When you break into a user-mode application in the kernel debugger, how do you connect a user-mode debugger?

devblogs.microsoft.com/oldnewthing/20160923-00

September 23, 2016

Raymond Chen

If you need to transfer control from one user-mode debugger to another, you can use a non-invasive debugger as a bridge. But what if you are broken into the kernel debugger, and you want to connect a user-mode debugger to the process? Since the system is broken into the kernel debugger, it is completely frozen. How do you unfreeze the system while still letting the crashed app sit in limbo?

What I do is patch in an infinite loop.

Change the bytes at the current instruction to EB FE, which as I noted some time ago is an unconditional short jump instruction, with an offset of −2. Since the jump target is calculated relative to the start of the instruction *after* the jump instruction, and since short jump instriction is itself two bytes long, jumping backward two bytes means that the jump instruction jumps to itself. In other words, you patched in an infinite loop.

Once that's done, resume execution in the kernel debugger. The system will roar back to life, except for the thread that crashed, which is now sitting and spinning. At this point, you can open a command prompt or whatever and connect the debugger of your choice. When that debugger breaks into the process, go find the thread that originally crashed, either by recognizing the thread by ID, or recognizing the stack, or by simply looking for the thread that is stuck in a tight infinite loop.

At this point, you can patch the original instruction back in and resume your debugging.

Note that this technique freezes only the thread that crashed. When you resume system execution, the threads that didn't crash will continue execution. Therefore, this solution works well if the problem involves only the thread that crashed or other threads which are not running (say, because they are blocked on a synchronization object). if the problem involves a thread that is running (say, because you have a race condition), then allowing those threads to continue execution means that you lose their state when the system resumes.

Exercise: Suppose there is no user-mode debugger installed on the system at all, and you can't install one. What can you do to at least get a user-mode dump of the crashed process, so you can analyze it offline?

Exercise: In the case where the problem involves multiple threads, what can you do to preserve the states of all the threads?

Raymond Chen

Follow