Forum: WD 9

Topic: DEBUG API Erreur: Descripteur non valide

Subject: Re: DEBUG API Erreur: Descripteur non valide

PostÃ© par: Anonyme

Contribution le : 12/3/2006 18:39:03

Bonsoir,

Continuant mes tests sur les DEBUG API, je me suis rendu compte que la structure déclarée était incorrecte:

_U est une structure uException est un EXCEPTION_DEBUG_INFO uCreateThread est un CREATE_THREAD_DEBUG_INFO uCreateProcessInfo est un CREATE_PROCESS_DEBUG_INFO uExitThread est un EXIT_THREAD_DEBUG_INFO uExitProcess est un EXIT_PROCESS_DEBUG_INFO uLoadDll est un LOAD_DLL_DEBUG_INFO uUnLoadDll est un UNLOAD_DLL_DEBUG_INFO uRipInfo est un RIP_INFO FIN DEBUG_EVENT est une structure dwDebugEventCode est un entier sans signe sur 4 octets//Type C : DWORD dwProcessId est un entier sans signe sur 4 octets //Type C : DWORD dwThreadId est un entier sans signe sur 4 octets //Type C : DWORD u est un _u

Je pensais que déclarer une structure transitoire ferait comme le mot clé "union" déclaré dans le fichier WINBASE.H:

typedef struct _DEBUG_EVENT { DWORD dwDebugEventCode; DWORD dwProcessId; DWORD dwThreadId; union { EXCEPTION_DEBUG_INFO Exception; CREATE_THREAD_DEBUG_INFO CreateThread; CREATE_PROCESS_DEBUG_INFO CreateProcessInfo; EXIT_THREAD_DEBUG_INFO ExitThread; EXIT_PROCESS_DEBUG_INFO ExitProcess; LOAD_DLL_DEBUG_INFO LoadDll; UNLOAD_DLL_DEBUG_INFO UnloadDll; OUTPUT_DEBUG_STRING_INFO DebugString; RIP_INFO RipInfo; } u; } DEBUG_EVENT, *LPDEBUG_EVENT;

Lors de la pratique, c'est autre chose. Je vais essayer d'etre le plus clair possible.
Prenons l'exemple de la structure _U déclarée plus haut: uException est mis en premier et plus loin nous trouvons uLoadDll. Je vais reprendre le corps du code de la boucle de debuggage:

TANTQUE bBoucle=Faux API("KERNEL32","WaitForDebugEvent",&dbEvent,DBG_INFINITE)
SI (dbEvent:dwProcessId = lpProcessInformation:dwProcessId) ALORS SELON dbEvent:dwDebugEventCode CAS EXCEPTION_DEBUG_EVENT SELON dbEvent:uException:ExceptionRecord:ExceptionCode CAS EXCEPTION_BREAKPOINT
//(1) L' evenement passe dans ce cas si uException est mis en 1er dans la structure _U.
//Pour info, dbEvent:uException:ExceptionRecord:ExceptionCode est bien égal à -2147483645 (0x80000003), //sinon, il est égal à 1 !! (Je ne sais pas pourquoi). FIN CAS CREATE_PROCESS_DEBUG_EVENT ListeAjoute(LIST_LOG,"Process Created: "+dbEvent:dwProcessId) CAS LOAD_DLL_DEBUG_EVENT //(2) dbEvent:uLoadDll:hFile est correct si uLoadDll est mis en 1er dans la structure _U, //sinon, dbEvent:uLoadDll:hFile est incorrect (soit 0, etc ...) CAS UNLOAD_DLL_DEBUG_EVENT ListeAjoute(LIST_LOG,"Dll Unload: "+dbEvent:uLoadDll:hFile) FIN SINON FIN API

("KERNEL32","ContinueDebugEvent",dbEvent:dwProcessId,dbEvent:dwThreadId,DBG_CONTINUE)
FIN

Je récapitule:
-> uLoadDll en 1er: le (2) correct mais pas le (1)
-> uException en 1er: le (1) correct mais pas le (2).

J'ai vérifié en codant le meme programme en C, mais en modifiant dans WINBASE.H la structure DEBUG_EVENT comme ceci (En fait j'ai oté le "union" en le remplacant par une structure: _U):

typedef struct _U { LOAD_DLL_DEBUG_INFO LoadDll; EXCEPTION_DEBUG_INFO Exception; CREATE_THREAD_DEBUG_INFO CreateThread; CREATE_PROCESS_DEBUG_INFO CreateProcessInfo; EXIT_THREAD_DEBUG_INFO ExitThread; EXIT_PROCESS_DEBUG_INFO ExitProcess; UNLOAD_DLL_DEBUG_INFO UnloadDll; OUTPUT_DEBUG_STRING_INFO DebugString; RIP_INFO RipInfo; } U, *LPU; typedef struct _DEBUG_EVENT { DWORD dwDebugEventCode; DWORD dwProcessId; DWORD dwThreadId; U u; } DEBUG_EVENT, *LPDEBUG_EVENT;

Et le plus surprenant, c'est que le programme réagi comme celui de windev suivant si LOAD_DLL_DEBUG_INFO LoadDll est en 1er ou non (idem pour EXCEPTION_DEBUG_INFO Exception).

Donc au final, je pense que la déclaration d'une structure n'est pas la bonne idée (Vous l'aviez deviné je pense ;))

Donc par quoi pourrais-je la remplacer ?

Merci d'avoir pris le temps de me lire.

Bonne soirée