Analyzing Amadey – a simple native malware

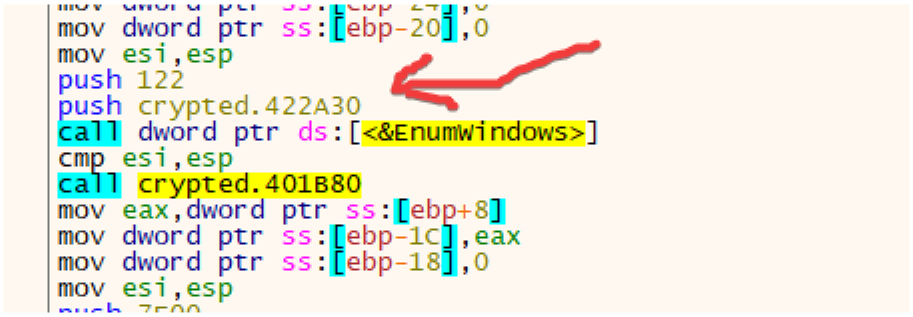
krabsonsecurity.com/2019/02/13/analyzing-amadey-a-simple-native-malware/

Posted on February 13, 2019

Apparently there is a new Russian botnet floating around by the name of Amadey. Despite the very high price tag on Russian forums, it is a very simplistic bot that is quite honestly poorly made.

> SHA-1: B7235E2981456D29412AD935BDBCA140B6AD0151
>
> Compiler info (from ExeInfo PE): Microsoft Visual C++ ver 5.0/6.0
>
> *Sample given by a friend.*

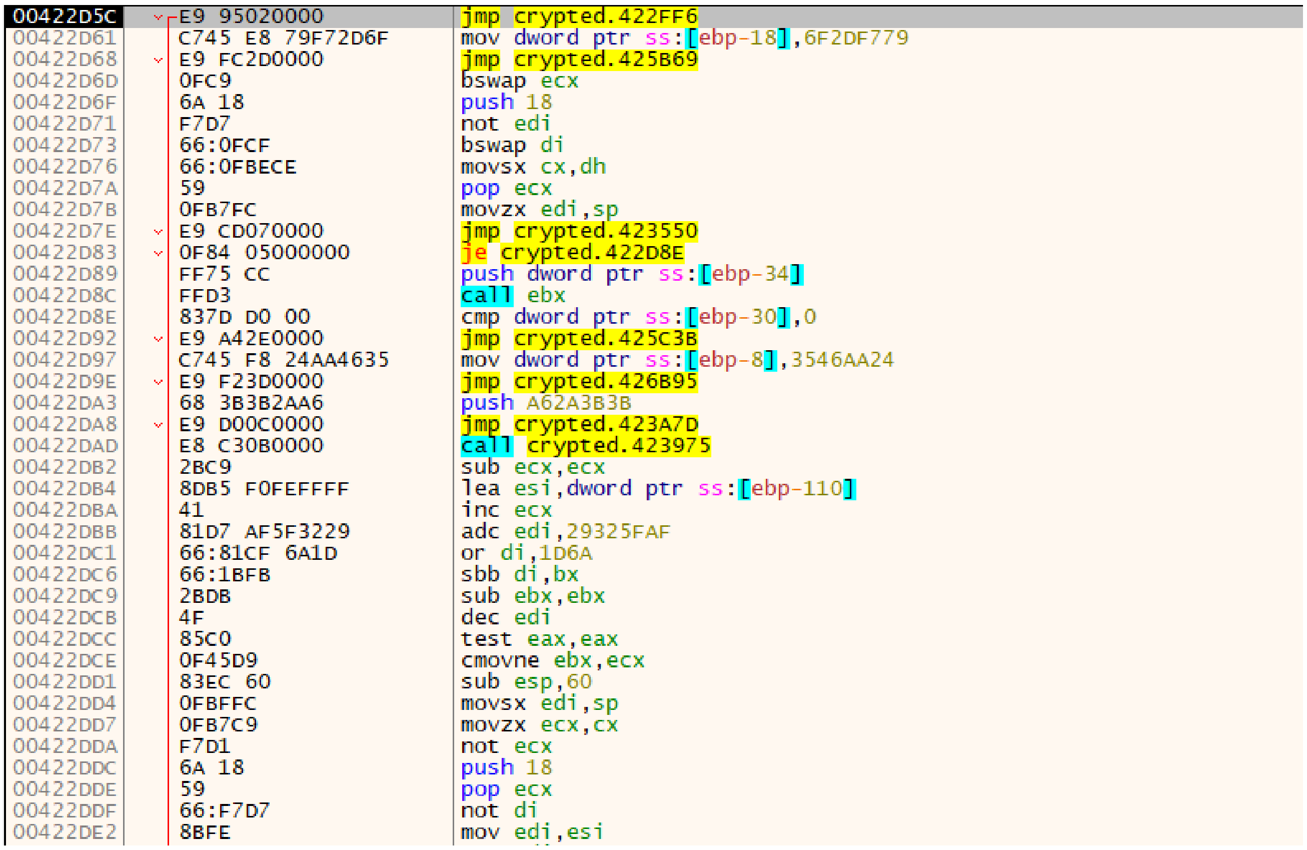
The payload was not spreaded directly but rather was packed with a crypter. The crypter seems to be TitanCrypt, based on the storage method (appended section which contains the payload which was encrypted and then base64 encoded). The crypter's code is encrypted using a self-decryptor as well as heavily obfuscated, and is executed by using windows API callbacks.



```
mov dword ptr ss:[ebp-20],0
mov esi,esp
push 122
push crypted.422A30
call dword ptr ds:[<&EnumWindows>]
cmp esi,esp
call crypted.401B80
mov eax,dword ptr ss:[ebp+8]
mov dword ptr ss:[ebp-1C],eax
mov dword ptr ss:[ebp-18],0
mov esi,esp
```

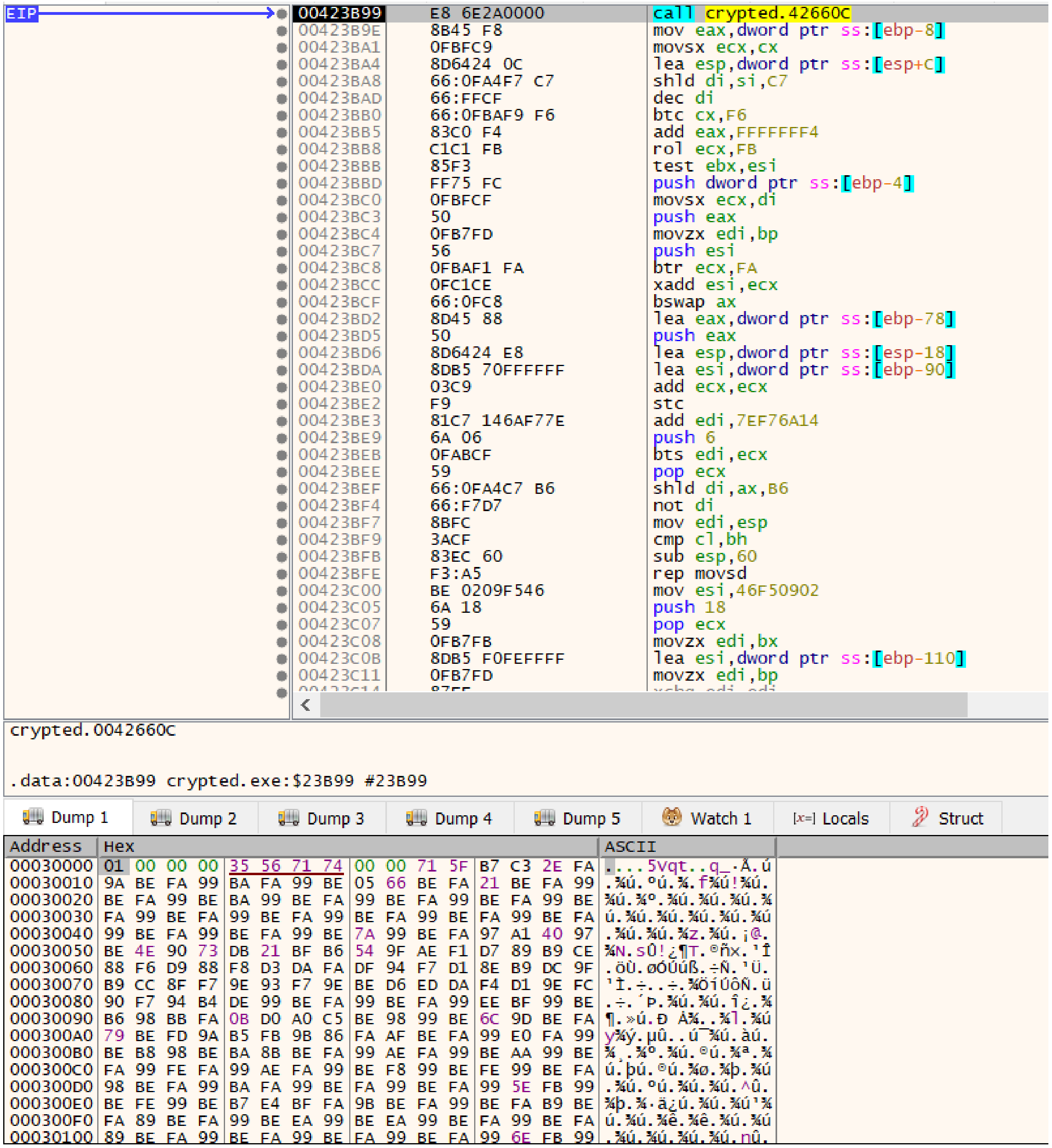
The code being passed as a callback function

The code is put into a layer of self decryption loop, after which we jump into a very obfuscated region of code. After some (and by some I mean a lot) of manual analysis, the important code where the payload is decrypted is identified.



```
00422D5C  E9 95020000        jmp crypted.422FF6
00422D61  C745 E8 79F72D6F   mov dword ptr ss:[ebp-18],6F2DF779
00422D68  E9 FC2D0000        jmp crypted.425B69
00422D6D  0FC9               bswap ecx
00422D6F  6A 18              push 18
00422D71  F7D7               not edi
00422D73  66:0FCF            bswap di
00422D76  66:0FBECE          movsx cx,dh
00422D7A  59                 pop ecx
00422D7B  0FB7FC             movzx edi,sp
00422D7E  E9 CD070000        jmp crypted.423550
00422D83  0F84 05000000      je crypted.422D8E
00422D89  FF75 CC            push dword ptr ss:[ebp-34]
00422D8C  FFD3               call ebx
00422D8E  837D D0 00         cmp dword ptr ss:[ebp-30],0
00422D92  E9 A42E0000        jmp crypted.425C3B
00422D97  C745 F8 24AA4635   mov dword ptr ss:[ebp-8],3546AA24
00422D9E  E9 F23D0000        jmp crypted.426B95
00422DA3  68 3B3B2AA6        push A62A3B3B
00422DA8  E9 D00C0000        jmp crypted.423A7D
00422DAD  E8 C30B0000        call crypted.423975
00422DB2  2BC9               sub ecx,ecx
00422DB4  8DB5 F0FEFFFF      lea esi,dword ptr ss:[ebp-110]
00422DBA  41                 inc ecx
00422DBB  81D7 AF5F3229      adc edi,29325FAF
00422DC1  66:81CF 6A1D       or di,1D6A
00422DC6  66:1BFB            sbb di,bx
00422DC9  2BDB               sub ebx,ebx
00422DCB  4F                 dec edi
00422DCC  85C0               test eax,eax
00422DCE  0F45D9             cmovne ebx,ecx
00422DD1  83EC 60            sub esp,60
00422DD4  0FBFFC             movsx edi,sp
00422DD7  0FB7C9             movzx ecx,cx
00422DDA  F7D1               not ecx
00422DDC  6A 18              push 18
00422DDE  59                 pop ecx
00422DDF  66:F7D7            not di
00422DE2  8BFE               mov edi,esi
```

The decryption call:

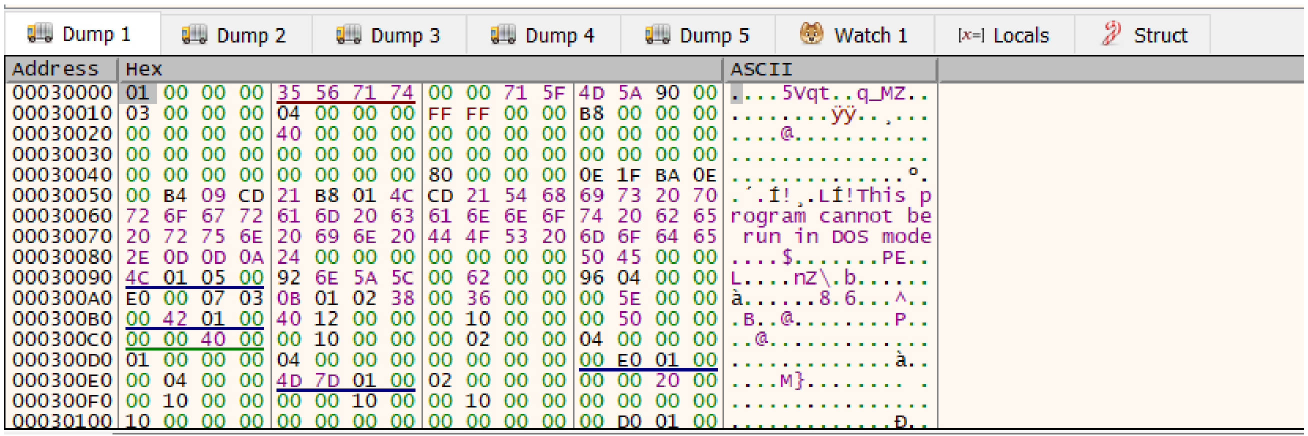


```
EIP  00423B99  E8 6E2A0000        call crypted.42660C
     00423B9E  8B45 F8            mov eax,dword ptr ss:[ebp-8]
     00423BA1  0FBFC9             movsx ecx,cx
     00423BA4  8D6424 0C          lea esp,dword ptr ss:[esp+C]
     00423BA8  66:0FA4F7 C7       shld di,si,C7
     00423BAD  66:FFCF            dec di
     00423BB0  66:0FBAF9 F6       btc cx,F6
     00423BB5  83C0 F4            add eax,FFFFFFF4
     00423BB8  C1C1 FB            rol ecx,FB
     00423BBB  85F3               test ebx,esi
     00423BBD  FF75 FC            push dword ptr ss:[ebp-4]
     00423BC0  0FBFCF             movsx ecx,di
     00423BC3  50                 push eax
     00423BC4  0FB7FD             movzx edi,bp
     00423BC7  56                 push esi
     00423BC8  0FBAF1 FA          btr ecx,FA
     00423BCC  0FC1CE             xadd esi,ecx
     00423BCF  66:0FC8            bswap ax
     00423BD2  8D45 88            lea eax,dword ptr ss:[ebp-78]
     00423BD5  50                 push eax
     00423BD6  8D6424 E8          lea esp,dword ptr ss:[esp-18]
     00423BDA  8DB5 70FFFFFF      lea esi,dword ptr ss:[ebp-90]
     00423BE0  03C9               add ecx,ecx
     00423BE2  F9                 stc
     00423BE3  81C7 146AF77E      add edi,7EF76A14
     00423BE9  6A 06              push 6
     00423BEB  0FABCF             bts edi,ecx
     00423BEE  59                 pop ecx
     00423BEF  66:0FA4C7 B6       shld di,ax,B6
     00423BF4  66:F7D7            not di
     00423BF7  8BFC               mov edi,esp
     00423BF9  3ACF               cmp cl,bh
     00423BFB  83EC 60            sub esp,60
     00423BFE  F3:A5              rep movsd
     00423C00  BE 0209F546        mov esi,46F50902
     00423C05  6A 18              push 18
     00423C07  59                 pop ecx
     00423C08  0FB7FB             movzx edi,bx
     00423C0B  8DB5 F0FEFFFF      lea esi,dword ptr ss:[ebp-110]
     00423C11  0FB7FD             movzx edi,bp

crypted.0042660C

.data:00423B99 crypted.exe:$23B99 #23B99
```

Dump 1 | Dump 2 | Dump 3 | Dump 4 | Dump 5 | Watch 1 | Locals | Struct

```
Address   Hex                                              ASCII
00030000  01 00 00 00 35 56 71 74 00 00 71 5F B7 C3 2E FA  ....5Vqt..q_·Ã.ú
00030010  9A BE FA 99 BA FA 99 BE 05 66 BE FA 21 BE FA 99  .¾ú.ºú.¾.f¾ú!¾ú.
00030020  BE FA 99 BE BA 99 BE FA 99 BE FA 99 BE FA 99 BE  ¾ú.¾º.¾ú.¾ú.¾ú.¾
00030030  FA 99 BE FA 99 BE FA 99 BE FA 99 BE FA 99 BE FA  ú.¾ú.¾ú.¾ú.¾ú.¾ú
00030040  99 BE FA 99 BE FA 99 BE 7A 99 BE FA 97 A1 40 97  .¾ú.¾ú.¾z.¾ú.¡@.
00030050  BE 4E 90 73 DB 21 BF B6 54 9F AE F1 D7 89 B9 CE  ¾N.sÛ!¿¶T.®ñ×.¹Î
00030060  88 F6 D9 88 F8 D3 DA FA DF 94 F7 D1 8E B9 DC 9F  .öÙ.øÓÚúß.÷Ñ.¹Ü.
00030070  B9 CC 8F F7 9E 93 F7 9E BE D6 ED DA F4 D1 9E FC  ¹Ì.÷..÷.¾ÖíÚôÑ.ü
00030080  90 F7 94 B4 DE 99 BE FA 99 BE FA 99 EE BF 99 BE  .÷.´Þ.¾ú.¾ú.î¿.¾
00030090  B6 98 BB FA 0B D0 A0 C5 BE 98 99 BE 6C 9D BE FA  ¶.»ú.Ð Å¾..¾l.¾ú
000300A0  79 BE FD 9A B5 FB 9B 86 FA AF BE FA 99 E0 FA 99  y¾ý.µû..ú¯¾ú.àú.
000300B0  BE B8 98 BE BA 8B BE FA 99 AE FA 99 BE AA 99 BE  ¾¸.¾º.¾ú.®ú.¾ª.¾
000300C0  FA 99 FE FA 99 AE FA 99 BE F8 99 BE FE 99 BE FA  ú.þú.®ú.¾ø.¾þ.¾ú
000300D0  98 BE FA 99 BA FA 99 BE FA 99 BE FA 99 5E FB 99  .¾ú.ºú.¾ú.¾ú.^û.
000300E0  BE FE 99 BE B7 E4 BF FA 9B BE FA 99 BE FA B9 BE  ¾þ.¾·ä¿ú.¾ú.¾ú¹¾
000300F0  FA 89 BE FA 99 BE EA 99 BE EA 99 BE FA 99 BE FA  ú.¾ú.¾ê.¾ê.¾ú.¾ú
00030100  89 BE FA 99 BE FA 99 BE FA 99 BE FA 99 6E FB 99  .¾ú.¾ú.¾ú.¾ú.nû.
```

The code decryption call



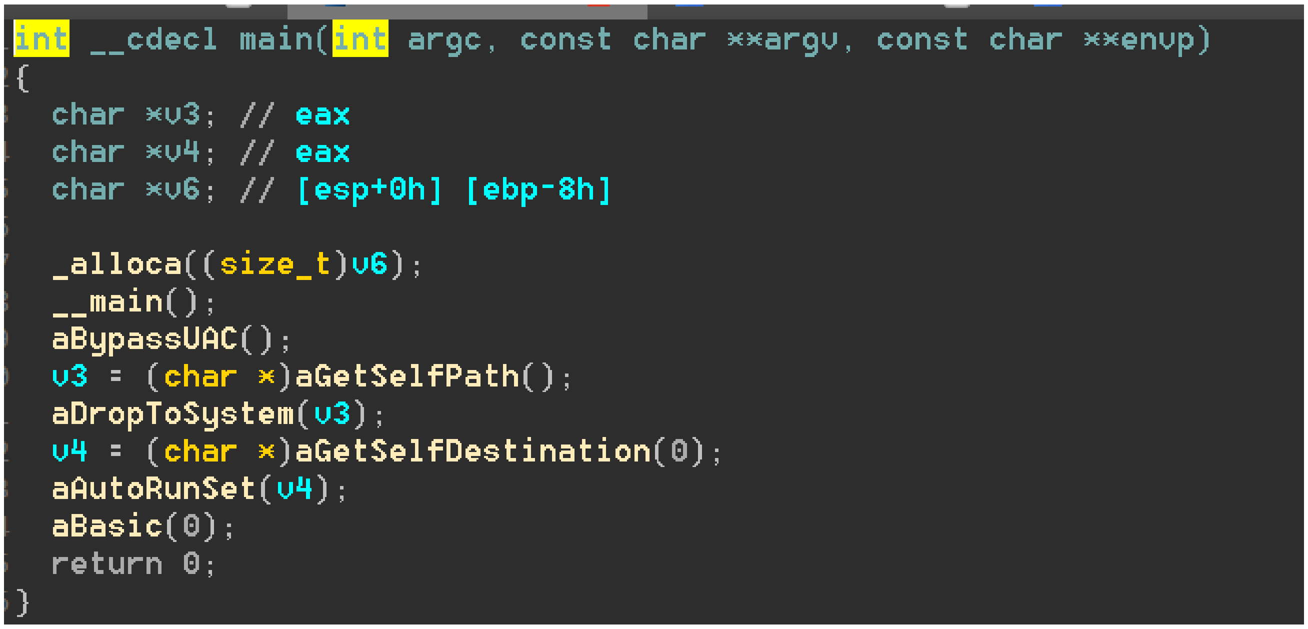
```
Address  Hex                                              ASCII
00030000 01 00 00 00 35 56 71 74 00 00 71 5F 4D 5A 90 00 ....5Vqt..q_MZ..
00030010 03 00 00 00 04 00 00 00 FF FF 00 00 B8 00 00 00 ........ÿÿ..¸...
00030020 00 00 00 00 40 00 00 00 00 00 00 00 00 00 00 00 ....@...........
00030030 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ................
00030040 00 00 00 00 00 00 00 00 80 00 00 00 0E 1F BA 0E ..............º.
00030050 00 B4 09 CD 21 B8 01 4C CD 21 54 68 69 73 20 70 .´.Í!¸.LÍ!This p
00030060 72 6F 67 72 61 6D 20 63 61 6E 6E 6F 74 20 62 65 rogram cannot be
00030070 20 72 75 6E 20 69 6E 20 44 4F 53 20 6D 6F 64 65  run in DOS mode
00030080 2E 0D 0D 0A 24 00 00 00 00 00 00 00 50 45 00 00 ....$.......PE..
00030090 4C 01 05 00 92 6E 5A 5C 00 62 00 00 96 04 00 00 L....nZ\.b......
000300A0 E0 00 07 03 0B 01 02 38 00 36 00 00 00 5E 00 00 à......8.6...^..
000300B0 00 42 01 00 40 12 00 00 00 10 00 00 00 50 00 00 .B..@........P..
000300C0 00 00 40 00 00 10 00 00 00 02 00 00 04 00 00 00 ..@.............
000300D0 01 00 00 00 04 00 00 00 00 00 00 00 00 E0 01 00 .............à..
000300E0 00 04 00 00 4D 7D 01 00 02 00 00 00 00 00 20 00 ....M}........ .
000300F0 00 10 00 00 00 00 10 00 00 10 00 00 00 00 00 00 ................
00030100 10 00 00 00 00 00 00 00 00 00 00 00 00 D0 01 00 .............Ð..
```

After the call

Stepping over the call, we see the region decrypted rather clearly. Dumping this, we get the actual payload.

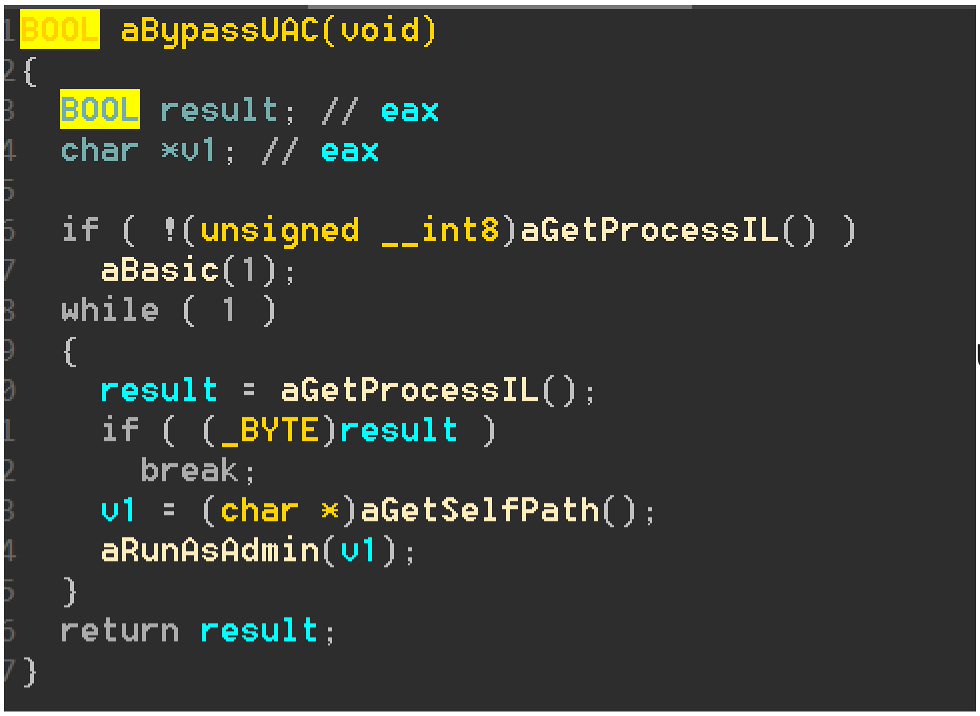
> SHA1: 3E4CD703DEEF2CFD1726095987766E2F062E9C57
>
> Compiler info: FreeBASIC Compiler v0.14 – 0.17

The malware in question is "Amadey", a new bot that is sold on a Russian forum. Link to thread content in Russian. It goes for $600 for a license, and for the high price cap the author is extra nice in his customer service – he delivered the symbols for us within the binary, allowing reverse engineers to inspect it with great ease :)



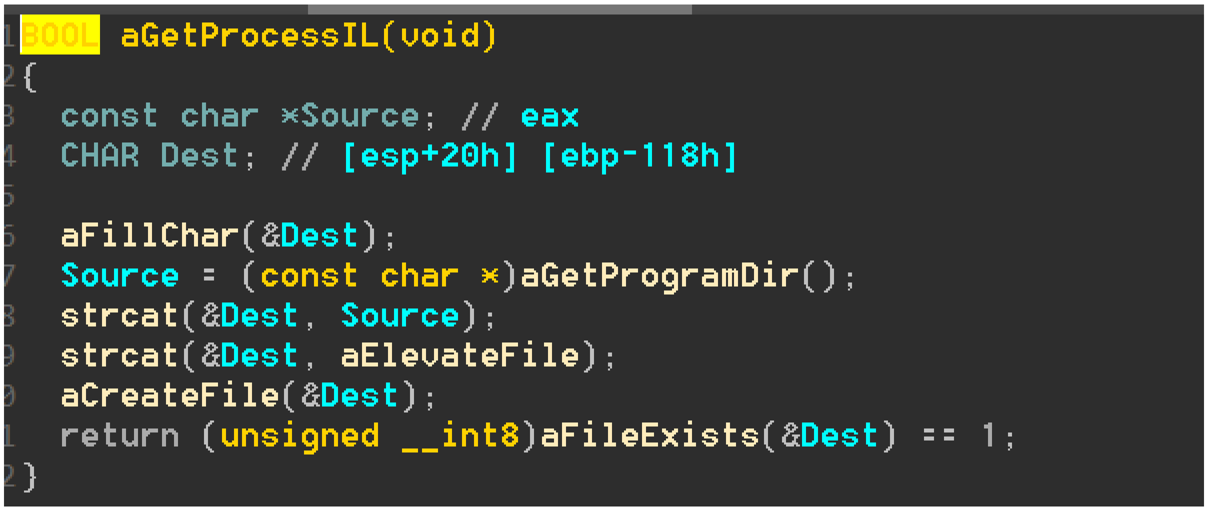
```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char *v3; // eax
  char *v4; // eax
  char *v6; // [esp+0h] [ebp-8h]

  _alloca((size_t)v6);
  __main();
  aBypassUAC();
  v3 = (char *)aGetSelfPath();
  aDropToSystem(v3);
  v4 = (char *)aGetSelfDestination(0);
  aAutoRunSet(v4);
  aBasic(0);
  return 0;
}
```

Main function



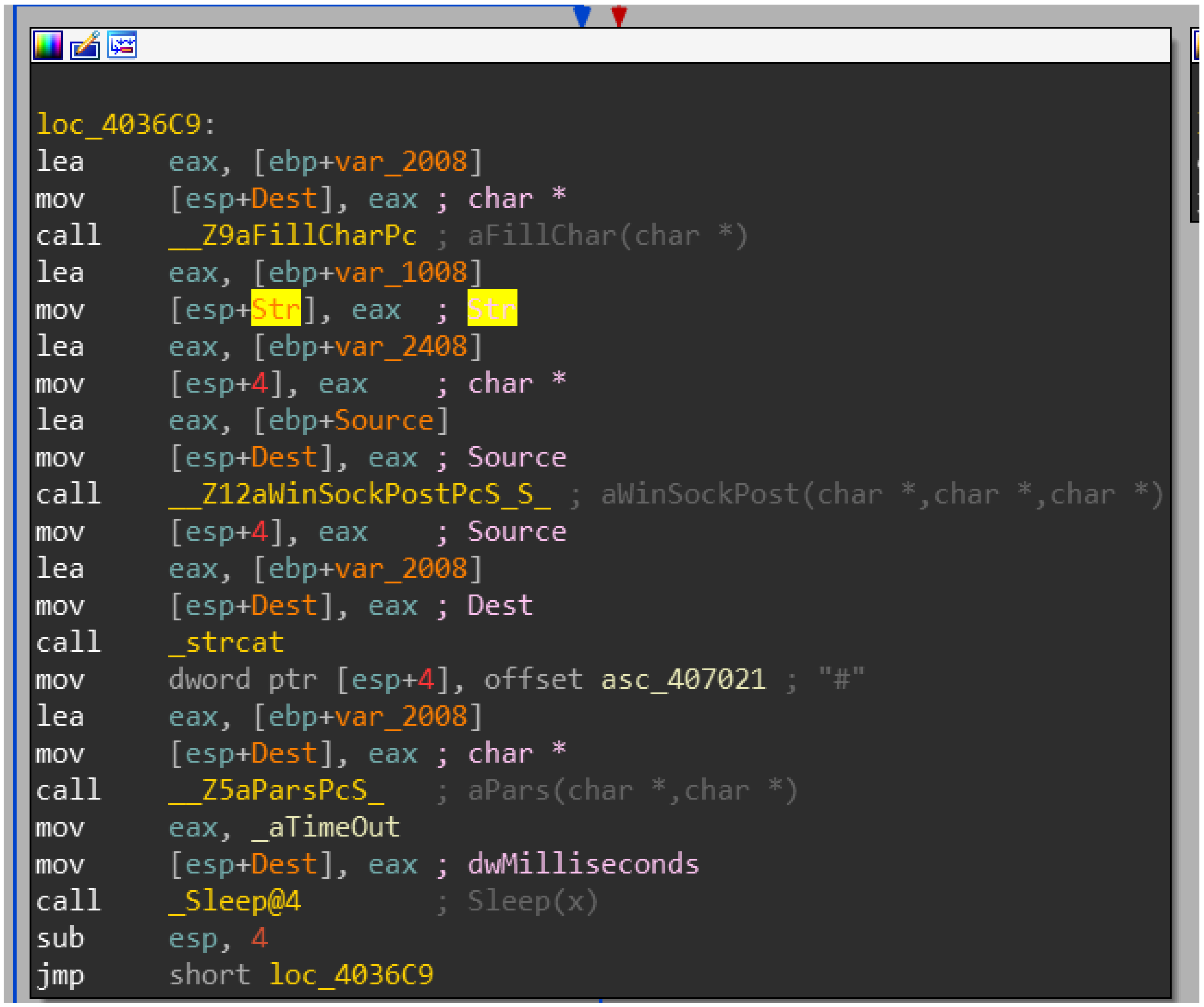
```
BOOL aBypassUAC(void)
{
  BOOL result; // eax
  char *v1; // eax

  if ( !(unsigned __int8)aGetProcessIL() )
    aBasic(1);
  while ( 1 )
  {
    result = aGetProcessIL();
    if ( (_BYTE)result )
      break;
    v1 = (char *)aGetSelfPath();
    aRunAsAdmin(v1);
  }
  return result;
}
```

UAC bypass just runs self as admin…



```
BOOL aGetProcessIL(void)
{
  const char *Source; // eax
  CHAR Dest; // [esp+20h] [ebp-118h]

  aFillChar(&Dest);
  Source = (const char *)aGetProgramDir();
  strcat(&Dest, Source);
  strcat(&Dest, aElevateFile);
  aCreateFile(&Dest);
  return (unsigned __int8)aFileExists(&Dest) == 1;
}
```

Terrible permission check by creating a file in a privileged folder
Startup is added by executing the command "REG ADD "HKCU\Software\Microsoft\Windows\CurrentVersion\Explorer\User Shell Folders" /f /v Startup /t REG_SZ /d path_to_folder_containing_the_file"


```
loc_4036C9:
lea     eax, [ebp+var_2008]
mov     [esp+Dest], eax ; char *
call    __Z9aFillCharPc ; aFillChar(char *)
lea     eax, [ebp+var_1008]
mov     [esp+Str], eax  ; Str
lea     eax, [ebp+var_2408]
mov     [esp+4], eax    ; char *
lea     eax, [ebp+Source]
mov     [esp+Dest], eax ; Source
call    __Z12aWinSockPostPcS_S_ ; aWinSockPost(char *,char *,char *)
mov     [esp+4], eax    ; Source
lea     eax, [ebp+var_2008]
mov     [esp+Dest], eax ; Dest
call    _strcat
mov     dword ptr [esp+4], offset asc_407021 ; "#"
lea     eax, [ebp+var_2008]
mov     [esp+Dest], eax ; char *
call    __Z5aParsPcS_   ; aPars(char *,char *)
mov     eax, _aTimeOut
mov     [esp+Dest], eax ; dwMilliseconds
call    _Sleep@4        ; Sleep(x)
sub     esp, 4
jmp     short loc_4036C9
```

The bot is not too interesting, it is in fact very simplistic. I would write more about the bot but…there is nothing else to write about. The programmer was nice enough to ship the file with symbols for us, making things a lot easier and in the process of doing so defeated the point of him encrypting strings. All files are available on virustotal and virusbay as usual.

View Comments ...