Why is my crash dump file filled with 0xAAAAAAAA?

devblogs.microsoft.com/oldnewthing/20161104-00

November 4, 2016

Raymond Chen

A customer was studying a minidump collected by Windows Error Reporting. The minidump includes the contents of the stack, but the contents are randomly filled with 0xAAAAAAAA.

```
00f3ac5c  00f3d1c0
00f3ac60  592ccae2 contoso!AppWndProc+0x1c5b
00f3ac64  aaaaaaaa
00f3ac68  aaaaaaaa
00f3ac6c  aaaaaaaa
00f3ac70  aaaaaaaa
00f3ac74  00000000
00f3ac78  00000000
00f3ac7c  58e75a46 contoso!WndProcGeneric
00f3ac80  504e7fea cohelp!allyourbuttons+0x5aba
00f3ac84  aaaaaaaa
00f3ac88  aaaaaaaa
00f3ac8c  00000000
00f3ac90  00000000
00f3ac94  0ee26838
00f3ac98  00000000
00f3ac9c  aaaaaaaa
00f3aca0  58ec7405 contoso!GetBlockBeforeCapture+0x2e
00f3aca4  0ee26838
00f3aca8  0fd6db10
00f3acac  00000000
00f3acb0  aaaaaaaa
00f3acb4  00f3ad04
00f3acb8  58ec732f contoso!FindDrawingFromGraphicId+0x136
00f3acbc  aaaaaaaa
00f3acc0  00000000
00f3acc4  00000000
00f3acc8  00000000
00f3accc  00000000
00f3acd0  aaaaaaaa
00f3acd4  aaaaaaaa
00f3acd8  aaaaaaaa
```

What’s going on here?

What's going on is that the minidump has been filtered. The customer missed this message from the debugger that was printed at the top of the debug session:

> User Mini Triage Dump File: Only registers, stack and portions of memory are available
>
> The user dump currently examined is a triage dump. Consequently, only a subset of debugger functionality will be available. If needed, please collect a minidump or a heap dump.
>
> - To create a mini user dump use the command: .dump /m <filename>
> - To create a full user dump use the command: .dump /ma <filename>
>
> Triage dumps have certain values on the stack and in the register contexts overwritten with pattern `0xAAAAAAAA`. If you see this value
>
> 1. the original value was not `NULL`
> 2. the original value was not a direct pointer to a loaded or unloaded image
> 3. the original value did not point to an object whose VFT points to a loaded or unloaded image (indirect pointer)
> 4. the original value did not point to the stack itself or any memory area added to the dump (TEB, PEB, memory for CLR stackwalk or exceptions, etc.)
> 5. the original value was not a valid handle value

After receiving this explanation, the customer was still a bit dubious. "A lot of function parameters in the dump are being given as `0xAAAAAAAA`, which suggests that they have been filtered out. But I thought constant strings and plain integers should still be on the stack. Does the fact that I don't see them mean that they were corrupted?"

If you look at the information banner printed by the debugger, you can see that plain integers are not on the list of things exempt from filtering. You might still see an integer if it happens to match a value that is exempt from filtering, such as if it happens to be zero or match a valid handle.

As for constant strings, it depends on how the constant string is stored. If it's a literal string embedded in a module, then it would be exempt from filtering according to rule 2. But if the string were copied to the heap or to the stack, then that would make it subject to filtering.

Raymond Chen

Follow