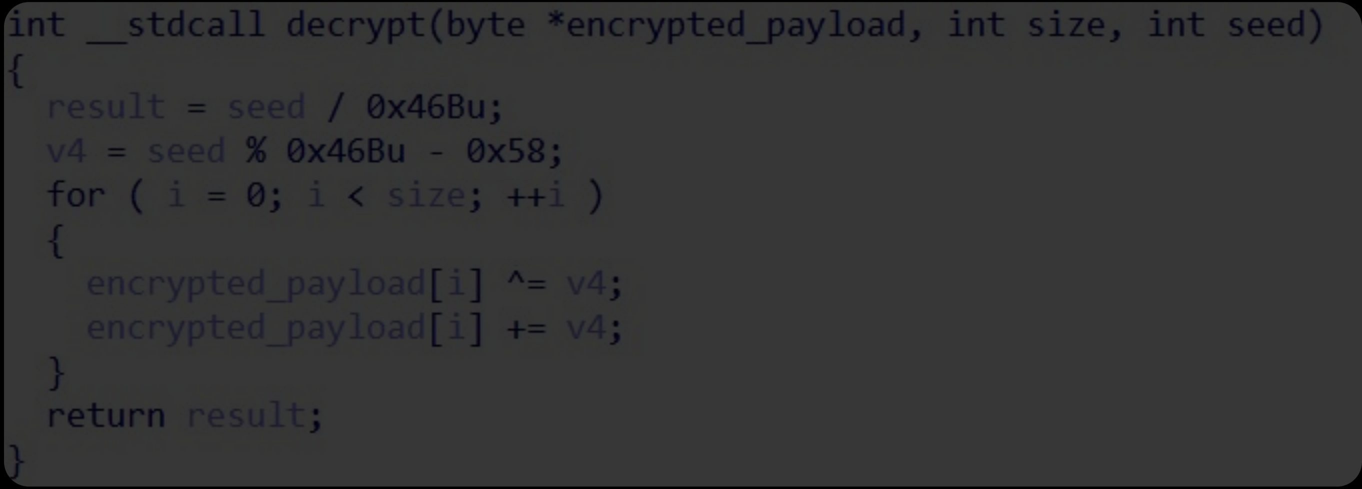
Shellcode



```
int __stdcall decrypt(byte *encrypted_payload, int size, int seed)
{
  result = seed / 0x46Bu;
  v4 = seed % 0x46Bu - 0x58;
  for ( i = 0; i < size; ++i )
  {
    encrypted_payload[i] ^= v4;
    encrypted_payload[i] += v4;
  }
  return result;
}
```

- Algorithme de dérivation de clé
- Déchiffre et charge l'étape suivante (FlowCloud)
- Protégée par VMProtect
- Extraction de l'adresse IP du C2

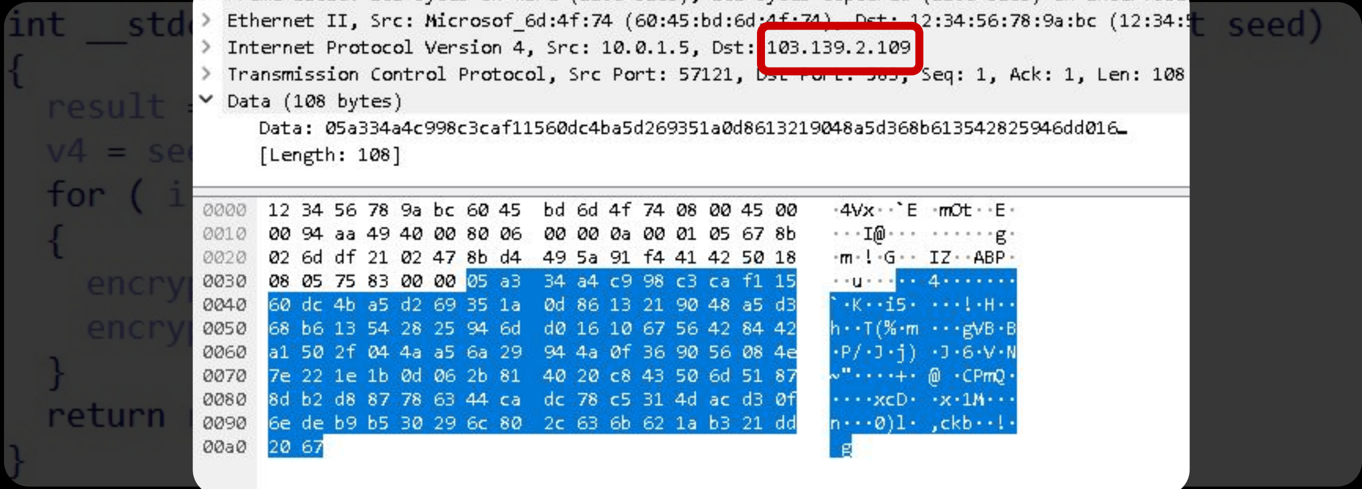
Shellcode



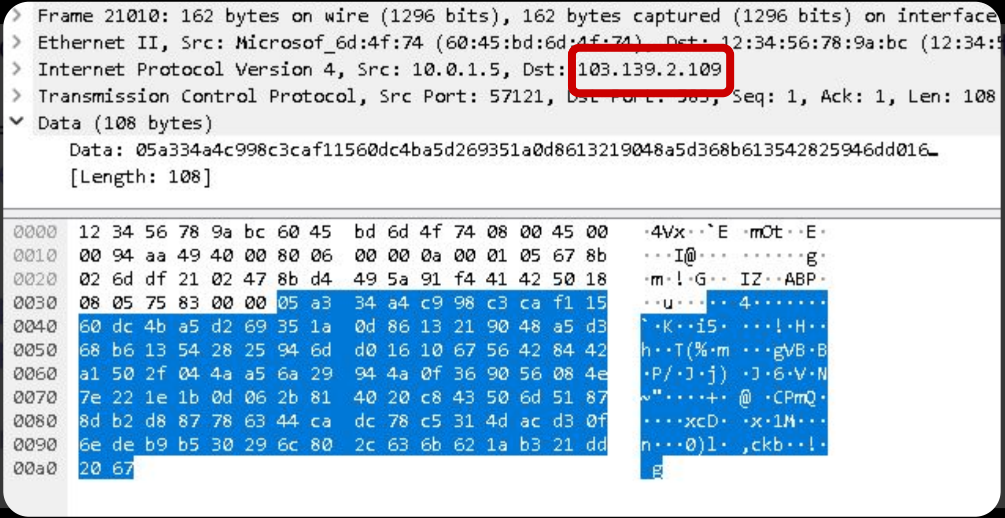
```
int __stdcall decrypt(byte *encrypted_payload, int size, int seed)
{
  result = seed / 0x46Bu;
  v4 = seed % 0x46Bu - 0x58;
  for ( i = 0; i < size; ++i )
  {
    encrypted_payload[i] ^= v4;
    encrypted_payload[i] += v4;
  }
  return result;
}
```

- Algorithme de dérivation de clé
- Déchiffre et charge l’étape suivante (FlowCloud)
- Protégée par VMProtect
- Extraction de l’adresse IP du C2

Shellcode



```
int __std                                              t seed)
{
  result =
  v4 = se
  for ( i
  {
    encry
    encry
  }
  return
}
```



```
> Frame 21010: 162 bytes on wire (1296 bits), 162 bytes captured (1296 bits) on interface
> Ethernet II, Src: Microsof_6d:4f:74 (60:45:bd:6d:4f:74), Dst: 12:34:56:78:9a:bc (12:34:5
> Internet Protocol Version 4, Src: 10.0.1.5, Dst: 103.139.2.109
> Transmission Control Protocol, Src Port: 57121, Dst Port: [illegible], Seq: 1, Ack: 1, Len: 108
v Data (108 bytes)
    Data: 05a334a4c998c3caf11560dc4ba5d269351a0d8613219048a5d368b61354282594 6dd016…
    [Length: 108]

0000  12 34 56 78 9a bc 60 45  bd 6d 4f 74 08 00 45 00   ·4Vx··`E ·mOt··E·
0010  00 94 aa 49 40 00 80 06  00 00 0a 00 01 05 67 8b   ···I@··· ······g·
0020  02 6d df 21 02 47 8b d4  49 5a 91 f4 41 42 50 18   ·m·!·G·· IZ··ABP·
0030  08 05 75 83 00 00 05 a3  34 a4 c9 98 c3 ca f1 15   ··u····· 4·······
0040  60 dc 4b a5 d2 69 35 1a  0d 86 13 21 90 48 a5 d3   `·K··i5· ···!·H··
0050  68 b6 13 54 28 25 94 6d  d0 16 10 67 56 42 84 42   h··T(%·m ···gVB·B
0060  a1 50 2f 04 4a a5 6a 29  94 4a 0f 36 90 56 08 4e   ·P/·J·j) ·J·6·V·N
0070  7e 22 1e 1b 0d 06 2b 81  40 20 c8 43 50 6d 51 87   ~"····+· @ ·CPmQ·
0080  8d b2 d8 87 78 63 44 ca  dc 78 c5 31 4d ac d3 0f   ····xcD· ·x·1M···
0090  6e de b9 b5 30 29 6c 80  2c 63 6b 62 1a b3 21 dd   n···0)l· ,ckb··!·
00a0  20 67                                               g
```

- Algorithme de dérivation de clé
- Déchiffre et charge l’étape suivante (FlowCloud)
- Protégée par VMProtect
- Extraction de l’adresse IP du C2

Bilan

Bilan

- Pivot via la création d'une règle YARA sur :
  - des mécanismes d'obfuscation
  - des algorithmes cryptographique
- Un nouveau fichier trouvé avec 0 détection sur VirusTotal
  - Associé à FlowCloud/TA410 (confiance élevée)
- Pivot via l'adresse IP compliqué

TDR by sekoia

Bilan

- Pivot via la création d'une règle YARA sur :
    - des mécanismes d'obfuscation
    - des algorithmes cryptographique
- Un nouveau fichier trouvé avec 0 détection sur VirusTotal
    - Associé à FlowCloud/TA410 (confiance élevée)
- Pivot via l'adresse IP compliqué

Bilan

- Pivot via la création d'une règle YARA sur :
  - des mécanismes d'obfuscation
  - des algorithmes cryptographique
- Un nouveau fichier trouvé avec 0 détection sur VirusTotal
  - Associé à FlowCloud/TA410 (confiance élevée)
- Pivot via l'adresse IP compliqué

Conclusion

Conclusion

- Présentation la démarche d’un *reverser*
- Les attaquants mettent à jour leur code
  - Mais on peut réussir à les suivre
- Panorama non exhaustif de l’analyse de *malware*

Questions?

Charles Meslay

TDR by sekoia
THREAT DETECTION & RESEARCH