

C/C++ Runtime Library Code Tampering in Software Supply Chain Attacks

Mohamad Mokbel

Trend Micro

CASE STUDY - SHADOWHAMMER

The actual implementation of the ShadowHammer poisoned function `__crtExitProcess()` resides inside the file `crt0dat.c` as follows:

```
void __cdecl __crtExitProcess (
    int status
)
{
    __crtCorExitProcess(status);

    /*
     * Either mscoree.dll isn't loaded,
     * or CorExitProcess isn't exported from mscoree.dll,
     * or CorExitProcess returned (should never happen).
     * Just call ExitProcess.
     */
    ExitProcess(status);
}
```

The CRT function `__crtCorExitProcess()` is responsible for checking if the process is part of a managed app, and if so, it calls the `CorExitProcess()`, otherwise it calls `ExitProcess()`. Said function is also defined in the `crt0dat.c`. The object file `crt0dat.obj` resides inside the library file `libcmt.lib`.

Contrast above benign implementation with ShadowHammer's implementation as shown in figure 1:

```
004F9736
004F9736
004F9736 ; Attributes: library function noreturn bp-based frame
004F9736 ; void __cdecl __noreturn __crtExitProcess(UINT uExitCode)
004F9736 __crtExitProcess proc near
004F9736 uExitCode= dword ptr 8
004F9736
004F9736 mov     edi, edi
004F9738 push   ebp
004F9739 mov     ebp, esp
004F973B push   [ebp+uExitCode]
004F973E call   malicious_code
004F9743 pop     ecx
004F9744 push   [ebp+uExitCode] ; uExitCode
004F9747 call   ds:ExitProcess
004F9747 __crtExitProcess endp
004F9747
```

Figure 1 ShadowHammer poisoned __crtExitProcess() runtime function

It is clear that the CRT function __crtExitProcess() was overwritten with a malicious function that contains the malware's shellcode (the call at address 0x004F973E). This is such an insidious modification that is very hard to detect.

Figure 2 shows the cross-reference graph of the __crtExitProcess() CRT function as referenced by the ShadowHammer compiled code. The graph shows all call paths (reachability) that lead to it, and all other calls it make itself. The actual call path that leads to executing ShadowHammer code is:

Start() -> __tmainCRTStartup() -> _fast_error_exit() -> __crtExitProcess() -> malicious_code()

The malicious_code() function is also reachable via the CRT functions, _malloc(), _doexit() and __mtinitlocknum().

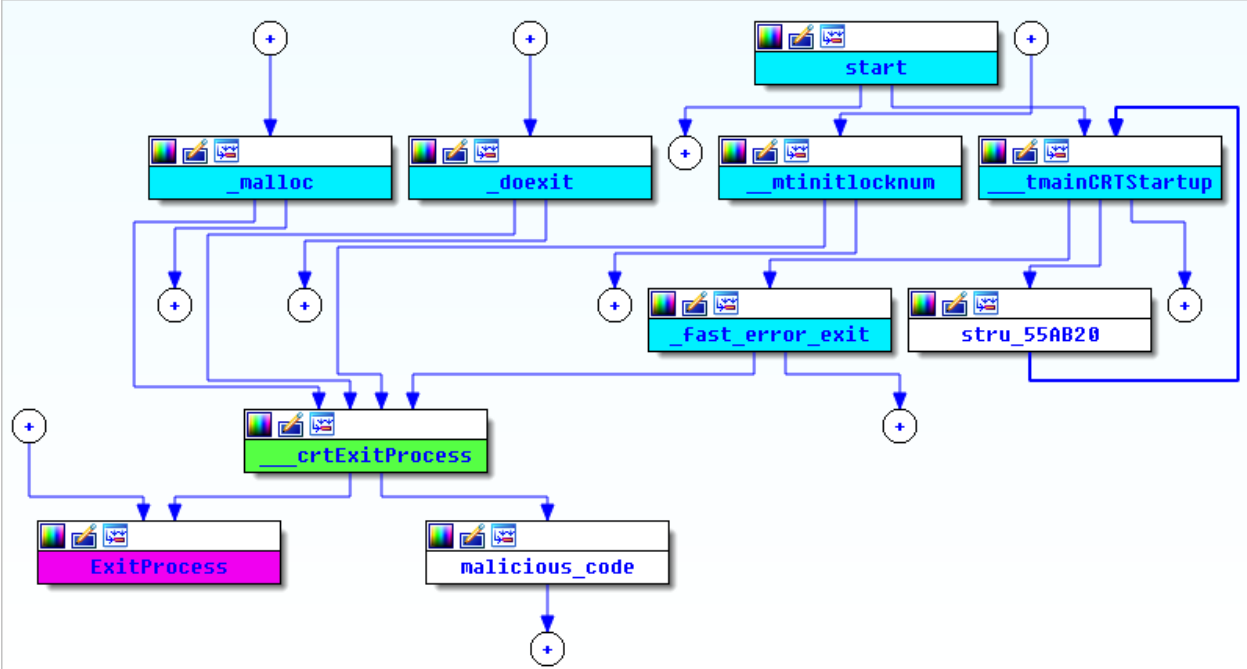


Figure 2 ShadowHammer poisoned function – call xref