Implement stack walking without AsyncGetCallTrace #66

Open **apangin** opened this issue on Nov 2, 2017 · 4 comments

Labels **enhancement**

apangin commented on Nov 2, 2017 • edited ▾ Member

There is a number of problems with `AsyncGetCallTrace` API.

- [x] It is not accurate: sometimes it can point to neighbour methods. See #51 and JDK-8022893.
- [x] It incorrectly handles certain corner cases. See JDK-8178287.
- [x] It fails to walk through VM frames in Oracle JDK 9. See #60.
- [x] It requires preloading of all jmethodIDs thus resulting in startup time overhead. See #14 (comment).
- [ ] It does not distinguish between interpreted, compiled and inlined frames.
- [x] In rare cases it may crash JVM. See #73.
- [ ] `-XX:MaxJavaStackTraceDepth` unobviously affects collected stack traces. See #89.
- [ ] Unloaded methods appears in the profile as `jvmtiError 23`.
- [ ] Cannot walk stack traces from methods compiled with Graal.
- [x] = has workaround in async-profiler

The idea is to implement Java stack walking on our own without relying on AGCT. Since the agent runs in the context of JVM, it can access VM structures, especially those exported through VMStructs. It should be possible to replicate stack walking logic of the VM inside async-profiler, though it might be challenging. The main risk is that differrent versions of JVM may have different stack layout, but VMStructs along with special handling of the known versions is likely to help.