JET BRAINS

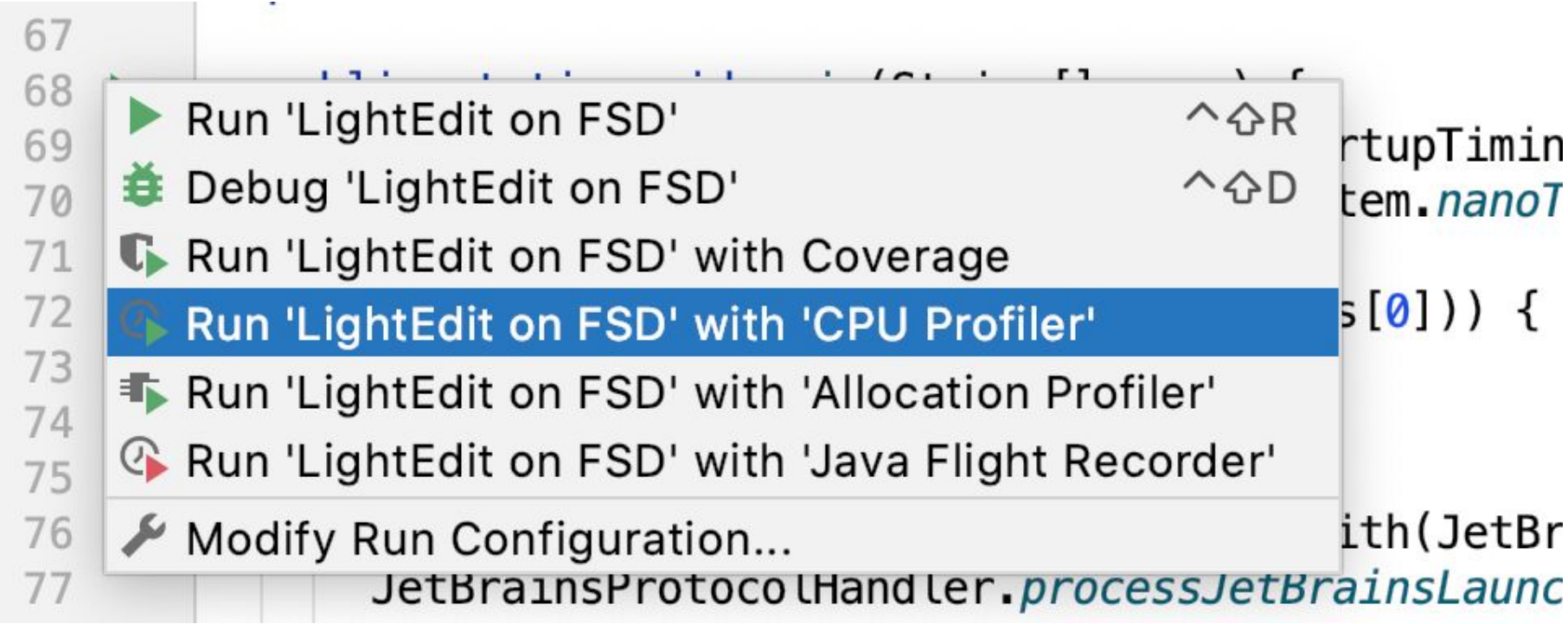
JVM-профайлер, который смог (стать кроссплатформенным)

Кирилл Тимофеев
kirill.timofeev@jetbrains.com

Зачем вообще это всё?

```
67
68      public static void main(String[] args) {
69        LinkedHashMap<@NonNls String, Long> startupTimin
70        startupTimings.put("startup begin", System.nanoT
71
72        if (args.length == 1 && "%f".equals(args[0])) {
73          args = NO_ARGS;
74        }
75
76        if (args.length == 1 && args[0].startsWith(JetBr
77          JetBrainsProtocolHandler.processJetBrainsLaunc
```

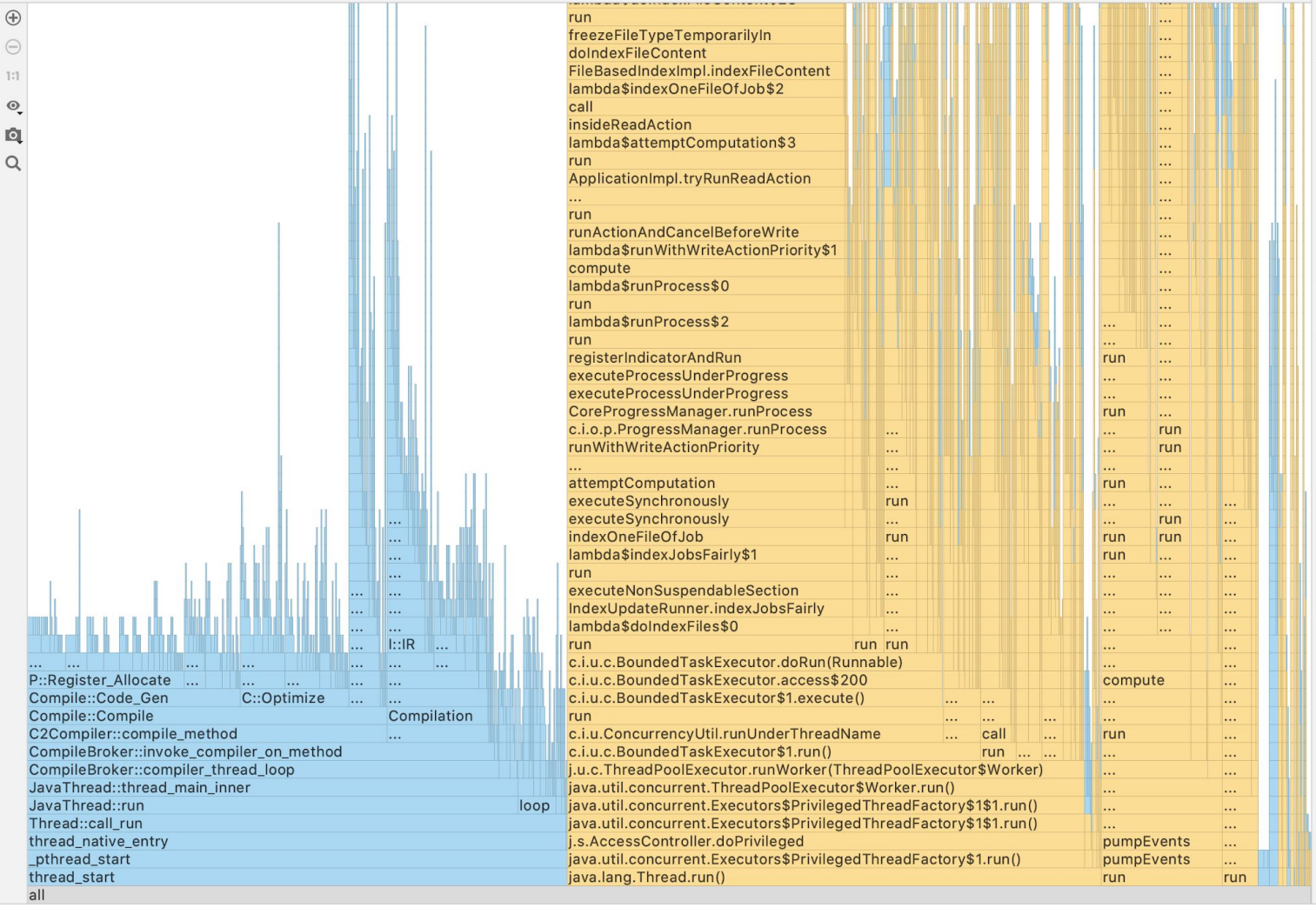
Зачем вообще это всё?



Flame Graph | Call Tree | Method List | CPU Timeline | Events

All threads merged

- AWT-AppKit id=25
- AWT-EventQueue-0 id=30
- AWT-Shutdown id=26
- AWTThreading pool-2-thread-1 id=36
- ApplicationImpl pooled thread 1 id=15
- ApplicationImpl pooled thread 10 id=38
- ApplicationImpl pooled thread 11 id=39
- ApplicationImpl pooled thread 12 id=40
- ApplicationImpl pooled thread 13 id=41
- ApplicationImpl pooled thread 14 id=42
- ApplicationImpl pooled thread 15 id=43
- ApplicationImpl pooled thread 16 id=44
- ApplicationImpl pooled thread 17 id=45
- ApplicationImpl pooled thread 18 id=46
- ApplicationImpl pooled thread 19 id=47
- ApplicationImpl pooled thread 2 id=16
- ApplicationImpl pooled thread 20 id=51
- ApplicationImpl pooled thread 3 id=17
- ApplicationImpl pooled thread 4 id=18
- ApplicationImpl pooled thread 5 id=19
- ApplicationImpl pooled thread 6 id=21
- ApplicationImpl pooled thread 7 id=22
- ApplicationImpl pooled thread 8 id=24
- ApplicationImpl pooled thread 9 id=37
- Attach Listener id=11
- Common-Cleaner id=8
- DefaultDispatcher-worker-1 id=76
- DestroyJavaVM id=13
- Finalizer id=3
- HandshakeCompletedNotify-Thread id=88
- I/O pool 1 id=48
- I/O pool 2 id=49
- I/O pool 3 id=50
- I/O pool 4 id=71
- I/O pool 5 id=74
- I/O pool 6 id=75
- Idea Main Thread id=12
- Java2D Disposer id=35
- Java2D Queue Flusher id=31



Flame Graph | Call Tree | Method List | CPU Timeline | Events

All threads merged

- AWT-AppKit id=25
- AWT-EventQueue-0 id=30
- AWT-Shutdown id=26
- AWTThreading pool-2-thread-1 id=36
- ApplicationImpl pooled thread 1 id=15
- ApplicationImpl pooled thread 10 id=38
- ApplicationImpl pooled thread 11 id=39
- ApplicationImpl pooled thread 12 id=40
- ApplicationImpl pooled thread 13 id=41
- ApplicationImpl pooled thread 14 id=42
- ApplicationImpl pooled thread 15 id=43
- ApplicationImpl pooled thread 16 id=44
- ApplicationImpl pooled thread 17 id=45
- ApplicationImpl pooled thread 18 id=46
- ApplicationImpl pooled thread 19 id=47
- ApplicationImpl pooled thread 2 id=16
- ApplicationImpl pooled thread 20 id=51
- ApplicationImpl pooled thread 3 id=17
- ApplicationImpl pooled thread 4 id=18
- ApplicationImpl pooled thread 5 id=19
- ApplicationImpl pooled thread 6 id=21
- ApplicationImpl pooled thread 7 id=22
- ApplicationImpl pooled thread 8 id=24
- ApplicationImpl pooled thread 9 id=37
- Attach Listener id=11
- Common-Cleaner id=8
- DefaultDispatcher-worker-1 id=76
- DestroyJavaVM id=13
- Finalizer id=3
- HandshakeCompletedNotify-Thread id=88
- I/O pool 1 id=48
- I/O pool 2 id=49
- I/O pool 3 id=50
- I/O pool 4 id=71
- I/O pool 5 id=74
- I/O pool 6 id=75
- Idea Main Thread id=12
- Java2D Disposer id=35
- Java2D Queue Flusher id=31

Method	Samples
42.1% thread_start	45,294
42.1% _pthread_start	45,294
41.9% thread_native_entry	45,168
41.9% Thread::call_run	45,166
38.2% ↓ JavaThread::run → JavaThread::thread_main_inner	41,124
36.8% CompileBroker::compiler_thread_loop	39,601
36.4% CompileBroker::invoke_compiler_on_method	39,197
28.1% C2Compiler::compile_method	30,215
7.9% Compiler::compile_method	8,479
3.0% GangWorker::loop	3,202
41.6% java.lang.**Thread.run**()	44,837
40.3% ↓ java.util.concurrent.**Executors$PrivilegedThreadFactory$1.run**() → java.security.**AccessController.doPrivileged**(PrivilegedAction, AccessContro	43,398
40.3% ↓ java.util.concurrent.**Executors$PrivilegedThreadFactory$1$1.run**() → java.util.concurrent.**ThreadPoolExecutor.runWorker**(ThreadPoolExecuto	43,396
32.2% com.intellij.util.concurrency.**BoundedTaskExecutor$1.run**()	34,681
29.3% com.intellij.util.**ConcurrencyUtil.runUnderThreadName**(String, Runnable)	31,561
29.3% com.intellij.util.concurrency.**BoundedTaskExecutor$1$$Lambda$478.1015774167.run**()	31,545
29.3% com.intellij.util.concurrency.**BoundedTaskExecutor$1.execute**()	31,543
29.3% com.intellij.util.concurrency.**BoundedTaskExecutor.access$200**(Runnable)	31,542
29.3% com.intellij.util.concurrency.**BoundedTaskExecutor.doRun**(Runnable)	31,542
2.9% com.intellij.util.concurrency.**BoundedTaskExecutor$1.execute**()	3,119
2.8% java.util.concurrent.**FutureTask.run**()	3,002
2.0% java.util.concurrent.**CompletableFuture$AsyncSupply.run**()	2,162
1.7% com.intellij.openapi.options.newEditor.**SettingsFilter$$Lambda$5509.513132836.run**()	1,832

Method	Samples ▼	Own Samples
com.intellij.util.concurrency.**BoundedTaskExecutor.doRun**(Runnable)	31,542	0
com.intellij.openapi.progress.impl.**CoreProgressManager.executeProcessUnderProgress**(Runnable, ProgressIndicator)	28,911	2
com.intellij.openapi.progress.impl.**ProgressManagerImpl.executeProcessUnderProgress**(Runnable, ProgressIndicator)	28,911	4
com.intellij.openapi.progress.impl.**CoreProgressManager.registerIndicatorAndRun**(ProgressIndicator, Thread, ProgressIndicator, Runnable)	28,909	12
com.intellij.openapi.progress.impl.**CoreProgressManager.runProcess**(Runnable, ProgressIndicator)	28,149	0
com.intellij.openapi.progress.impl.**CoreProgressManager$$Lambda$424.1827051599.run**()	28,140	0
com.intellij.openapi.progress.impl.**CoreProgressManager.lambda$runProcess$2**(ProgressIndicator, Runnable)	28,140	1
com.intellij.openapi.progress.util.**ProgressIndicatorUtils.runInReadActionWithWriteActionPriority**(Runnable, ProgressIndicator)	25,255	1
com.intellij.openapi.progress.**ProgressManager.runProcess**(Computable, ProgressIndicator)	25,254	0
com.intellij.openapi.progress.util.**ProgressIndicatorUtils.runWithWriteActionPriority**(Runnable, ProgressIndicator)	25,254	0
com.intellij.openapi.progress.util.**ProgressIndicatorUtils.lambda$runInReadActionWithWriteActionPriority$0**(Ref, Runnable)	25,244	0
com.intellij.openapi.progress.util.**ProgressIndicatorUtils$$Lambda$876.405815208.run**()	25,244	0
com.intellij.openapi.progress.**ProgressManager.lambda$runProcess$0**(Ref, Computable)	25,244	0
com.intellij.openapi.progress.util.**ProgressIndicatorUtils$$Lambda$878.943336498.compute**()	25,244	0
com.intellij.openapi.progress.util.**ProgressIndicatorUtils.lambda$runWithWriteActionPriority$1**(ApplicationEx, Runnable, Runnable)	25,244	0

Flame Graph | Call Tree | Method List | CPU Timeline | Events

All threads merged

AWT-AppKit id=25
AWT-EventQueue-0 id=30
AWT-Shutdown id=26
AWTThreading pool-2-thread-1 id=36
ApplicationImpl pooled thread 1 id=15
ApplicationImpl pooled thread 10 id=38
ApplicationImpl pooled thread 11 id=39
ApplicationImpl pooled thread 12 id=40
ApplicationImpl pooled thread 13 id=41
ApplicationImpl pooled thread 14 id=42
ApplicationImpl pooled thread 15 id=43
ApplicationImpl pooled thread 16 id=44
ApplicationImpl pooled thread 17 id=45
ApplicationImpl pooled thread 18 id=46
ApplicationImpl pooled thread 19 id=47
ApplicationImpl pooled thread 2 id=16
ApplicationImpl pooled thread 20 id=51
ApplicationImpl pooled thread 3 id=17
ApplicationImpl pooled thread 4 id=18
ApplicationImpl pooled thread 5 id=19
ApplicationImpl pooled thread 6 id=21
ApplicationImpl pooled thread 7 id=22
ApplicationImpl pooled thread 8 id=24
ApplicationImpl pooled thread 9 id=37
Attach Listener id=11
Common-Cleaner id=8
DefaultDispatcher-worker-1 id=76
DestroyJavaVM id=13
Finalizer id=3
HandshakeCompletedNotify-Thread id=88
I/O pool 1 id=48
I/O pool 2 id=49
I/O pool 3 id=50
I/O pool 4 id=71
I/O pool 5 id=74
I/O pool 6 id=75
Idea Main Thread id=12
Java2D Disposer id=35
Java2D Queue Flusher id=31

Method	Samples ▼	Own Samples
java.security.**AccessController.doPrivileged**(PrivilegedAction, AccessControlContext)	53,509	2
thread_start	45,299	0
_pthread_start	45,299	0
thread_native_entry	45,173	0
Thread::call_run	45,171	0
java.lang.**Thread.run**()	44,850	0
java.util.concurrent.**ThreadPoolExecutor$Worker.run**()	43,398	0
java.util.concurrent.**Executors$PrivilegedThreadFactory$1.run**()	43,398	0
java.util.concurrent.**ThreadPoolExecutor.runWorker**(ThreadPoolExecutor$Worker)	43,398	11
java.util.concurrent.**Executors$PrivilegedThreadFactory$1$1.run**()	43,396	0
java.util.concurrent.**Executors$PrivilegedThreadFactory$1$1.run**()	43,396	0
JavaThread::thread_main_inner	41,129	0
JavaThread::run	41,129	0
CompileBroker::compiler_thread_loop	39,601	4
CompileBroker::invoke_compiler_on_method	39,197	13
com.intellij.openapi.progress.impl.**CoreProgressManager.executeProcessUnderProgress**(Runnable, ProgressIndicator)	37,349	2
com.intellij.openapi.progress.impl.**ProgressManagerImpl.executeProcessUnderProgress**(Runnable, ProgressIndicator)	37,349	6
com.intellij.openapi.progress.impl.**CoreProgressManager.registerIndicatorAndRun**(ProgressIndicator, Thread, ProgressIndicator, Runnable)	37,342	18
com.intellij.util.concurrency.**BoundedTaskExecutor$1.run**()	34,681	0
com.intellij.util.concurrency.**BoundedTaskExecutor$1.execute**()	34,662	0
com.intellij.util.concurrency.**BoundedTaskExecutor.doRun**(Runnable)	34,661	0
com.intellij.util.concurrency.**BoundedTaskExecutor.access$200**(Runnable)	34,661	0
com.intellij.openapi.progress.impl.**CoreProgressManager.runProcess**(Runnable, ProgressIndicator)	33,464	0

Back Traces | Merged Callees | Callee List

☑ Collapse 218 recursive calls

- ↘100.0% 37,349 samples com.intellij.openapi.progress.impl.**CoreProgressManager.executeProcessUnderProgress**(Runnable, ProgressIndicator)
  - ↘ 99.9% 37,321 samples com.intellij.openapi.progress.impl.**CoreProgressManager.registerIndicatorAndRun**(ProgressIndicator, Thread, ProgressIndicator, Runnable)
    - ↘ 82.5% 30,804 samples com.intellij.openapi.progress.impl.**CoreProgressManager$$Lambda$424.1827051599.run**()
      - ↘ 82.5% 30,804 samples com.intellij.openapi.progress.impl.**CoreProgressManager.lambda$runProcess$2**(ProgressIndicator, Runnable)
      - ↘ 6.2% 2,302 samples com.intellij.codeInsight.daemon.impl.**PassExecutorService$ScheduledPass$$Lambda$4712.852618057.run**()
      - ↘ 5.1% 1,923 samples com.intellij.serviceContainer.**ServiceComponentAdapter$doCreateInstance$1.run**()
      - ↘ 2.0% 764 samples com.intellij.psi.stubs.**StubUpdatingIndex$$Lambda$903.558875044.run**()
      - ↘ 1.7% 648 samples com.intellij.concurrency.**ApplierCompleter$$Lambda$4774.248446060.run**()
      - ↘ < 1% 354 samples com.intellij.serviceContainer.**ComponentManagerImpl$getOrCreateLightService$1.run**()
      - ↘ < 1% 146 samples com.intellij.openapi.progress.impl.**CoreProgressManager$$Lambda$2781.1341166794.run**()
      - ↘ < 1% 143 samples com.intellij.util.indexing.**ChangedFilesCollector$$Lambda$2687.828064720.run**()
      - ↘ < 1% 126 samples com.intellij.psi.impl.source.tree.**LazyParseableElement$$Lambda$3077.442629903.run**()
      - ↘ < 1% 17 samples com.intellij.psi.stubs.**StubIndexImpl$$Lambda$3235.751504310.run**()
      - ↘ < 1% 14 samples java.util.**HashMap.computeIfAbsent**(Object, Function)


Flame Graph
Call Tree
Method List
CPU Timeline
Events
0 sec
45 sec
1 min
1 min, 15 sec
1 min, 30 sec
AWT-AppKit id=25
AWT-EventQueue-0 id=30
AWT-Shutdown id=26
AWTThreading pool-2-thread-1 id=36
ApplicationImpl pooled thread 1 id=15
ApplicationImpl pooled thread 10 id=38
ApplicationImpl pooled thread 11 id=39
ApplicationImpl pooled thread 12 id=40
ApplicationImpl pooled thread 13 id=41
ApplicationImpl pooled thread 14 id=42
ApplicationImpl pooled thread 15 id=43
ApplicationImpl pooled thread 16 id=44
ApplicationImpl pooled thread 17 id=45
ApplicationImpl pooled thread 18 id=46
ApplicationImpl pooled thread 19 id=47
ApplicationImpl pooled thread 2 id=16
ApplicationImpl pooled thread 20 id=51
ApplicationImpl pooled thread 3 id=17
ApplicationImpl pooled thread 4 id=18
ApplicationImpl pooled thread 5 id=19
ApplicationImpl pooled thread 6 id=21
ApplicationImpl pooled thread 7 id=22
ApplicationImpl pooled thread 8 id=24
ApplicationImpl pooled thread 9 id=37
Attach Listener id=11
Common-Cleaner id=8
DefaultDispatcher-worker-1 id=76
DestroyJavaVM id=13
Finalizer id=3
HandshakeCompletedNotify-Thread id=88
I/O pool 1 id=48
I/O pool 2 id=49
I/O pool 3 id=50
I/O pool 4 id=71
I/O pool 5 id=74
I/O pool 6 id=75
Idea Main Thread id=12
Java2D Disposer id=35
Java2D Queue Flusher id=31

Какой бы профайлер выбрать?

	Точный*	Нативные стеки	Работает на Windows	Бесплатный
Visual VM	–	–	+	+
Java Flight Recorder	+/–	–	+	+
Любой* платный профайлер	+/–	–	+	–
async-profiler	+	+	–	+
Написать свой	???	???	???	:)

- Андрей Паньгин, Safepoint — и пусть весь мир подождёт: https://youtu.be/rthWVvU9gWo
- Nitsan Wakart, Profilers are lying hobbitses: https://youtu.be/7IkHIqPeFjY

Какой бы профайлер выбрать?

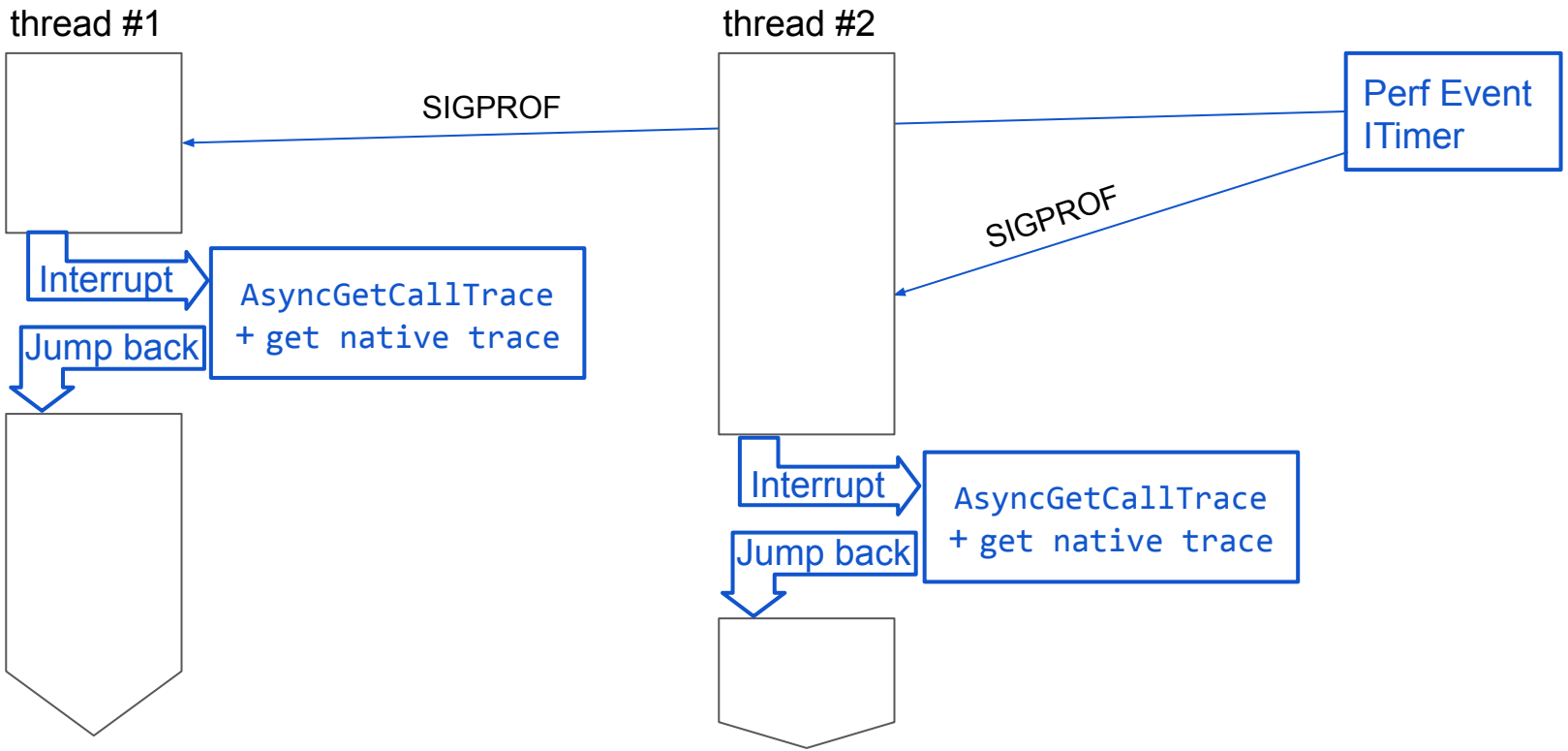
	Точный*	Нативные стеки	Работает на Windows	Бесплатный
Visual VM	–	–	+	+
Java Flight Recorder	+/–	–	+	+
Любой* платный профайлер	+/–	–	+	–
async-profiler	+	+	–	+
Написать свой	???	???	???	:)

- Андрей Паньгин, Safepoint — и пусть весь мир подождёт: https://youtu.be/rthWVvU9gWo
- Nitsan Wakart, Profilers are lying hobbitses: https://youtu.be/7IkHIqPeFjY

Что такое async-profiler

- Спасибо @AndreiPangin
- Основан на AsyncGetCallTrace и POSIX сигналах
- Фиксы пролога/эпилога и других проблем раскрутки стека
- Нативные стеки, Kernel символы на линуксе, perf events, трекинг TLAB аллокаций, итд...
- Записывает jfr дампы, которые можно открыть внутри IntelliJ IDEA

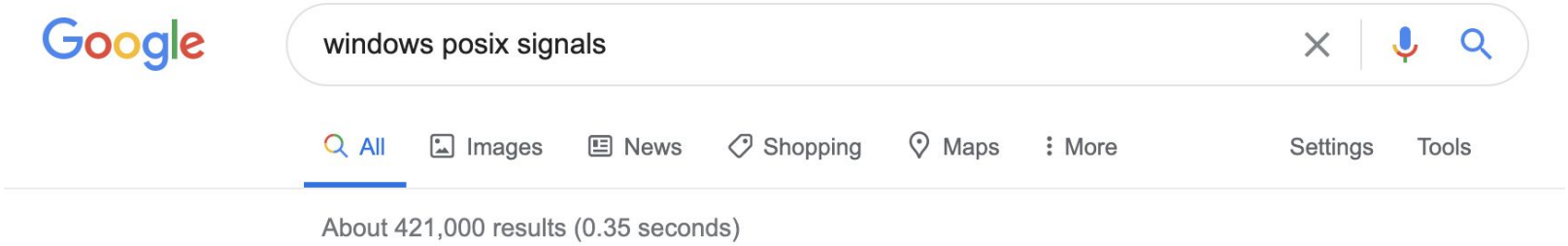
Что такое async-profiler



А почему вообще POSIX сигналы

- Единственный способ средствами ОС остановить тред
- AsyncGetCallTrace создан для вызова из обработчика сигнала
- Обработчик выполняет свой код в прерванном треде

Windows POSIX signals google



Windows POSIX signals google

Windows is not **POSIX**. It does not have **signals**. The only '**signals**' that console programs get is if they call SetConsoleCtrlHandler , in which case it can be notified that the user has pressed Ctrl+C, Ctrl+Break, closed the console window, logged off, or shut the system down. Sep 26, 2008

stackoverflow.com › questions › sending-an-arbitrary-sig...

Sending an arbitrary Signal in Windows? - Stack Overflow

About Featured Snippets Feedback

POSIX signals с точки зрения программиста

Написали обработчик:

```
void signalHandler(int signo, siginfo_t* siginfo, void* ucontext) {
   //actually call AsyncGetCallTrace somewhere inside
   Profiler::_instance.recordSample(ucontext);
}
```

POSIX signals с точки зрения программиста

Написали обработчик:

```
void signalHandler(int signo, siginfo_t* siginfo, void* ucontext) {
   //actually call AsyncGetCallTrace somewhere inside
   Profiler::_instance.recordSample(ucontext);
}
```

Зарегистрировали:

```
void setupHandler() {
   struct sigaction sa{};
   sigemptyset(&sa.sa_mask);
   sa.sa_sigaction = signalHandler;
   sa.sa_flags = SA_SIGINFO | SA_RESTART;
   sigaction(SIGPROF, &sa, NULL);
}
```

POSIX signals с точки зрения программиста

Написали обработчик:

```
void signalHandler(int signo, siginfo_t* siginfo, void* ucontext) {
   //actually call AsyncGetCallTrace somewhere inside
   Profiler::_instance.recordSample(ucontext);
}
```

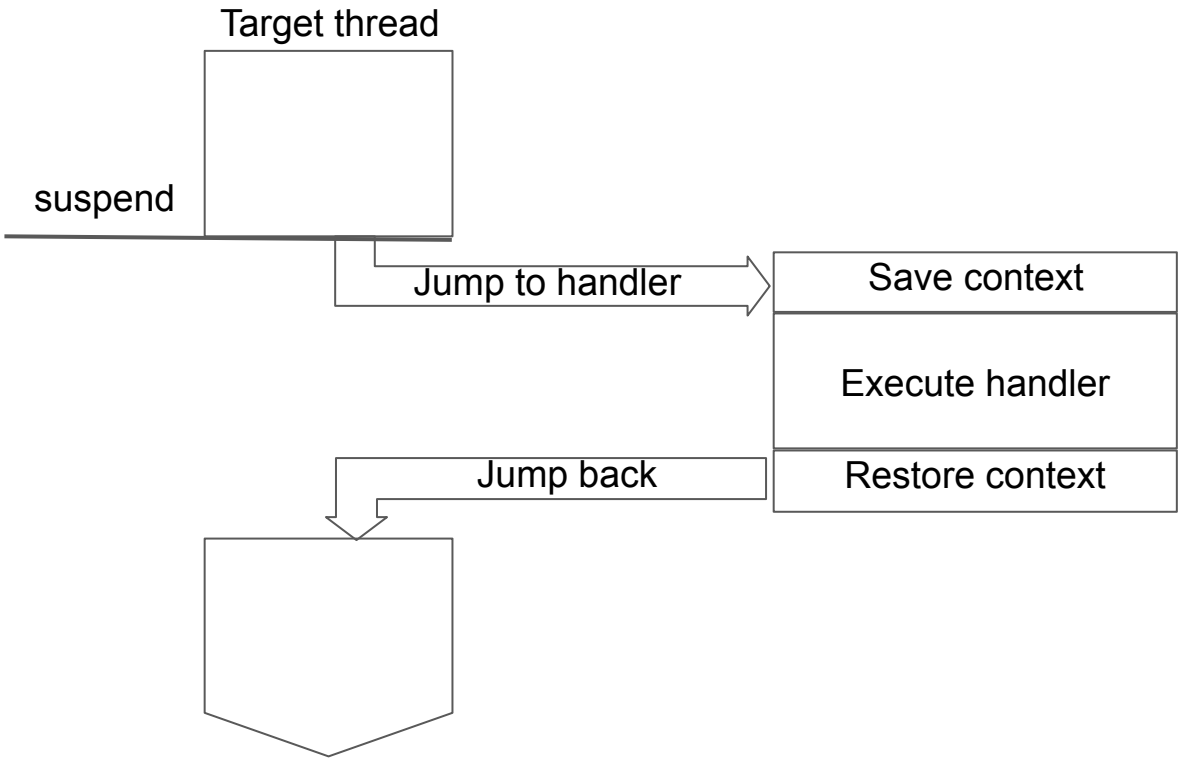
Зарегистрировали:

```
void setupHandler() {
   struct sigaction sa{};
   sigemptyset(&sa.sa_mask);
   sa.sa_sigaction = signalHandler;
   sa.sa_flags = SA_SIGINFO | SA_RESTART;
   sigaction(SIGPROF, &sa, NULL);
}
```

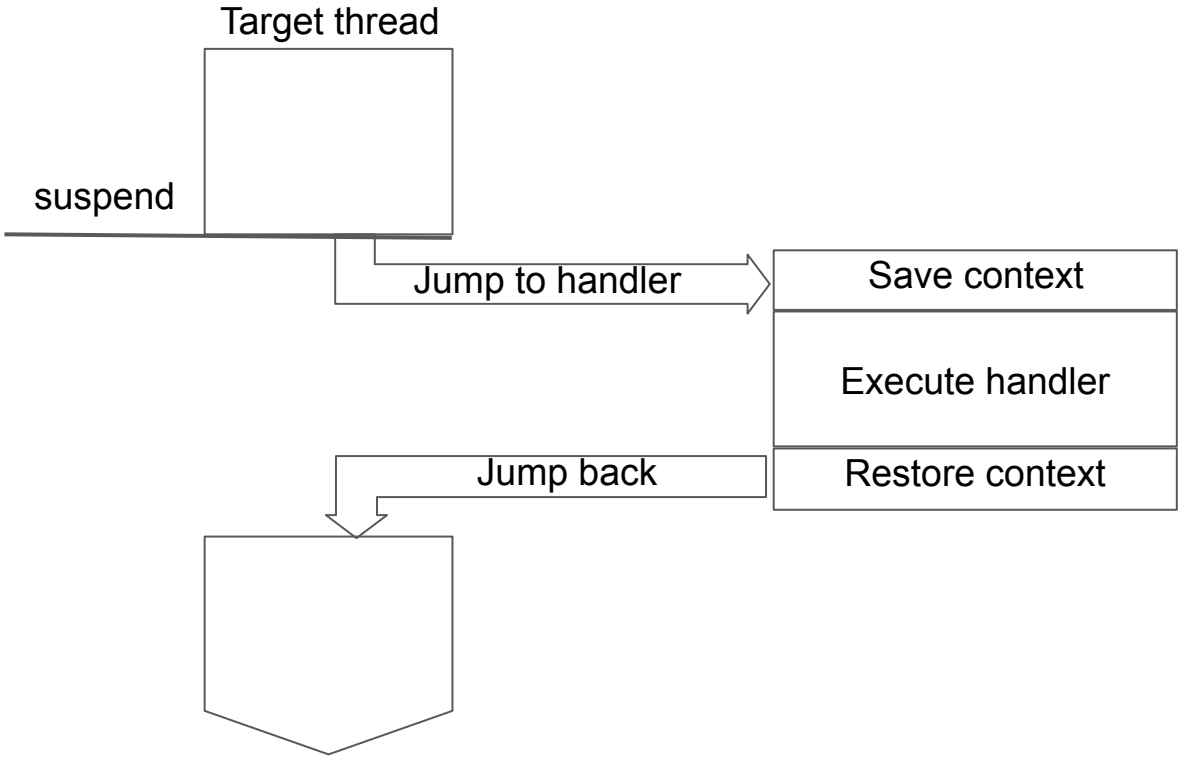
Вызываем из нашего процесса:

```
pthread_kill(tid, SIGPROF);
```

POSIX signals с точки зрения ОС



Давайте сделаем то же самое на Windows руками

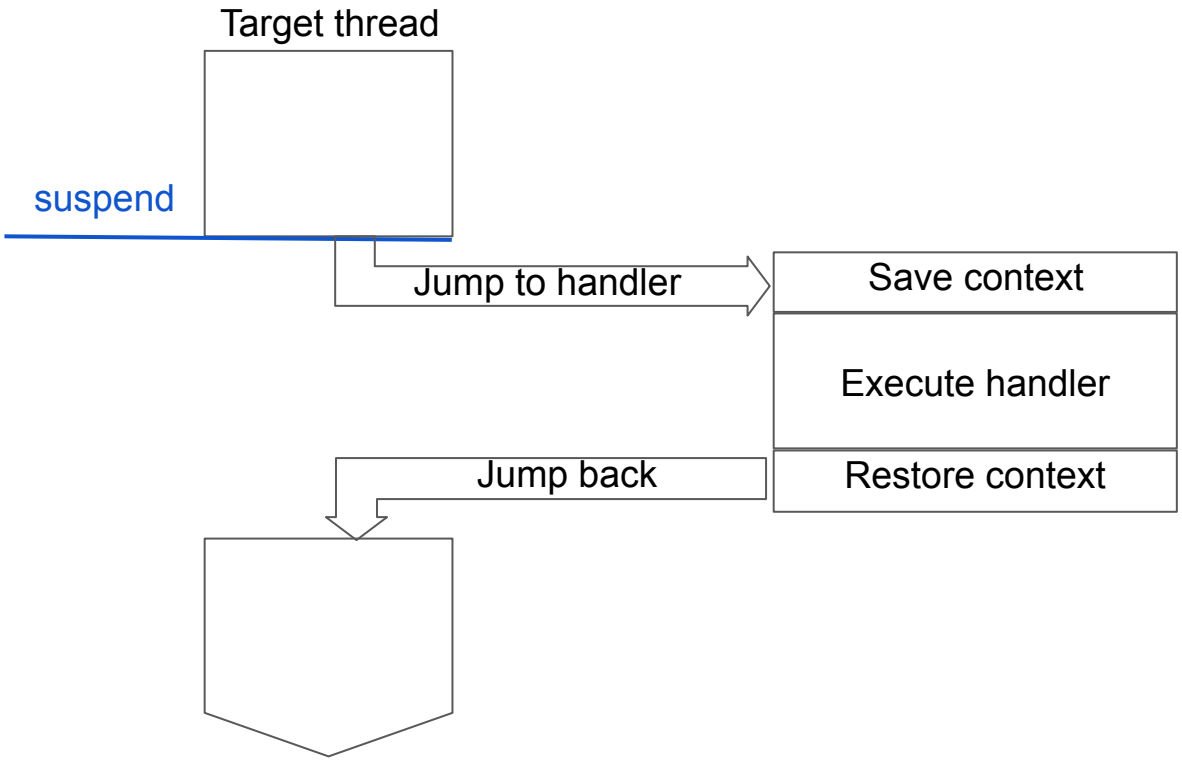


Оффтоп: про незаметное изменение вашего кода

Примеры из Java мира:

- Байткод трансформация агентами: ASM и Byte Buddy
- Spring Proxy
- Aspect-oriented programming
- Кодогенерация во время сборки проекта

Останавливаем поток



Останавливаем поток

- `DWORD SuspendThread(HANDLE hThread)`

Останавливаем поток

- `DWORD SuspendThread(HANDLE hThread)`
- `BOOL  GetThreadContext(HANDLE hThread, LPCONTEXT lpContext)`

https://devblogs.microsoft.com/oldnewthing/20150205-00/?p=44743

А что за LPCONTEXT?

A pointer to a CONTEXT structure that receives the appropriate context of the specified thread.

The CONTEXT structure is highly processor specific.

Однако, нас интересует только x86-64.

Refer to the WinNT.h header file for processor-specific definitions of this structures and any alignment requirements.

https://docs.microsoft.com/en-us/windows/win32/api/processthreadsapi/nf-processthreadsapi-getthreadcontext

Так обратимся же к WinNT.h

```
typedef struct DECLSPEC_ALIGN(16)
_CONTEXT {
// ...
// Control flags.
DWORD ContextFlags;
DWORD MxCsr;
// Segment Registers
// and processor flags.
WORD   SegCs, SegDs, SegEs;
WORD   SegFs, SegGs, SegSs;
DWORD  EFlags;
// Debug registers
DWORD64 Dr0, Dr1, Dr2;
DWORD64 Dr3, Dr6, Dr7;
// Special debug control registers.
DWORD64 DebugControl;
DWORD64 LastBranchToRip;
DWORD64 LastBranchFromRip;
DWORD64 LastExceptionToRip;
DWORD64 LastExceptionFromRip;
```

```
// Integer registers.
DWORD64 Rax, Rcx, Rdx;
DWORD64 Rbx, Rsp, Rbp;
DWORD64 Rsi, Rdi;
DWORD64 R8, R9, R10, R11;
DWORD64 R12, R13, R14, R15;

// Program counter.
DWORD64 Rip;


// Vector registers.
M128A   VectorRegister[26];
DWORD64 VectorControl;
```

```
// Floating point state.
union {
    XMM_SAVE_AREA32 FltSave;
    struct {
        M128A Header[2];
        M128A Legacy[8];
        M128A Xmm0;
        M128A Xmm1;
        M128A Xmm2;
        M128A Xmm3;
        M128A Xmm4;
        M128A Xmm5;
        M128A Xmm6;
        M128A Xmm7;
        M128A Xmm8;
        M128A Xmm9;
        M128A Xmm10;
        M128A Xmm11;
        M128A Xmm12;
        M128A Xmm13;
        M128A Xmm14;
        M128A Xmm15;
    } DUMMYSTRUCTNAME;
} DUMMYUNIONNAME;
```

Два важных нам регистра

```
DWORD64 Rsp;

// Program counter.
DWORD64 Rip;
```

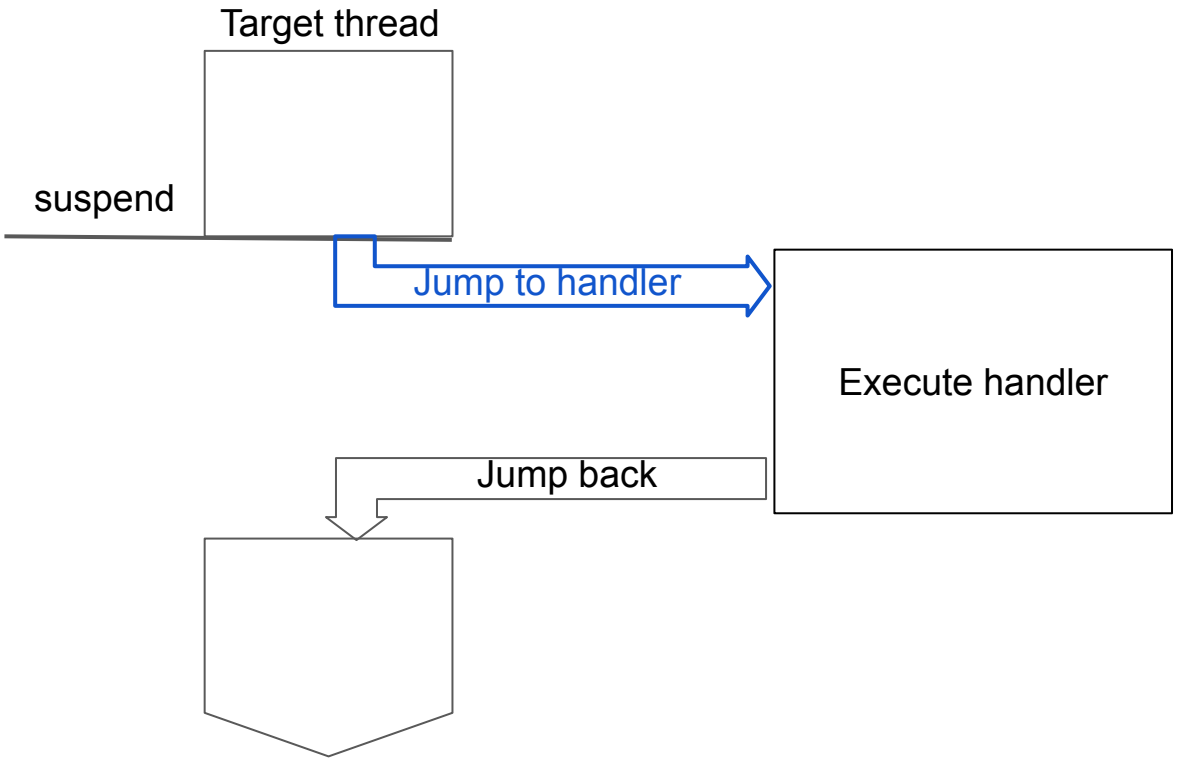
Два важных нам регистра

```
DWORD64 Rsp;

// Program counter.
DWORD64 Rip;
```

Два важных нам регистра

```
DWORD64 Rsp;

// Program counter.
DWORD64 Rip;
```

Передаём управление в обработчик



Передаём управление в обработчик

- `BOOL GetThreadContext(HANDLE hThread, LPCONTEXT lpContext)`

Передаём управление в обработчик

- `BOOL GetThreadContext(HANDLE hThread, LPCONTEXT lpContext)`
- `BOOL SetThreadContext(HANDLE hThread, const CONTEXT *lpContext)`

Передаём управление в обработчик

- `BOOL GetThreadContext(HANDLE hThread, LPCONTEXT lpContext)`
- `BOOL SetThreadContext(HANDLE hThread, const CONTEXT *lpContext)`

- `DWORD SuspendThread(HANDLE hThread)`

Передаём управление в обработчик

- `BOOL GetThreadContext(HANDLE hThread, LPCONTEXT lpContext)`
- `BOOL SetThreadContext(HANDLE hThread, const CONTEXT *lpContext)`

- `DWORD SuspendThread(HANDLE hThread)`
- `DWORD ResumeThread(HANDLE hThread)`

https://medium.com/tenable-techblog/api-series-setthreadcontext-d08c9f84458d

Передаём управление в обработчик [2]

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Передаём управление в обработчик [2]

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Передаём управление в обработчик [2]

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Передаём управление в обработчик [2]

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Передаём управление в обработчик [2]

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

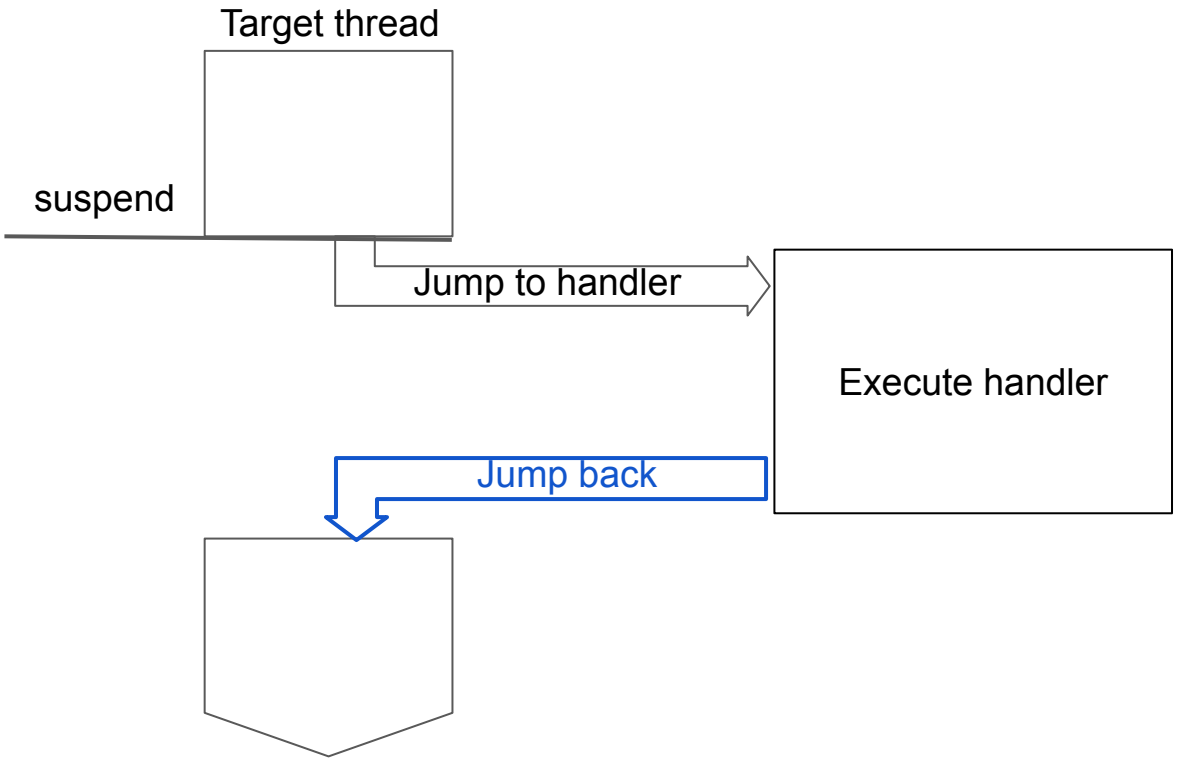
Передаём управление в обработчик [2]

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Передаём управление в обработчик [2]

```
void ourHandler() { printf("Hello world from thread %ld!\n", GetCurrentThreadId()); }

void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

А как возвращать управление программе?



А как возвращать управление программе? [1]

```
void ourHandler() { printf("Hello world from thread %ld!\n", GetCurrentThreadId()); }

void doCallHandler() {
   ourHandler();
}
```

А как возвращать управление программе? [2]

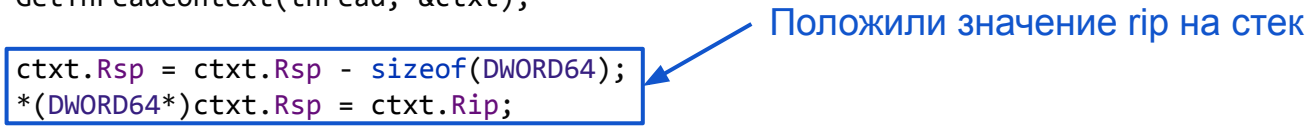
```
void ourHandler() { printf("Hello world from thread %ld!\n", GetCurrentThreadId()); }

void doCallHandler() {
   ourHandler();
}
```

```
(lldb) disas -n doCallHandler
jokerconf_sample.exe`doCallHandler():
    <+0>:  push   rdi
    <+2>:  sub    rsp, 0x20
    ....
    <+21>: call   0x7ff747e882c0 ; ourHandler()
    <+26>: add    rsp, 0x20
    <+30>: pop    rdi
    <+31>: ret
```

```
(lldb) disas -n ourHandler
jokerconf_sample.exe`ourHandler():
  <+0>:  push   rdi
  <+2>:  sub    rsp, 0x20
  ....
  <+28>: call   0x7ff747e8119f ; printf
  <+33>: add    rsp, 0x20
  <+37>: pop    rdi
  <+38>: ret
```

Передаём управление в обработчик (и обратно)

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = ctxt.Rip;
   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```



Передаём управление в обработчик (и обратно)

```
void doInterruptThread(HANDLE thread, void* handler) {
    SuspendThread(thread);
    CONTEXT ctxt;
    ctxt.ContextFlags = CONTEXT_FULL;
    GetThreadContext(thread, &ctxt);

    ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
    *(DWORD64*)ctxt.Rsp = ctxt.Rip;
    ctxt.Rip = (DWORD64) handler;

    SetThreadContext(thread, &ctxt);
    ResumeThread(thread);
}
```

Положили значение rip на стек

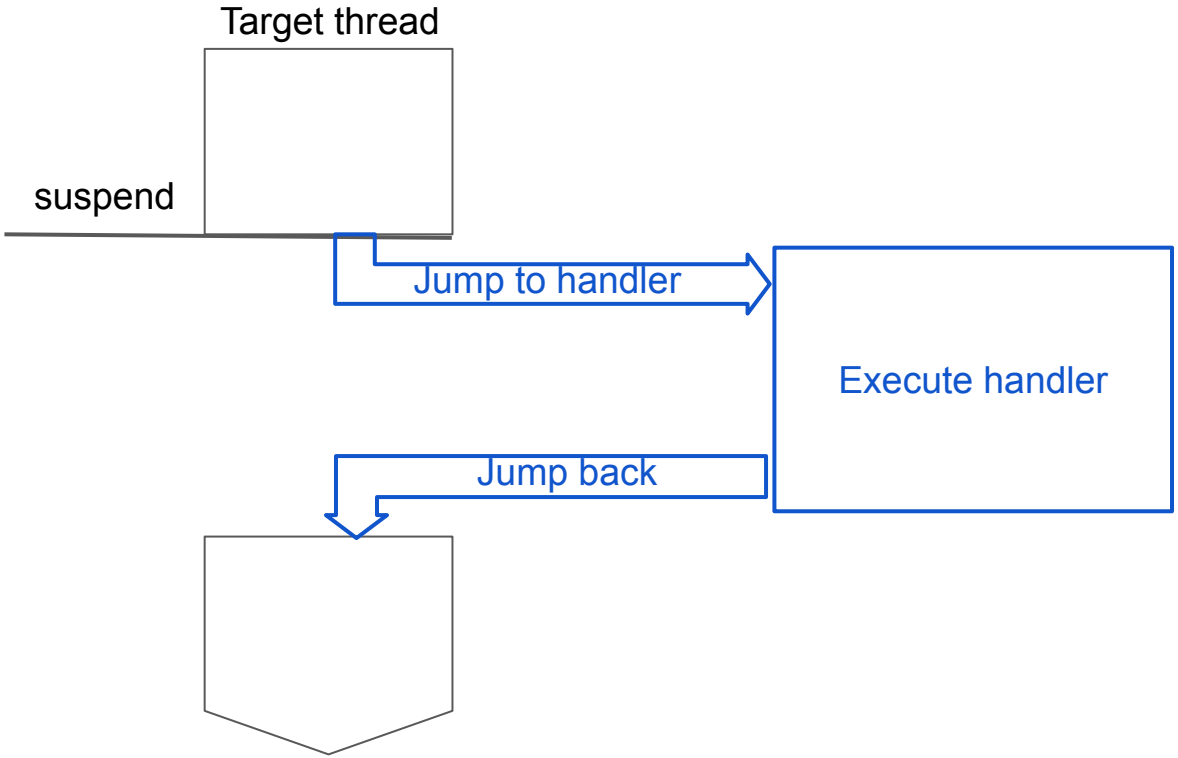
Передали управление по адресу

Передаём управление в обработчик (и обратно)

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);

   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = ctxt.Rip;
   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

call (DWORD64)handler

Передаём управление в обработчик (и обратно)



Вызываем наш обработчик

```
void ourHandler() { printf("Hello world from thread %ld!\n", GetCurrentThreadId()); }

void callOtherFunction(HANDLE myPoorLittleThread) {
   doInterruptThread(myPoorLittleThread, ourHandler);
}

int main() {
   HANDLE thread = threadHandle(GetCurrentThreadId());
   _beginthread(callOtherFunction, 0, thread);
   printf("main() is running on thread: %ld\n", GetCurrentThreadId());
   ull result = doWork();
   printf("result: %llu\n", result);
   return 0;
}
```

Вызываем наш обработчик [1]

```
c:\Work\jokerconf-sample\cmake-build-debug\jokerconf_sample.exe
main() is running on thread: 11612
Hello world from thread 11612!

Process finished with exit code -1073740972 (0xC0000354)
```

Вызываем наш обработчик [1]

УПС :(

```
c:\Work\jokerconf-sample\cmake-build-debug\jokerconf_sample.exe
main() is running on thread: 11612
Hello world from thread 11612!

Process finished with exit code -1073740972 (0xC0000354)
```

Что такое Calling Convention

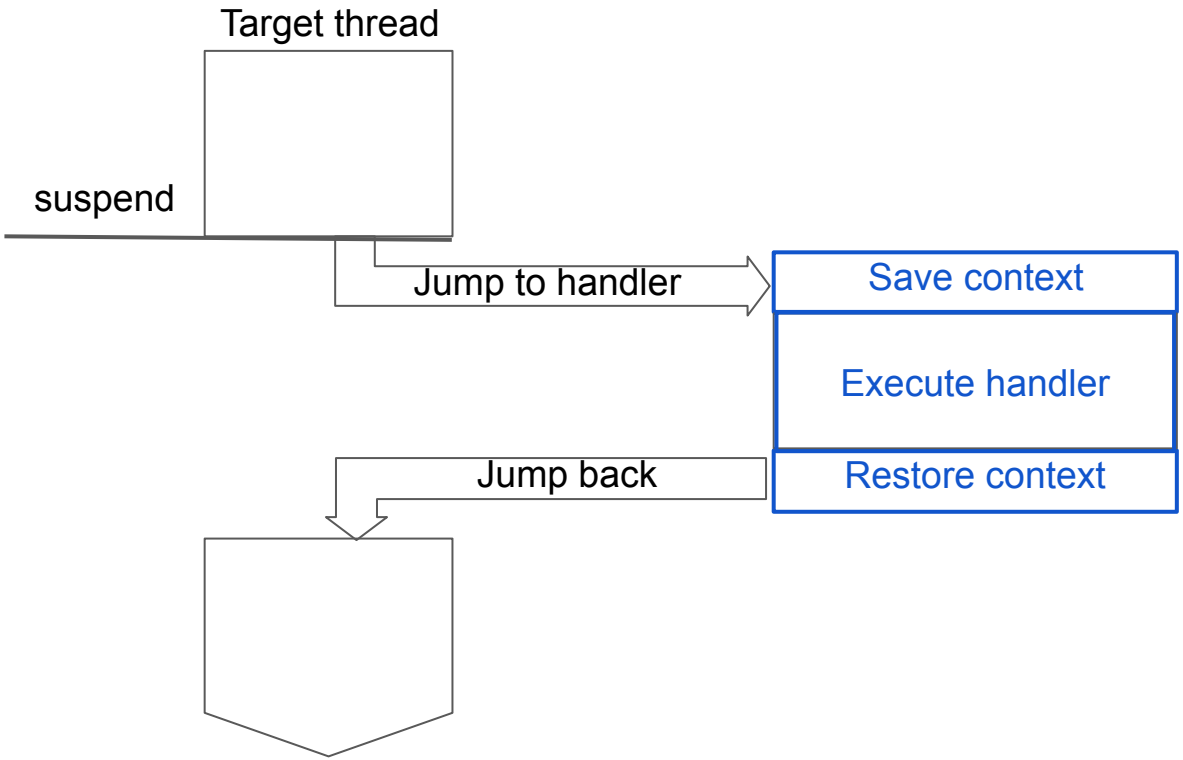
Описание технических особенностей вызова подпрограммы, определяющее:

- способы сохранения контекста перед вызовом подпрограммы
- способы передачи параметров подпрограммам
- способы вызова подпрограмм
- способы возврата результатов выполнения подпрограммы
- способы передачи управления из подпрограммы обратно в точку вызова

Microsoft x64 calling convention

- Нужно сохранять регистры(Volatile) , но можно не все (Nonvolatile)
- Нужно выравнивание на 0x10 байт стека перед вызовом
- Нужно выделить ещё 0x20 байта стека перед вызовом (shadow space)
- А если не выравнивать стек, то, может случится беда

Теперь сохраняем контекст



Теперь сохраняем контекст и вызываем функцию

```
pushfq
push  rax
push  rcx
push  rdx
push  r8
push  r9
push  r10
push  r11
push  r12
```

```
sub   rsp, 020h  ; выделяем shadow space
mov   r12, rsp   ; сохраняем rsp
and   rsp, -010h ; выравниваем rsp

; позже заменим на адрес нужной функции
mov   rax, 0101010101010101h
call rax

mov   rsp, r12   ; восстанавливаем rsp
add   rsp, 020h
```

```
pop   r12
pop   r11
pop   r10
pop   r9
pop   r8
pop   rdx
pop   rcx
pop   rax
popfq
```

А как нам выполнить произвольный обработчик?

```
void *loadHandler(HANDLE process, void *code, size_t code_size) {
   auto buffer = VirtualAlloc(nullptr, code_size,
                              MEM_COMMIT | MEM_RESERVE, PAGE_EXECUTE_READWRITE);
   WriteProcessMemory(process, buffer, code, code_size, nullptr);
   FlushInstructionCache(process, buffer, code_size);
   return buffer;
}
```

Теперь мы можем исполнять то, что было записано в `void *code`, ура

(примерно то же самое делает JIT компилятор)

А давайте попробуем ещё раз

```
void ourHandler() { printf("Hello world from thread %ld!\n", GetCurrentThreadId()); }

unsigned char code[] = { /**/ };

void callOtherFunction(HANDLE myPoorLittleThread) {
   putPointer(find_start(code, sizeof(code), 0x01, 8), ourHandler); // заменили на нужный адрес
   void *handler = loadHandler(getProcess(), &code, sizeof(code));
   doInterruptThread(myPoorLittleThread, handler);
}


int main() {
   HANDLE thread = threadHandle(GetCurrentThreadId());
   _beginthread(callOtherFunction, 0, thread);
   printf("main() is running on thread: %ld\n", GetCurrentThreadId());
   ull result = doWork();
   printf("result: %llu\n", result);
   return 0;
}
```

А давайте попробуем ещё раз [2]

```
c:\Work\jokerconf-sample\cmake-build-relwithdebinfo\jokerconf_sample.exe
main() is running on thread: 20220
Hello world from thread 20220!
result: 6526521903664095233

Process finished with exit code 0
```

А давайте попробуем ещё раз [2]

УРА!

```
c:\Work\jokerconf-sample\cmake-build-relwithdebinfo\jokerconf_sample.exe
main() is running on thread: 20220
Hello world from thread 20220!
result: 6526521903664095233

Process finished with exit code 0
```

Теперь всё работает, ура

Но, кажется, чего-то не хватает...

Теперь всё работает, ура

Но, кажется, чего-то не хватает...

наша функция:

```
void ourHandler();
```

POSIX обработчик:

```
void signalHandler(int signo, siginfo_t* siginfo, void* ucontext) {
   Profiler::_instance.recordSample(ucontext);
}
```

Теперь всё работает, ура

Но, кажется, чего-то не хватает...

наша функция:

```
void ourHandler();
```

POSIX обработчик:

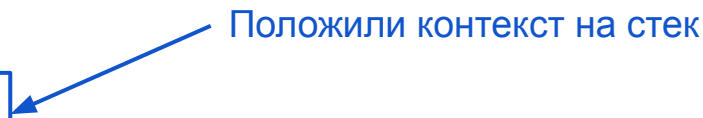
```
void signalHandler(int signo, siginfo_t* siginfo, void* ucontext) {
   Profiler::_instance.recordSample(ucontext);
}
```

Будем передавать контекст в наш обработчик

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);
   CONTEXT *ctxtCopy = saveContext(&ctxt);

   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = ctxt.Rip;
   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = (DWORD64) ctxtCopy;
   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Будем передавать контекст в наш обработчик

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);
   CONTEXT *ctxtCopy = saveContext(&ctxt);

   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = ctxt.Rip;
   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = (DWORD64) ctxtCopy;
   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```



Будем передавать контекст в наш обработчик [2]

```
void ourHandler(CONTEXT* ctxt) {
   // TODO: call profiler
}

void callOtherFunction(HANDLE myPoorLittleThread) {
   doInterruptThread(myPoorLittleThread, ourHandler);
}

int main() {
   HANDLE thread = threadHandle(GetCurrentThreadId());
   _beginthread(callOtherFunction, 0, thread);
   printf("main() is running on thread: %ld\n", GetCurrentThreadId());
   ull result = doWork();
   printf("result: %llu\n", result);
   return 0;
}
```

Вопрос в чат: чем плохо printf до ResumeThread?

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);
   // что плохого тут может случиться?
   printf("Rip: 0x%llx, Rsp: 0x%llx\n", ctxt.Rip, ctxt.Rsp);
   CONTEXT *ctxtCopy = saveContext(&ctxt);

   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = ctxt.Rip;
   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = (DWORD64) ctxtCopy;
   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Вопрос в чат: чем плохо printf до ResumeThread?

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);
   CONTEXT ctxt;
   ctxt.ContextFlags = CONTEXT_FULL;
   GetThreadContext(thread, &ctxt);
   // что плохого тут может случиться?
   printf("Rip: 0x%llx, Rsp: 0x%llx\n", ctxt.Rip, ctxt.Rsp);
   CONTEXT *ctxtCopy = saveContext(&ctxt);

   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = ctxt.Rip;
   ctxt.Rsp = ctxt.Rsp - sizeof(DWORD64);
   *(DWORD64*)ctxt.Rsp = (DWORD64) ctxtCopy;
   ctxt.Rip = (DWORD64) handler;

   SetThreadContext(thread, &ctxt);
   ResumeThread(thread);
}
```

Варианты ответа:

1. Всё будет хорошо
2. Программа упадёт на этой строчке
3. Программа никогда не завершится
4. Undefined behavior

Теперь всё точно работает, ура

наша функция:

```
void ourHandler(CONTEXT* ctxt) {
   Profiler::_instance.recordSample(ctxt);
}
```

posix обработчик:

```
void signalHandler(int signo, siginfo_t* siginfo, void* ucontext) {
   Profiler::_instance.recordSample(ucontext);
}
```

Давайте проверим это на jdk

Давайте проверим это на jdk

```
# A fatal error has been detected by the Java Runtime Environment:
#
#  Internal Error (c:\buildagent\work\src\hotspot\cpu\x86\bytes_x86.hpp:43), pid=7140, tid=12276
#  assert(p != 0LL) failed: null pointer
```

Давайте проверим это на jdk

```
# A fatal error has been detected by the Java Runtime Environment:
#
#  Internal Error (c:\buildagent\work\src\hotspot\cpu\x86\bytes_x86.hpp:43), pid=7140, tid=12276
#  assert(p != 0LL) failed: null pointer
```

```
# A fatal error has been detected by the Java Runtime Environment:
#
#  EXCEPTION_ACCESS_VIOLATION (0xc0000005) at pc=0x00007fff58df7375, pid=18984, tid=18504
```

Давайте проверим это на jdk

```
# A fatal error has been detected by the Java Runtime Environment:
#
#  Internal Error (c:\buildagent\work\src\hotspot\cpu\x86\bytes_x86.hpp:43), pid=7140, tid=12276
#  assert(p != 0LL) failed: null pointer
```

```
# A fatal error has been detected by the Java Runtime Environment:
#
#  EXCEPTION_ACCESS_VIOLATION (0xc0000005) at pc=0x00007fff58df7375, pid=18984, tid=18504
```

```
# A fatal error has been detected by the Java Runtime Environment:
#
#  Internal Error (c:/BuildAgent/work/src/hotspot/cpu/x86/macroAssembler_x86.cpp:903), pid=20540, tid=15636
#  assert(false) failed: DEBUG MESSAGE: i2c adapter must return to an interpreter frame
```

Об использовании правильных инструментов

src/hotspot/cpu/x86/stubGenerator_x86_64.cpp:

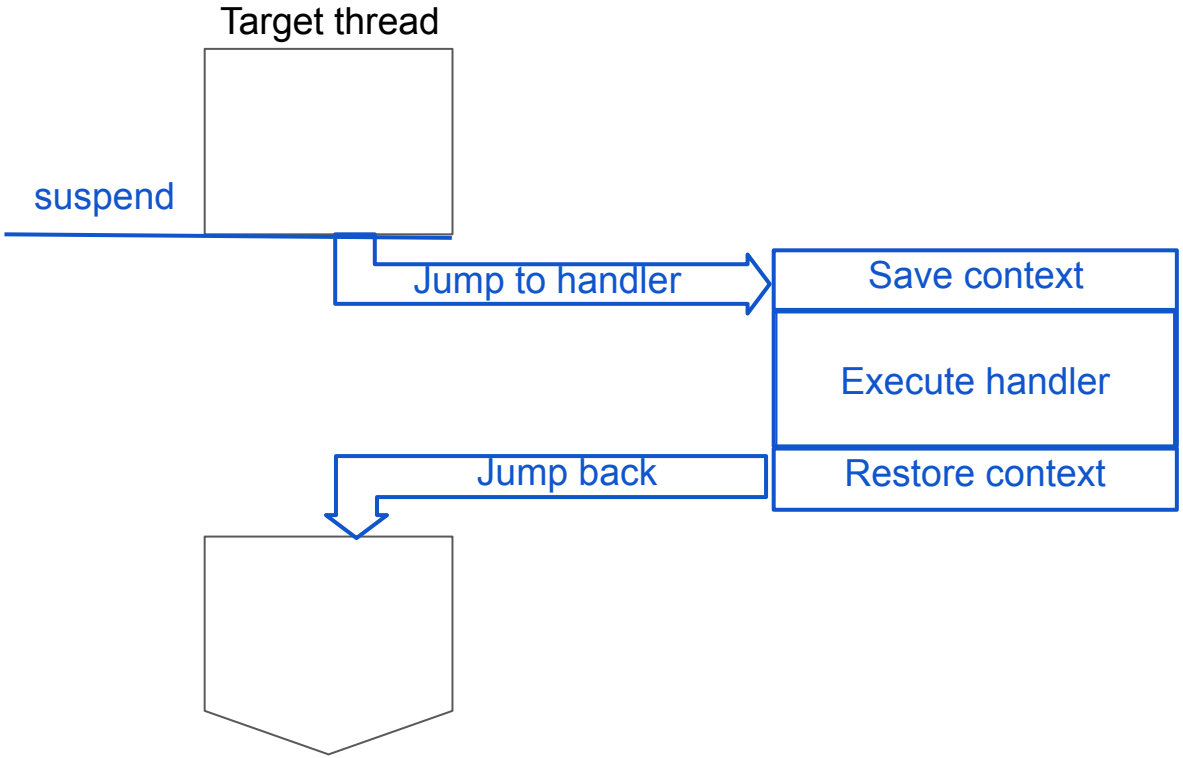
```
address generate_my_forte_stub() {
   StubCodeMark mark(this, "StubRoutines", "my_trampoline_stub");
   address start = __ pc();
   int total_frame_words;
   RegisterSaver::save_live_registers(_masm, &total_frame_words, true);
   __ movptr(rcx, Address(rsp, wordSize * (total_frame_words + 1))); //ctxt pointer
   __ mov64(rax, 0x0101010101010101); // target function
   __ mov(r12, rsp); // remember sp
   __ andptr(rsp, -16);
   __ call(rax);
   __ mov(rsp, r12); // restore sp
   RegisterSaver::restore_live_registers(_masm, true);
   __ xchgptr(rsp, Address(rsp, wordSize * 1)); // swap with original rsp
   // jmp qword [rip+0] (0xff 0x25 0x0 0x0 0x0 0x0)
   __ emit_int16((unsigned char)0xff, (unsigned char)0x25);
   __ emit_int32((unsigned char)0x0, (unsigned char)0x0, (unsigned char)0x0, (unsigned char)0x0);
   __ emit_int64(0xcccccccccccccccc); // return address
   __ nop();
   return start;
}
```

Об использовании правильных инструментов

```
0x2c04d191294: 55                                  push   rbp
0x2c04d191295: 48 8b ec                            mov    rbp, rsp
0x2c04d191298: 9c                                  pushfq
0x2c04d191299: 48 83 ec 08                         sub    rsp, 0x8
0x2c04d19129d: 48 89 64 24 d8                      mov    qword ptr [rsp - 0x28], rsp
0x2c04d1912a2: 48 81 ec 80 00 00 00                sub    rsp, 0x80
0x2c04d1912a9: 48 89 44 24 78                      mov    qword ptr [rsp + 0x78], rax
0x2c04d1912ae: 48 89 4c 24 70                      mov    qword ptr [rsp + 0x70], rcx
0x2c04d1912b3: 48 89 54 24 68                      mov    qword ptr [rsp + 0x68], rdx
0x2c04d1912b8: 48 89 5c 24 60                      mov    qword ptr [rsp + 0x60], rbx
0x2c04d1912bd: 48 89 6c 24 50                      mov    qword ptr [rsp + 0x50], rbp
0x2c04d1912c2: 48 89 74 24 48                      mov    qword ptr [rsp + 0x48], rsi
0x2c04d1912c7: 48 89 7c 24 40                      mov    qword ptr [rsp + 0x40], rdi
0x2c04d1912cc: 4c 89 44 24 38                      mov    qword ptr [rsp + 0x38], r8
0x2c04d1912d1: 4c 89 4c 24 30                      mov    qword ptr [rsp + 0x30], r9
0x2c04d1912d6: 4c 89 54 24 28                      mov    qword ptr [rsp + 0x28], r10
0x2c04d1912db: 4c 89 5c 24 20                      mov    qword ptr [rsp + 0x20], r11
0x2c04d1912e0: 4c 89 64 24 18                      mov    qword ptr [rsp + 0x18], r12
0x2c04d1912e5: 4c 89 6c 24 10                      mov    qword ptr [rsp + 0x10], r13
0x2c04d1912ea: 4c 89 74 24 08                      mov    qword ptr [rsp + 0x8], r14
0x2c04d1912ef: 4c 89 3c 24                         mov    qword ptr [rsp], r15
0x2c04d1912f3: 48 81 ec 80 0a 00 00                sub    rsp, 0xa80
0x2c04d1912fa: 48 0f ae 04 24                      fxsave64 [rsp]
0x2c04d1912ff: c4 e3 7d 19 84 24 40 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x240], ymm0, 0x1
0x2c04d19130a: c4 e3 7d 19 8c 24 50 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x250], ymm1, 0x1
0x2c04d191315: c4 e3 7d 19 94 24 60 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x260], ymm2, 0x1
0x2c04d191320: c4 e3 7d 19 9c 24 70 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x270], ymm3, 0x1
0x2c04d19132b: c4 e3 7d 19 a4 24 80 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x280], ymm4, 0x1
0x2c04d191336: c4 e3 7d 19 ac 24 90 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x290], ymm5, 0x1
0x2c04d191341: c4 e3 7d 19 b4 24 a0 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x2a0], ymm6, 0x1
0x2c04d19134c: c4 e3 7d 19 bc 24 b0 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x2b0], ymm7, 0x1
0x2c04d191357: c4 63 7d 19 84 24 c0 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x2c0], ymm8, 0x1
0x2c04d191362: c4 63 7d 19 8c 24 d0 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x2d0], ymm9, 0x1
0x2c04d19136d: c4 63 7d 19 94 24 e0 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x2e0], ymm10, 0x1
0x2c04d191378: c4 63 7d 19 9c 24 f0 02 00 00 01    vextractf128 xmmword ptr [rsp + 0x2f0], ymm11, 0x1
0x2c04d191383: c4 63 7d 19 a4 24 00 03 00 00 01    vextractf128 xmmword ptr [rsp + 0x300], ymm12, 0x1
0x2c04d19138e: c4 63 7d 19 ac 24 10 03 00 00 01    vextractf128 xmmword ptr [rsp + 0x310], ymm13, 0x1
0x2c04d191399: c4 63 7d 19 b4 24 20 03 00 00 01    vextractf128 xmmword ptr [rsp + 0x320], ymm14, 0x1
0x2c04d1913a4: c4 63 7d 19 bc 24 30 03 00 00 01    vextractf128 xmmword ptr [rsp + 0x330], ymm15, 0x1
0x2c04d1913af: c5 f8 77                            vzeroupper
0x2c04d1913b2: 48 83 ec 20                         sub    rsp, 0x20
0x2c04d1913b6: 48 8b 8c 24 48 0b 00 00             mov    rcx, qword ptr [rsp + 0xb48]
0x2c04d1913be: 48 b8 01 01 01 01 01 01 01 01       movabs rax, 0x101010101010101
0x2c04d1913c8: 4c 8b e4                            mov    r12, rsp
0x2c04d1913cb: 48 83 e4 f0                         and    rsp, -0x10
0x2c04d1913cf: ff d0                               call   rax
```

```
0x2c04d1913d1: 49 8b e4                            mov    rsp, r12
0x2c04d1913d4: 48 83 c4 20                         add    rsp, 0x20
0x2c04d1913d8: c5 f8 77                            vzeroupper
0x2c04d1913db: c4 e3 7d 18 84 24 40 02 00 00 01    vinsertf128 ymm0, ymm0, xmmword ptr [rsp + 0x240], 0x1
0x2c04d1913e6: c4 e3 75 18 8c 24 50 02 00 00 01    vinsertf128 ymm1, ymm1, xmmword ptr [rsp + 0x250], 0x1
0x2c04d1913f1: c4 e3 6d 18 94 24 60 02 00 00 01    vinsertf128 ymm2, ymm2, xmmword ptr [rsp + 0x260], 0x1
0x2c04d1913fc: c4 e3 65 18 9c 24 70 02 00 00 01    vinsertf128 ymm3, ymm3, xmmword ptr [rsp + 0x270], 0x1
0x2c04d191407: c4 e3 5d 18 a4 24 80 02 00 00 01    vinsertf128 ymm4, ymm4, xmmword ptr [rsp + 0x280], 0x1
0x2c04d191412: c4 e3 55 18 ac 24 90 02 00 00 01    vinsertf128 ymm5, ymm5, xmmword ptr [rsp + 0x290], 0x1
0x2c04d19141d: c4 e3 4d 18 b4 24 a0 02 00 00 01    vinsertf128 ymm6, ymm6, xmmword ptr [rsp + 0x2a0], 0x1
0x2c04d191428: c4 e3 45 18 bc 24 b0 02 00 00 01    vinsertf128 ymm7, ymm7, xmmword ptr [rsp + 0x2b0], 0x1
0x2c04d191433: c4 63 3d 18 84 24 c0 02 00 00 01    vinsertf128 ymm8, ymm8, xmmword ptr [rsp + 0x2c0], 0x1
0x2c04d19143e: c4 63 35 18 8c 24 d0 02 00 00 01    vinsertf128 ymm9, ymm9, xmmword ptr [rsp + 0x2d0], 0x1
0x2c04d191449: c4 63 2d 18 94 24 e0 02 00 00 01    vinsertf128 ymm10, ymm10, xmmword ptr [rsp + 0x2e0], 0x1
0x2c04d191454: c4 63 25 18 9c 24 f0 02 00 00 01    vinsertf128 ymm11, ymm11, xmmword ptr [rsp + 0x2f0], 0x1
0x2c04d19145f: c4 63 1d 18 a4 24 00 03 00 00 01    vinsertf128 ymm12, ymm12, xmmword ptr [rsp + 0x300], 0x1
0x2c04d19146a: c4 63 15 18 ac 24 10 03 00 00 01    vinsertf128 ymm13, ymm13, xmmword ptr [rsp + 0x310], 0x1
0x2c04d191475: c4 63 0d 18 b4 24 20 03 00 00 01    vinsertf128 ymm14, ymm14, xmmword ptr [rsp + 0x320], 0x1
0x2c04d191480: c4 63 05 18 bc 24 30 03 00 00 01    vinsertf128 ymm15, ymm15, xmmword ptr [rsp + 0x330], 0x1
0x2c04d19148b: 48 0f ae 0c 24                      fxrstor64 [rsp]
0x2c04d191490: 48 81 c4 80 0a 00 00                add    rsp, 0xa80
0x2c04d191497: 4c 8b 3c 24                         mov    r15, qword ptr [rsp]
0x2c04d19149b: 4c 8b 74 24 08                      mov    r14, qword ptr [rsp + 0x8]
0x2c04d1914a0: 4c 8b 6c 24 10                      mov    r13, qword ptr [rsp + 0x10]
0x2c04d1914a5: 4c 8b 64 24 18                      mov    r12, qword ptr [rsp + 0x18]
0x2c04d1914aa: 4c 8b 5c 24 20                      mov    r11, qword ptr [rsp + 0x20]
0x2c04d1914af: 4c 8b 54 24 28                      mov    r10, qword ptr [rsp + 0x28]
0x2c04d1914b4: 4c 8b 4c 24 30                      mov    r9, qword ptr [rsp + 0x30]
0x2c04d1914b9: 4c 8b 44 24 38                      mov    r8, qword ptr [rsp + 0x38]
0x2c04d1914be: 48 8b 7c 24 40                      mov    rdi, qword ptr [rsp + 0x40]
0x2c04d1914c3: 48 8b 74 24 48                      mov    rsi, qword ptr [rsp + 0x48]
0x2c04d1914c8: 48 8b 6c 24 50                      mov    rbp, qword ptr [rsp + 0x50]
0x2c04d1914cd: 48 8b 5c 24 60                      mov    rbx, qword ptr [rsp + 0x60]
0x2c04d1914d2: 48 8b 54 24 68                      mov    rdx, qword ptr [rsp + 0x68]
0x2c04d1914d7: 48 8b 4c 24 70                      mov    rcx, qword ptr [rsp + 0x70]
0x2c04d1914dc: 48 8b 44 24 78                      mov    rax, qword ptr [rsp + 0x78]
0x2c04d1914e1: 48 81 c4 80 00 00 00                add    rsp, 0x80
0x2c04d1914e8: 48 83 c4 08                         add    rsp, 0x8
0x2c04d1914ec: 9d                                  popfq
0x2c04d1914ed: 5d                                  pop    rbp
0x2c04d1914ee: 48 87 64 24 08                      xchg   qword ptr [rsp + 0x8], rsp
0x2c04d1914f3: ff 25 00 00 00 00                   jmp    qword ptr [rip]
0x2c04d1914f9: cc                                  int3
0x2c04d1914fa: cc                                  int3
0x2c04d1914fb: cc                                  int3
0x2c04d1914fc: cc                                  int3
0x2c04d1914fd: cc                                  int3
0x2c04d1914fe: cc                                  int3
0x2c04d1914ff: cc                                  int3
0x2c04d191500: cc                                  int3
0x2c04d191501: 90                                  nop
```

Мы научились эмулировать POSIX сигналы



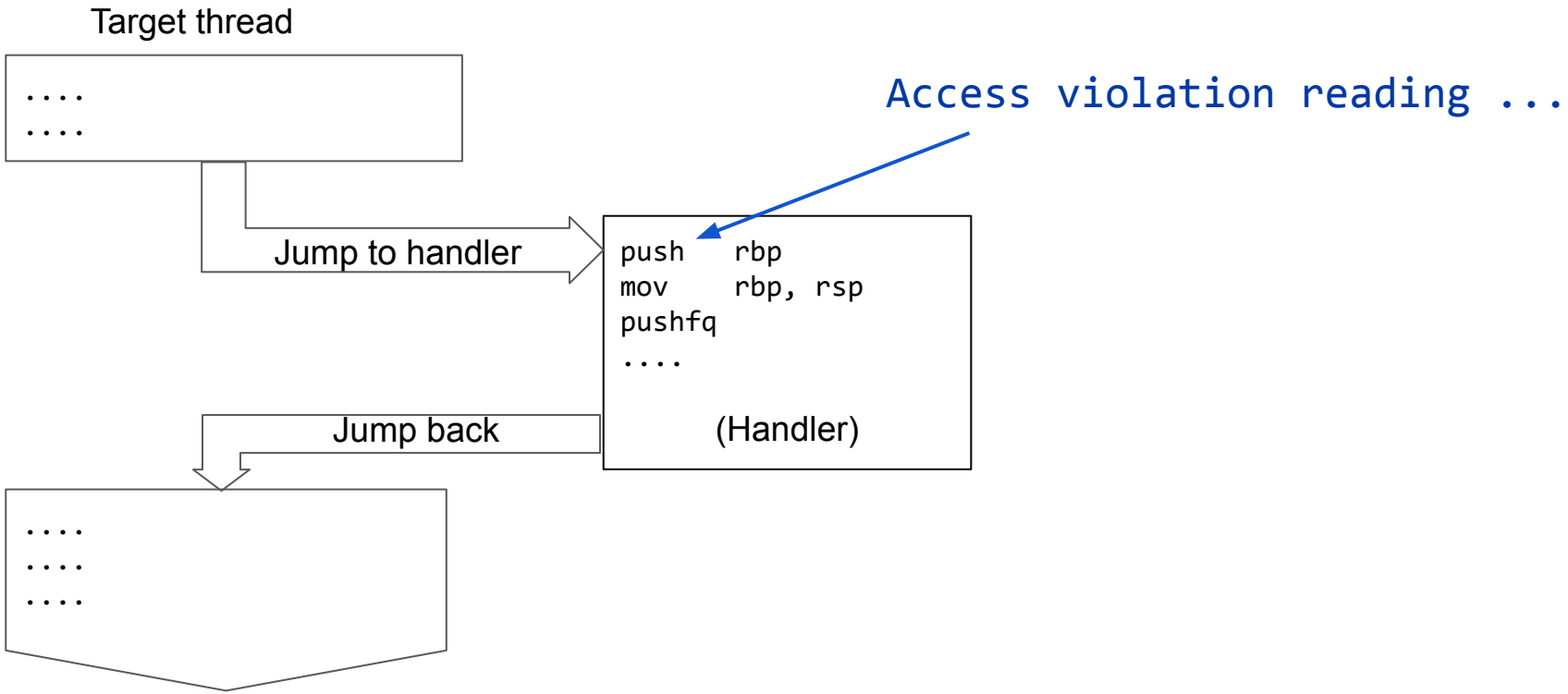
```
> java -agentpath:agent-final-final-v2.dll

@ideaUltimateArgs com.intellij.idea.Main
```

Но нам это не помогло...

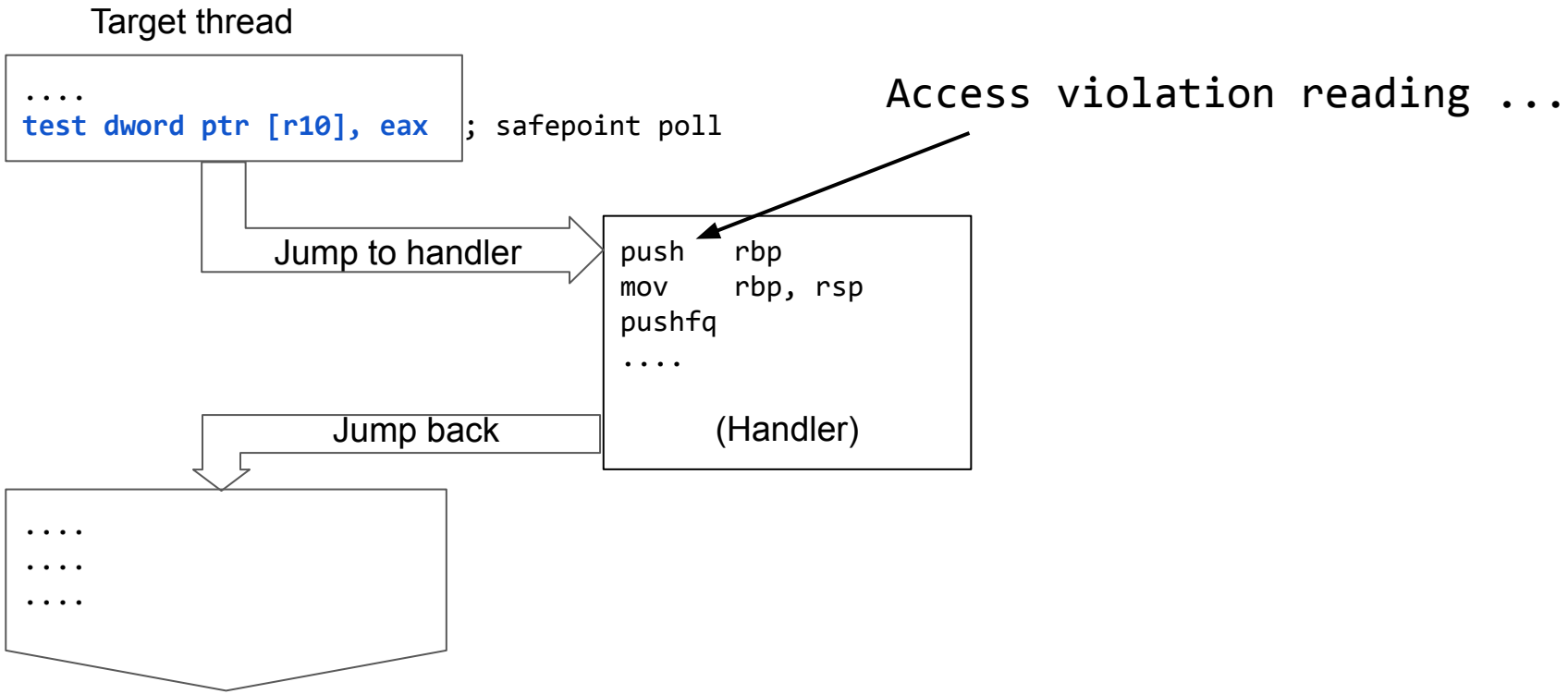
```
Exception 0xc0000005 encountered at address 0x2dd39fa0000: Access violation
reading location 0x2dd09130008
```

🤔🤔🤔

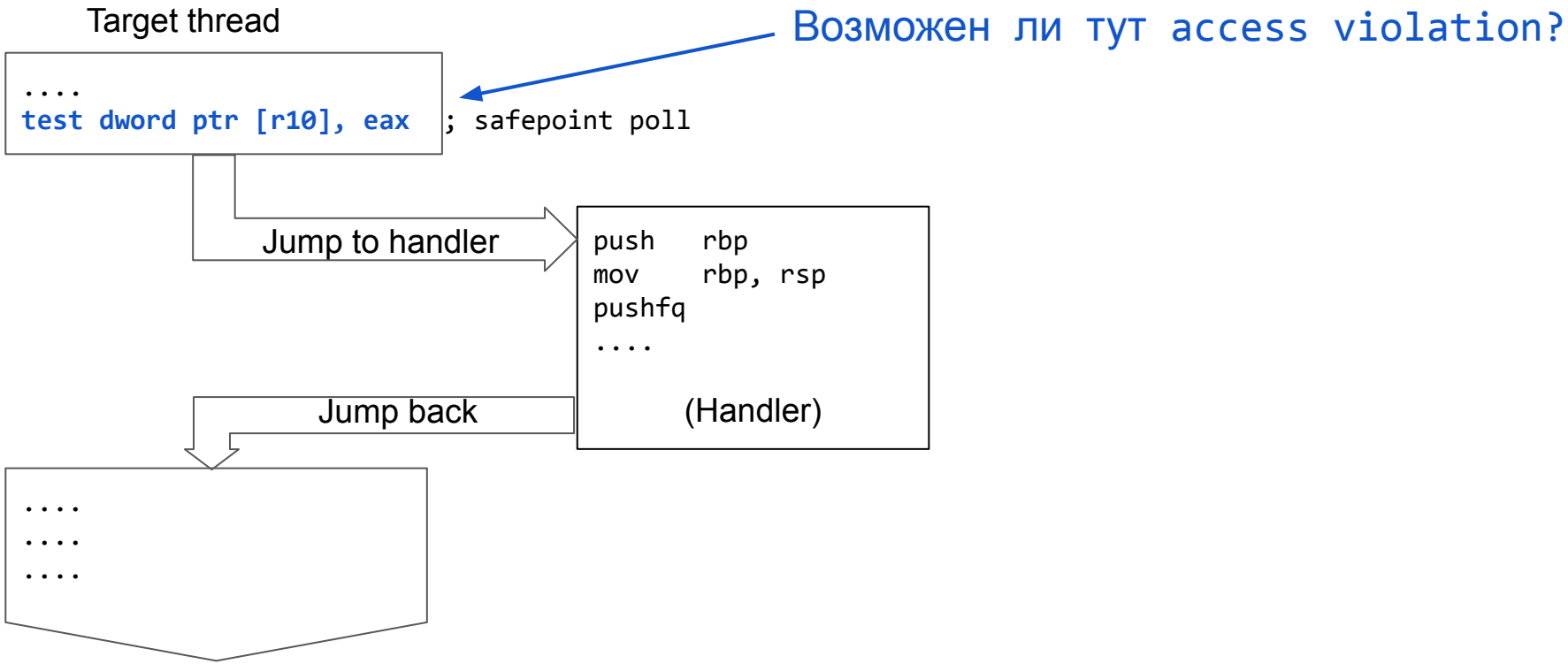
Но нам это не помогло...



Но нам это не помогло...



Но нам это не помогло...



Но нам это не помогло...

```
> java -Xlog:os ...
```

JEP 158: Unified JVM Logging: https://openjdk.java.net/jeps/158

Но нам это не помогло...

```
Exception 0xc0000005 encountered at address 0x2dd39fa0000: Access violation
reading location 0x2dd09130008
```

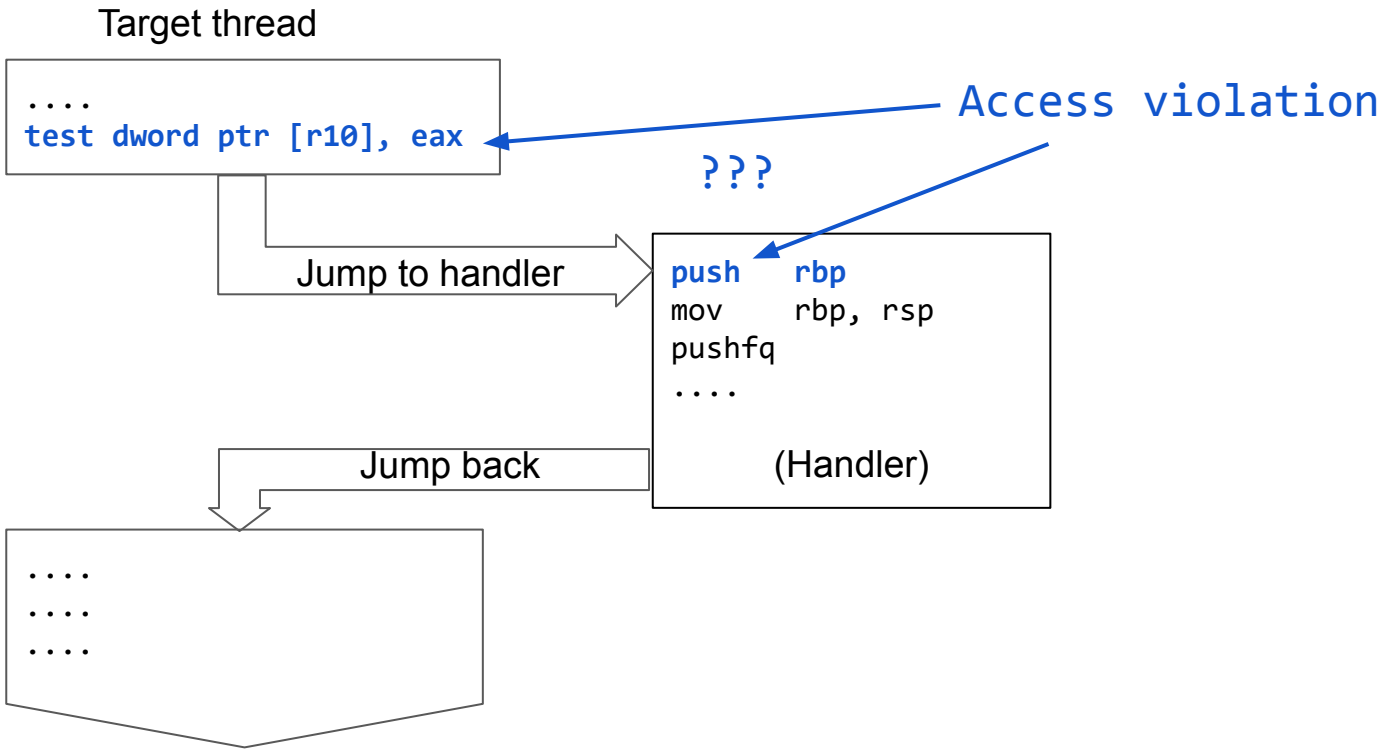
```
> java -Xlog:os ...

....
[0.058s][info][os] SafePoint Polling address, bad (protected)
page:0x000002dd09130000, good (unprotected) page:0x000002dd09131000
....
```

Но нам это не помогло...

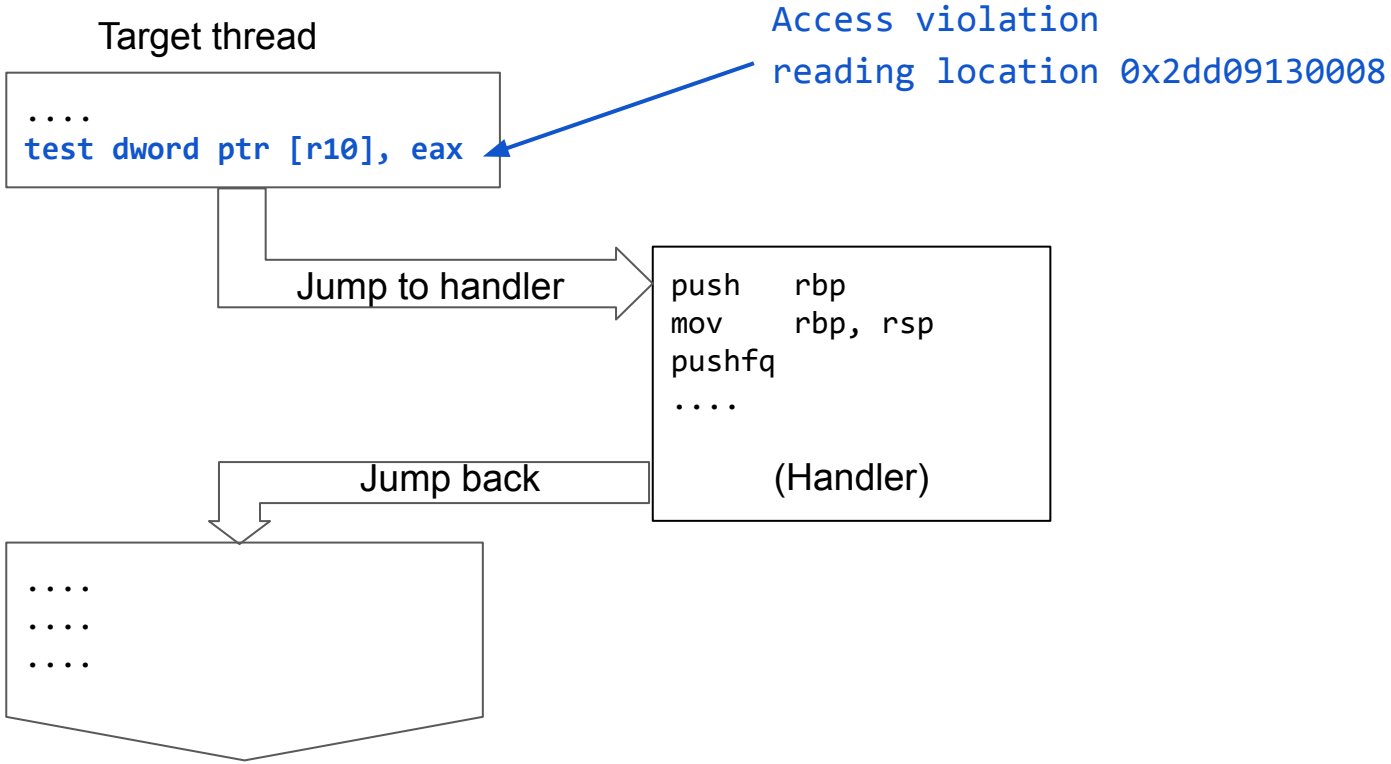
```
Exception 0xc0000005 encountered at address 0x2dd39fa0000: Access violation
reading location 0x2dd09130008
```

```
> java -Xlog:os ...

....
[0.058s][info][os] SafePoint Polling address, bad (protected)
page:0x000002dd09130000, good (unprotected) page:0x000002dd09131000
....
```

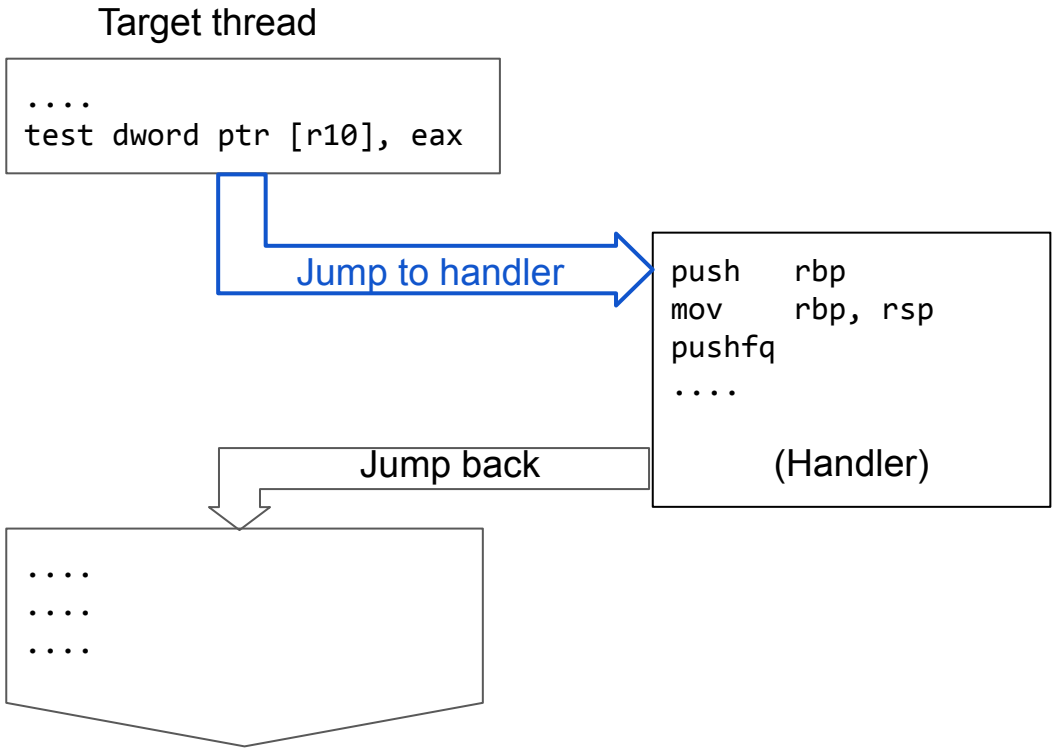
Но нам это не помогло...



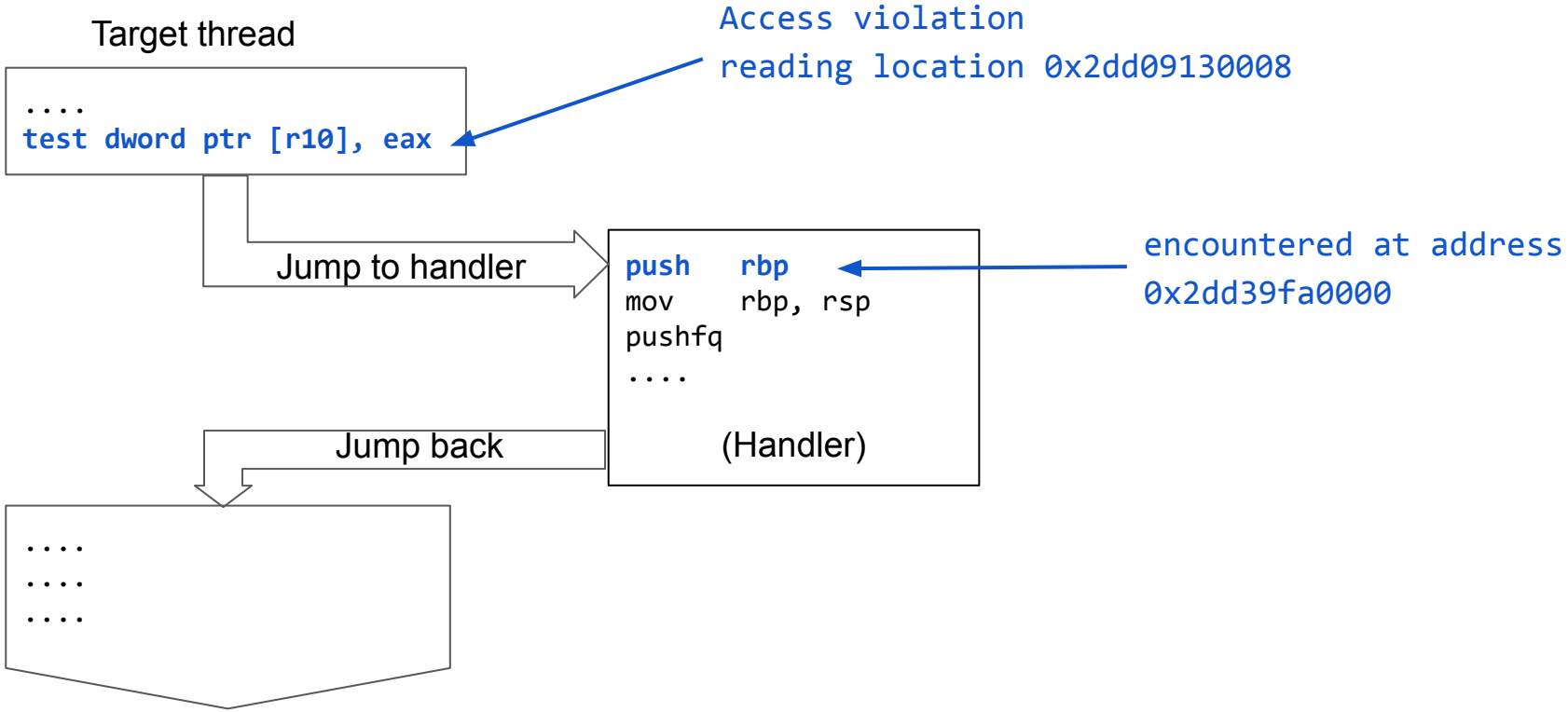
А что, собственно, случилось?



А что, собственно, случилось?



А что, собственно, случилось?




Ну нахер
Пойду на Kotlin писать

А зачем вызывать AGCT на том же треде?

А зачем вызывать AGCT на том же треде?

По коду ровным слоем разложены ассерты:

```
assert(JavaThread::current() == thread,
       "AsyncGetCallTrace must be called by the current interrupted thread");
```

А зачем вызывать AGCT на том же треде?

По коду ровным слоем разложены ассерты:


```
assert(JavaThread::current() == thread,
       "AsyncGetCallTrace must be called by the current interrupted thread");
```

Пришло время плана Б

Вспомним, как работает JFR (и почему он без safepoint bias):

1. Останавливает целевой поток средствами ОС
2. Пытается получить java часть стека остановленного потока
3. Пробуждает целевой поток

Пришло время плана Б [2]

Нам нужно сделать так, чтобы не срабатывал этот ассерт:

```
assert(JavaThread::current() == thread,
       "AsyncGetCallTrace must be called by the current interrupted thread");
```

Тогда мы сможем делать так:

1. Останавливает целевой поток средствами ОС
2. **Получаем полный (java+native) стек остановленного потока**
3. Пробуждает целевой поток

Как можно выключить ассерт

Вариантов, на самом деле, много:

Как можно выключить ассерт

Вариантов, на самом деле, много:

- Отправить патч в OpenJDK

Как можно выключить ассерт

Вариантов, на самом деле, много:

- ~~Отправить патч в OpenJDK~~ 😱
- Обойти граф вызова функций, начиная от AGCT и в каждой заменить все вызовы этого assert-а nop-ами

Как можно выключить ассерт

Вариантов, на самом деле, много:

- ~~Отправить патч в OpenJDK~~ 😱
- ~~Обойти граф вызова функций, начиная от AGCT и в каждой заменить все вызовы этого assert-а nop-ами~~ 😱
- Замокать все вызовы `JavaThread::current()`

Как можно выключить ассерт

Вариантов, на самом деле, много:

- ~~Отправить патч в OpenJDK~~ 😱
- ~~Обойти граф вызова функций, начиная от ACCT и в каждой заменить все вызовы этого assert-а пор-ами~~ 😱
- ~~Замокать все вызовы `JavaThread::current()`~~ 😱
- Подменить данные, которые возвращает `JavaThread::current()` для конкретного потока

Как можно выключить ассерт

Вариантов, на самом деле, много:

- ~~Отправить патч в OpenJDK~~ 😱
- ~~Обойти граф вызова функций, начиная от ACCT и в каждой заменить все вызовы этого assert-а пор-ами~~ 😱
- ~~Замокать все вызовы `JavaThread::current()`~~ 😱
- Подменить данные, которые возвращает `JavaThread::current()` для конкретного потока 🤔🤔🤔

Две ипостаси JavaThread::current()

```
// Current thread is maintained as a thread-local variable
static __declspec(thread) Thread* _thr_current;

inline Thread* Thread::current_or_null() {
#ifndef USE_LIBRARY_BASED_TLS_ONLY
 return _thr_current;
#else
 return current_or_null_safe();
#endif
}

inline Thread* Thread::current_or_null_safe() {
 if (ThreadLocalStorage::is_initialized()) {
   return ThreadLocalStorage::thread();
 }
 return NULL;
}
```

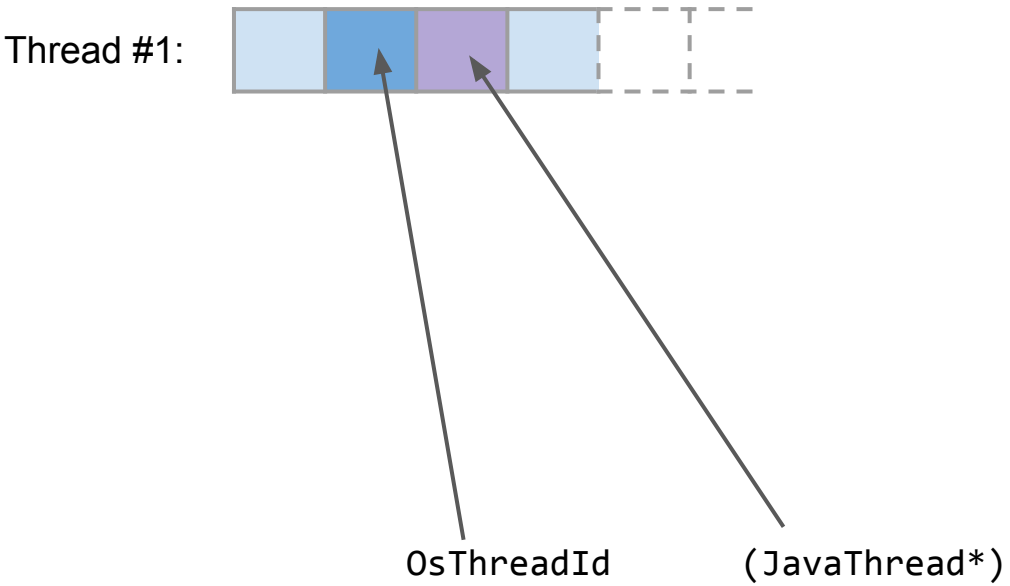
TLS операционной системы

```
static DWORD _thread_key;

void ThreadLocalStorage::init() {
 _thread_key = TlsAlloc();
}

Thread* ThreadLocalStorage::thread() {
 return (Thread*) TlsGetValue(_thread_key);
}

void ThreadLocalStorage::set_thread(Thread* current) {
 TlsSetValue(_thread_key, current);
}
```

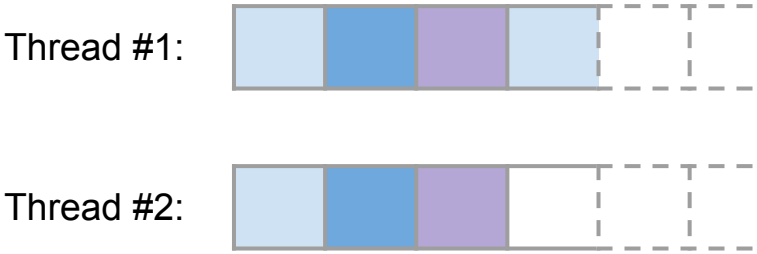
https://docs.microsoft.com/en-us/windows/win32/procthread/thread-local-storage

TLS операционной системы

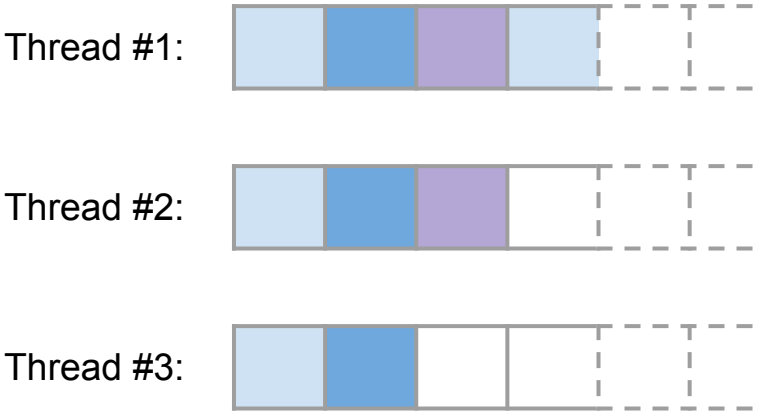
TLS операционной системы



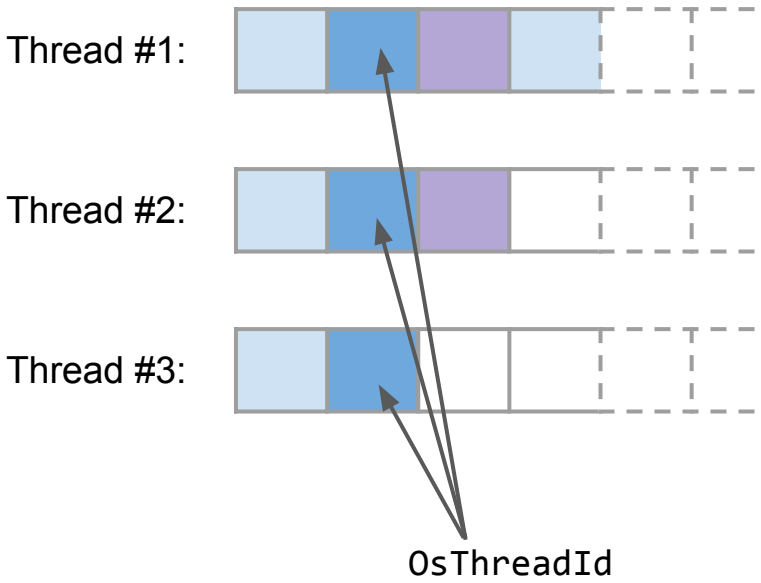
TLS операционной системы



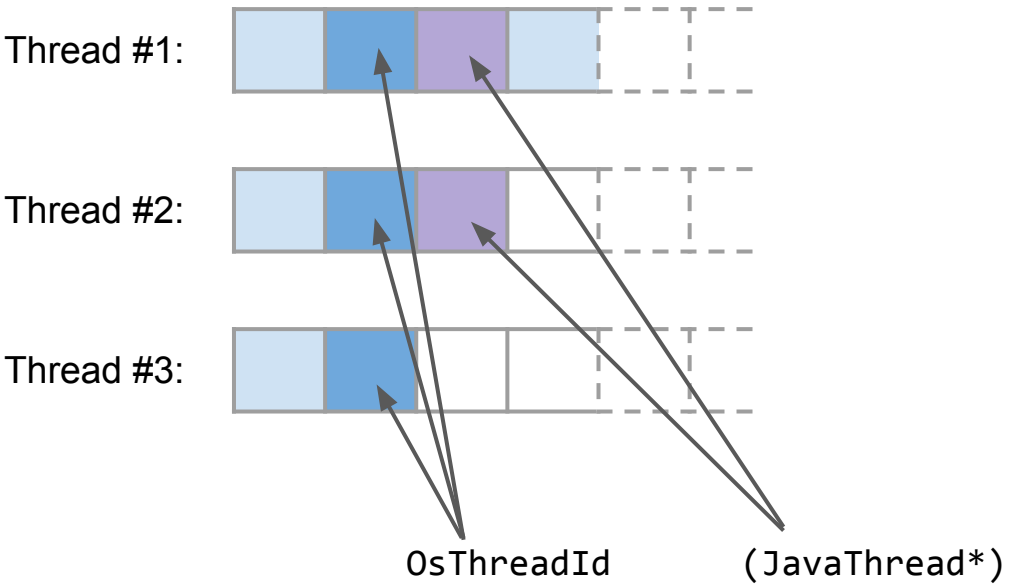
TLS операционной системы



TLS операционной системы



TLS операционной системы




O(N)

Как мы находим эти оффсеты

```
bool initOsTlsOffsets(VMThread* vm_thread) {

  for (DWORD i = 0; i < OS_TLS_MAX_COUNT; i++) {
    if (TlsGetValue(i) == vm_thread) {
      _tls_index = i;
      return true;
    }
  }

  return false;
}
```

С компиляторным thread local всё сложнее

Очевидно, значение хранится где-то в памяти

С компиляторным thread local всё сложнее

Очевидно, значение хранится где-то в памяти

```
Thread::current():
      push   rbx
      sub    rsp, 0x20
      mov    ecx, dword ptr [rip + 0x11f1238]
      mov    rax, qword ptr gs:[0x58]
      mov    ebx, 0x8
      mov    rax, qword ptr [rax + 8*rcx]
      mov    rbx, qword ptr [rbx + rax]
      test   rbx, rbx
      ....
```

С компиляторным thread local всё сложнее

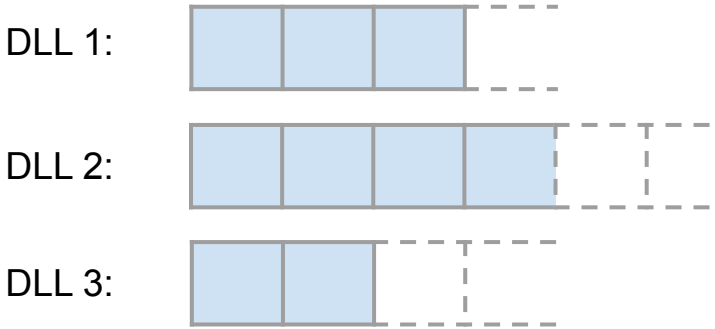
Очевидно, значение хранится где-то в памяти

```
Thread::current():
      push    rbx
      sub     rsp, 0x20
      mov     ecx, dword ptr [rip + 0x11f1238]
      mov     rax, qword ptr gs:[0x58]
      mov     ebx, 0x8
      mov     rax, qword ptr [rax + 8*rcx]
      mov     rbx, qword ptr [rbx + rax]
      test    rbx, rbx
      ....
```

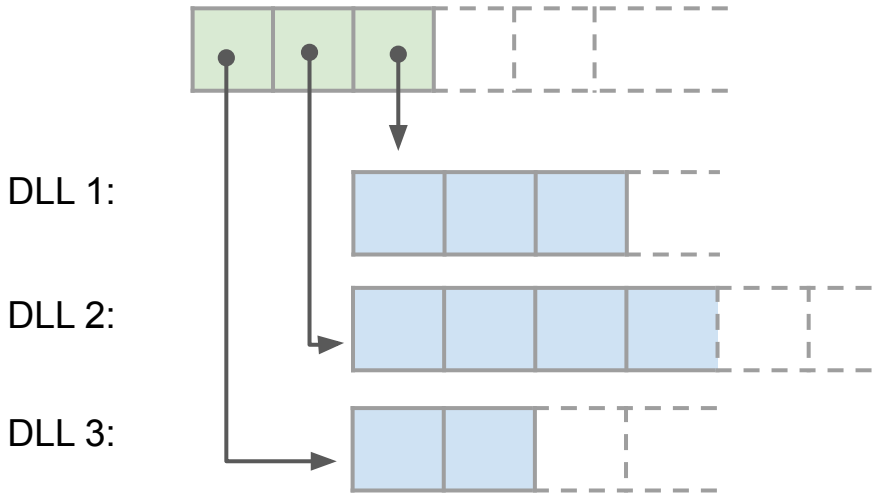
```
(lldb) disas --pc
jvm.dll`Thread::current():
->  0x7fff8623c972 <+34>: test   rbx, rbx
    0x7fff8623c975 <+37>: jne    0x7fff8623c9b0
    0x7fff8623c977 <+39>: call   0x7fff86740780
    0x7fff8623c97c <+44>: test   al, al
(lldb) register read rbx
     rbx = 0x000002bc3869a800
(lldb) frame variable
(JavaThread *) current = 0x000002bc3869a800
```

http://www.nynaeve.net/?p=190

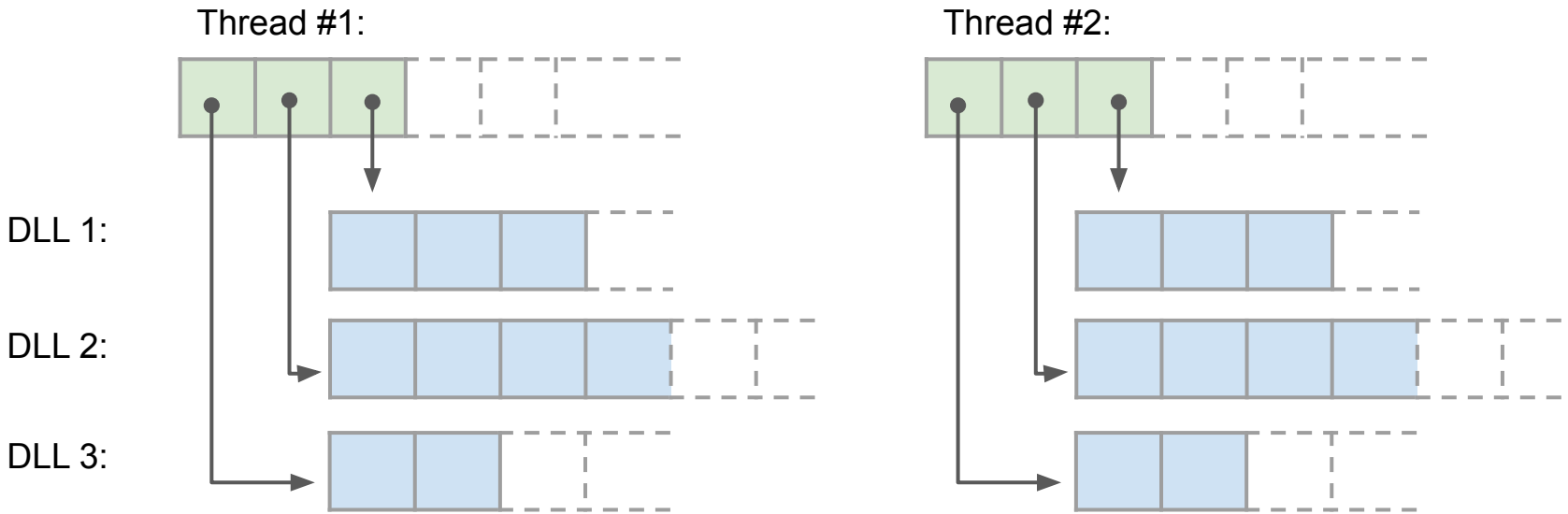
С компиляторным thread local всё сложнее



С компиляторным thread local всё сложнее



С компиляторным thread local всё сложнее



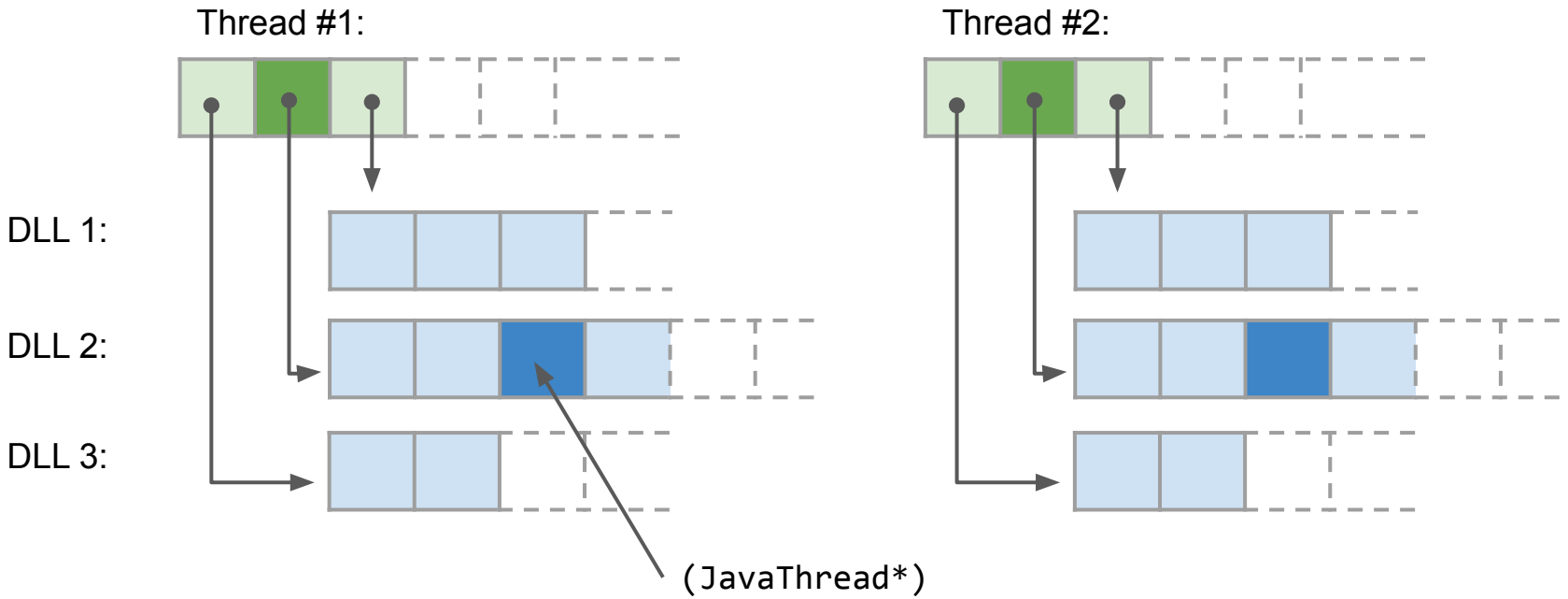
Как мы находим эти оффсеты [2]

```
bool initCompilerTlsOffsets(const void *target_addr) {
 for (int dll_index = 0; dll_index < DLL_COUNT; dll_index++) {

   for (int i = 1; i <= MAX_TLS_VARIABLES_COUNT; i++) {
     auto maybe_target_addr = get_addr_by_offsets(dll_index, i);

     if (maybe_target_addr == target_addr) {
         _compiler_tls_index = dll_index;
         _thread_offset_in_compiler_tls = i;
         return true;
     }
   }
 }
 return false;
}
```

Как мы находим эти оффсеты [2]

```
bool initCompilerTlsOffsets(const void *target_addr) {
 for (int dll_index = 0; dll_index < DLL_COUNT; dll_index++) {

   for (int i = 1; i <= MAX_TLS_VARIABLES_COUNT; i++) {
     auto maybe_target_addr = get_addr_by_offsets(dll_index, i);

     if (maybe_target_addr == target_addr) {
         _compiler_tls_index = dll_index;
         _thread_offset_in_compiler_tls = i;
         return true;
     }
   }
 }
 return false;
}
```

С компиляторным thread local всё сложнее



Как оно теперь работает

```
void get_sample_from_other_thread(CONTEXT *ctxt, HANDLE suspendedThread, DWORD tid) {
  uintptr_t *thread_obj = get_thread_obj_pointer(suspendedThread);

  replace_current_thread_obj_pointer(thread_obj);

  Profiler::_instance.recordSampleForThread(tid, &ctxt, 1);
}
```

Как оно теперь работает

```
void get_sample_from_other_thread(CONTEXT *ctxt, HANDLE suspendedThread, DWORD tid) {
  uintptr_t *thread_obj = get_thread_obj_pointer(suspendedThread);

  replace_current_thread_obj_pointer(thread_obj);

  Profiler::_instance.recordSampleForThread(tid, &ctxt, 1);
}
```

Как оно теперь работает

```
void get_sample_from_other_thread(CONTEXT *ctxt, HANDLE suspendedThread, DWORD tid) {
  uintptr_t *thread_obj = get_thread_obj_pointer(suspendedThread);

  replace_current_thread_obj_pointer(thread_obj);

  Profiler::_instance.recordSampleForThread(tid, &ctxt, 1);
}
```

Как оно теперь работает

```
void get_sample_from_other_thread(CONTEXT *ctxt, HANDLE suspendedThread, DWORD tid) {
  uintptr_t *thread_obj = get_thread_obj_pointer(suspendedThread);

  replace_current_thread_obj_pointer(thread_obj);

  Profiler::_instance.recordSampleForThread(tid, &ctxt, 1);
}
```

Вопрос в чат: чем плохо printf до ResumeThread?

```
void doInterruptThread(HANDLE thread, void* handler) {
   SuspendThread(thread);

   doRecordSampleForThread(thread);
   // что плохого тут может случиться
   printf("Sample Recorded for thread %p", thread);

   ResumeThread(thread);
}
```

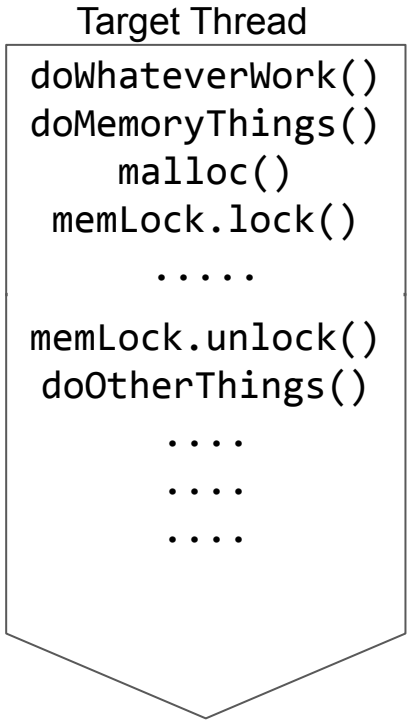
Вопрос в чат: чем плохо printf до ResumeThread?

```
void doInterruptThread(HANDLE thread, void* handler) {
    SuspendThread(thread);

    doRecordSampleForThread(thread);
    // что плохого тут может случиться
    printf("Sample Recorded for thread %p", thread);

    ResumeThread(thread);
}
```

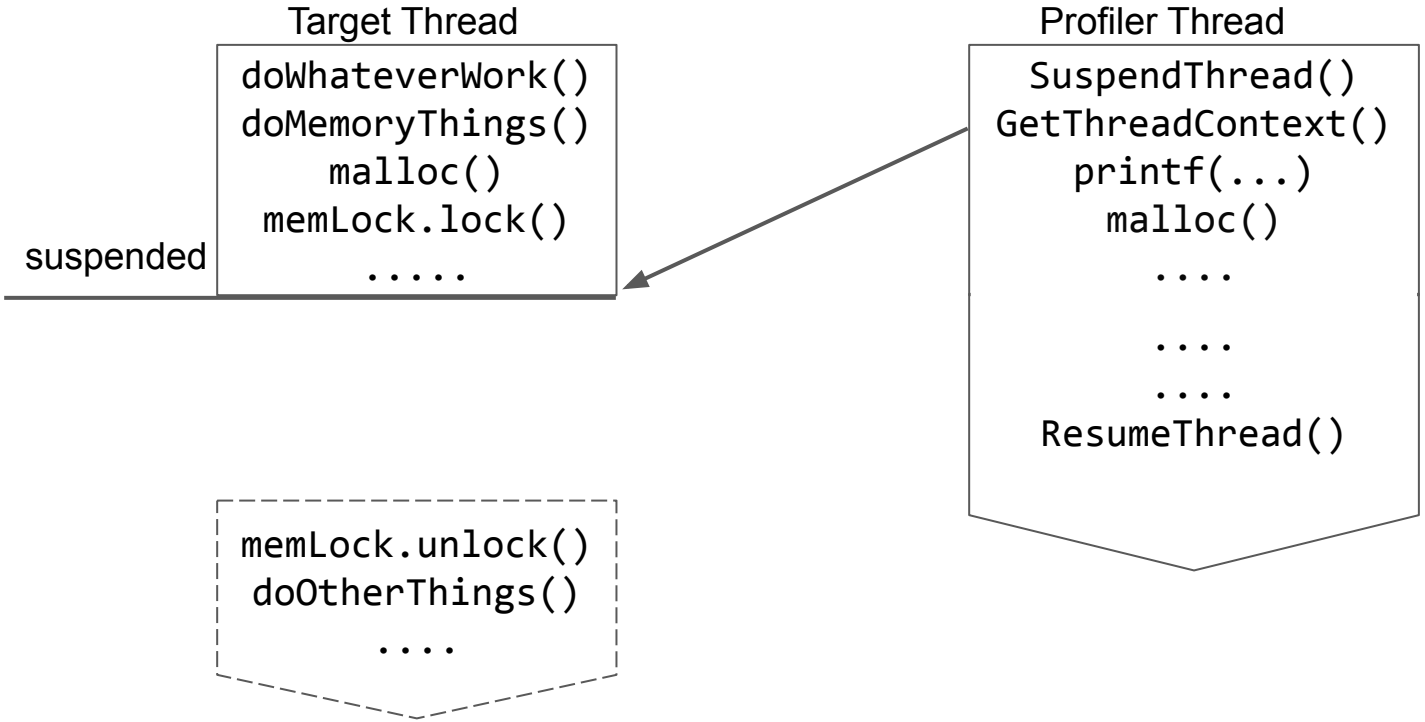
Варианты ответа:

1. Всё будет хорошо
2. Программа упадёт на этой строчке
3. Программа никогда не завершится
4. Undefined behavior

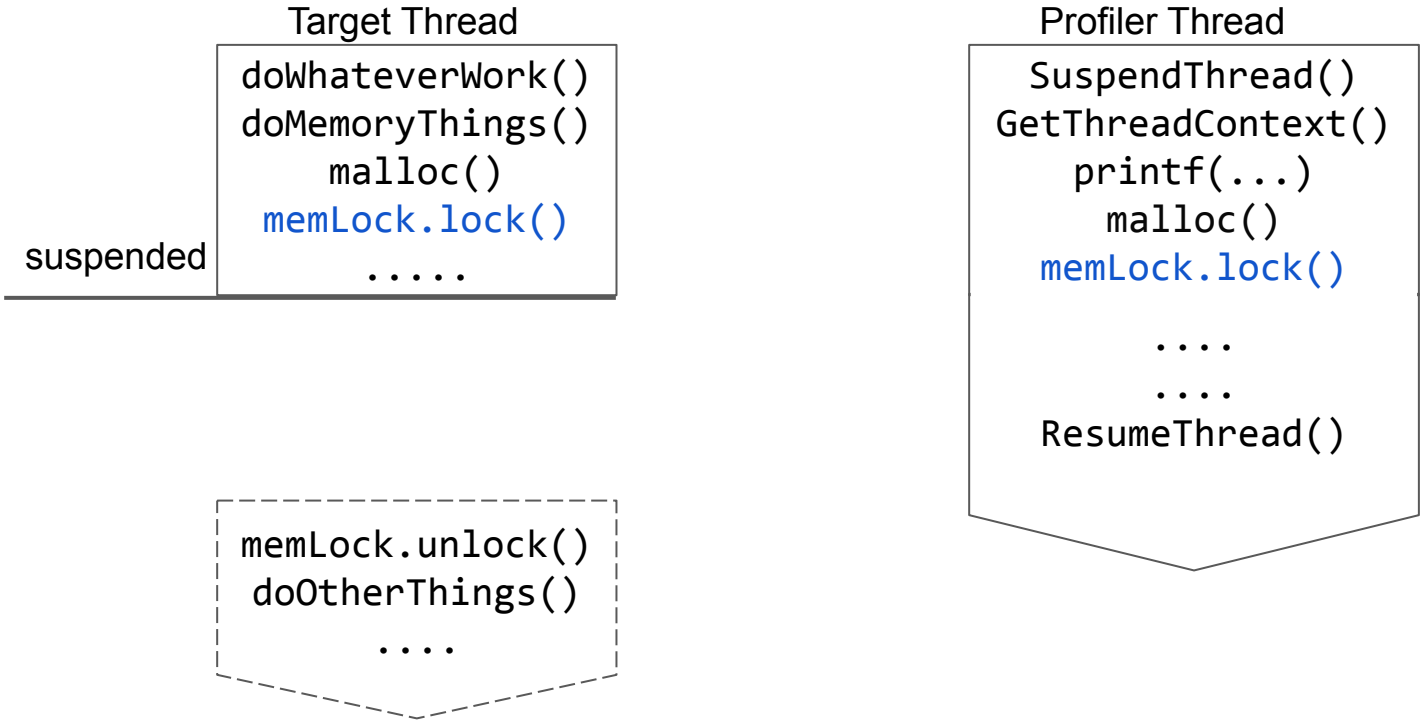
Что же случится?



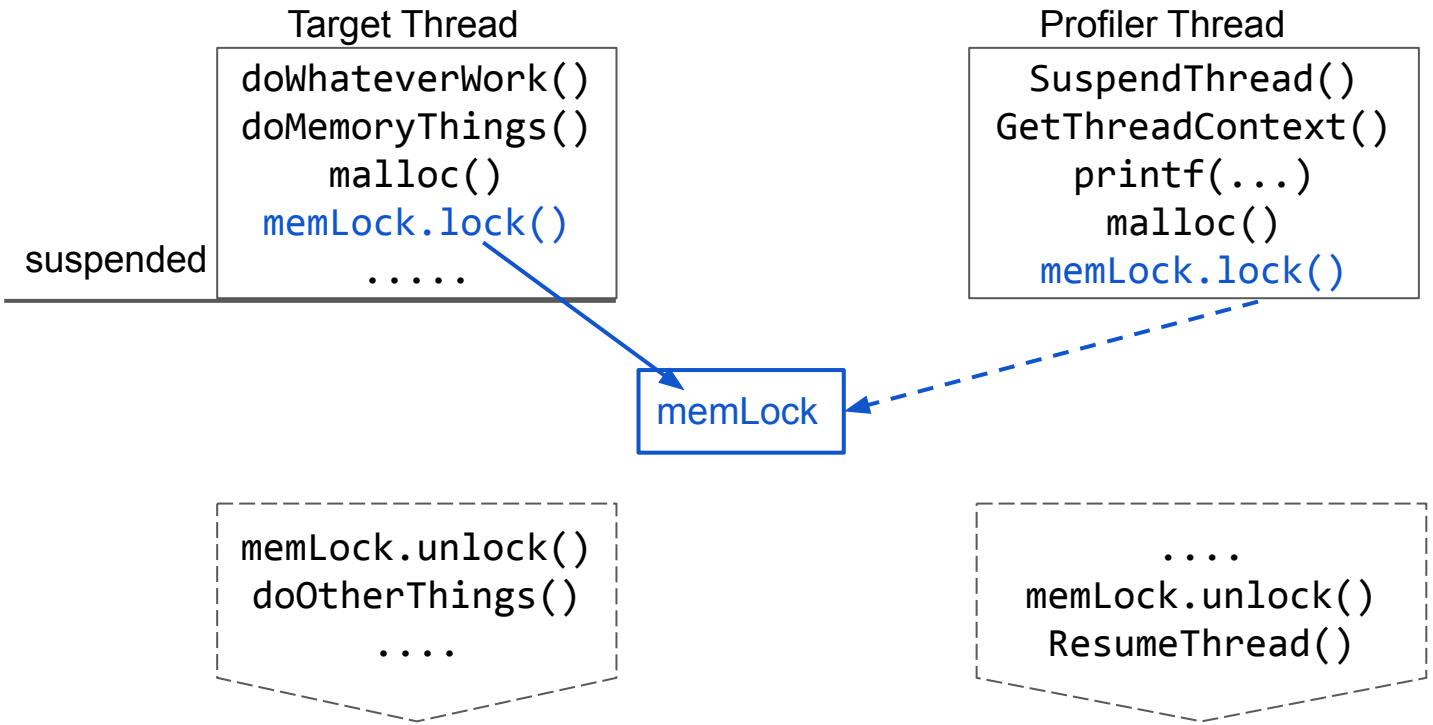
Что же случится?



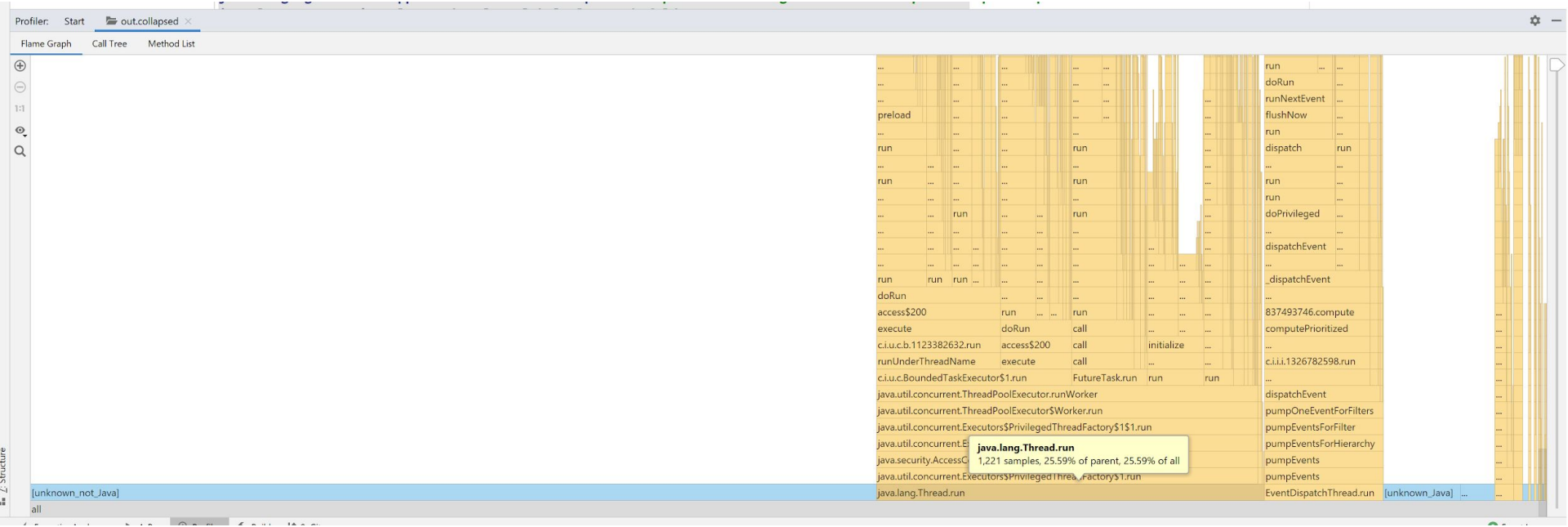
Что же случится?



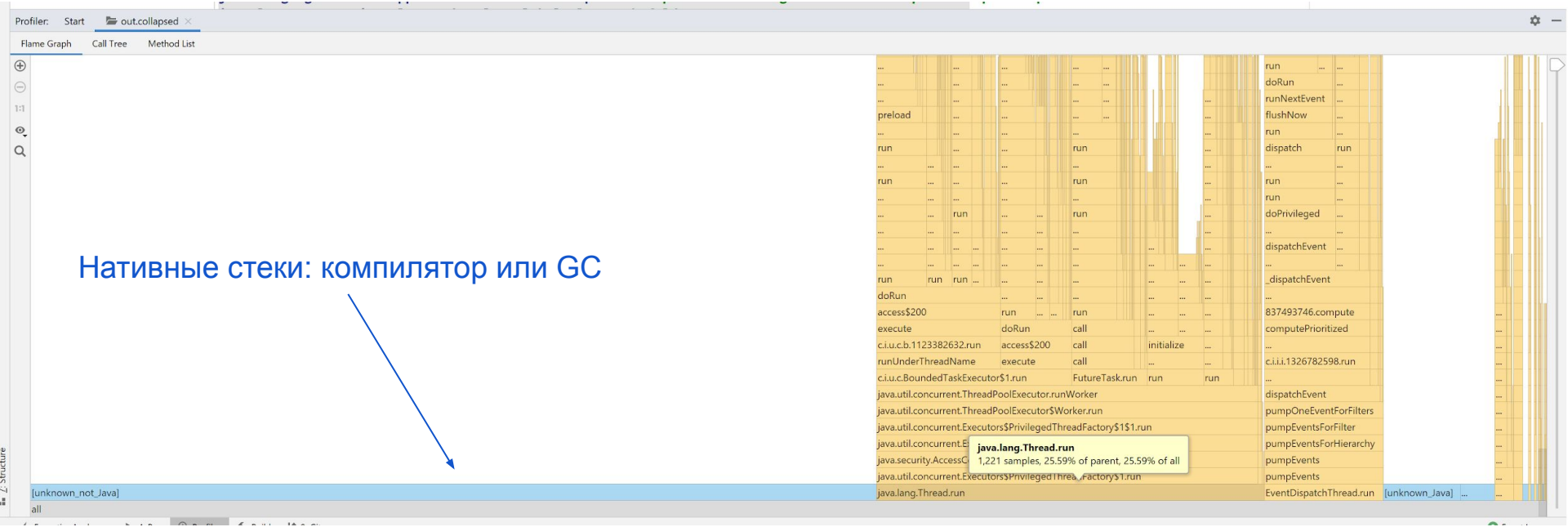
Случится дедлок



Ура, работает



Ура, работает



Windows x64 data-based unwinding

- Загрузка данных из .pdata сегмента exe или dll
- `BOOLEAN RtlAddFunctionTable(FunctionTable, EntryCount, BaseAddress)`
  - `BOOLEAN RtlDeleteFunctionTable(FunctionTable)`
- `BOOLEAN RtlInstallFunctionTableCallback(TableIdentifier, BaseAddress, Length, Callback, Context, OutOfProcessCallbackDll)`

Windows x64 data-based unwinding [2]

Но нам не так важен совсем низкий уровень, у нас же есть удобный API:

- `PRUNTIME_FUNCTION RtlLookupFunctionEntry(ControlPc, ImageBase, HistoryTable)`
- `PEXCEPTION_ROUTINE RtlVirtualUnwind(HandlerType, ImageBase, ControlPc, FunctionEntry, ContextRecord, *HandlerData, EstablisherFrame, ContextPointers)`

Windows x64 data-based unwinding [3]

```
int getNativeTrace(CONTEXT ctxt, const void **callchain, StackBounds& bounds) {
   int depth = 0;
   while (is_valid_context(&ctx, bounds)) {
       callchain[depth++] = (const void *) ctx.Rip;
       uint64_t image_base;
       auto *entry = RtlLookupFunctionEntry(ctx.Rip, &image_base, nullptr);
       if (entry) {
           void *data;
           DWORD64 establisher;
           RtlVirtualUnwind(0, image_base, ctx.Rip, entry, &ctx, &data, &establisher, nullptr);
       } else {
           ctx.Rip = *reinterpret_cast<uint64_t const *>(ctx.Rsp);
           ctx.Rsp += sizeof(uint64_t);
       }
   }
   return depth;
}
```

Windows x64 data-based unwinding [3]

```
int getNativeTrace(CONTEXT ctxt, const void **callchain, StackBounds& bounds) {
   int depth = 0;
   while (is_valid_context(&ctx, bounds)) {
       callchain[depth++] = (const void *) ctx.Rip;
       uint64_t image_base;
       auto *entry = RtlLookupFunctionEntry(ctx.Rip, &image_base, nullptr);
       if (entry) {
           void *data;
           DWORD64 establisher;
           RtlVirtualUnwind(0, image_base, ctx.Rip, entry, &ctx, &data, &establisher, nullptr);
       } else {
           ctx.Rip = *reinterpret_cast<uint64_t const *>(ctx.Rsp);
           ctx.Rsp += sizeof(uint64_t);
       }
   }
   return depth;
}
```

Windows x64 data-based unwinding [4]

Чёртовы ~~гуки~~ локи, они научились прятаться даже тут!

```
thread #5, name = 'sampler thread'
frame #0: ntdll.dll`ZwWaitForAlertByThreadId + 20
frame #1: ntdll.dll`RtlLookupFunctionEntry + 1728
frame #2: ntdll.dll`RtlLookupFunctionEntry + 73
frame #3: profiler.exe`getNativeTrace
frame #4: profiler.exe`sampler
```

```
thread #6, name = 'dll loader', state = SUSPENDED
frame #0:  ntdll.dll`RtlInitializeCriticalSectionEx+721
frame #1:  ntdll.dll`RtlGetExtendedContextLength + 249
frame #2:  ntdll.dll`RtlPcToFileHeader + 5037
frame #3:  ntdll.dll`RtlFreeUnicodeString + 1179
frame #4:  ntdll.dll`RtlDosPathNameTo... + 806
frame #5:  ntdll.dll`LdrShutdownThread + 5120
frame #6:  ntdll.dll`LdrShutdownThread + 1685
frame #7:  ntdll.dll`RtlFreeUnicodeString + 1248
frame #8:  ntdll.dll`RtlGetVersion + 775
frame #9:  ntdll.dll`LdrGetDllHandleByMapping + 2136
...............
frame #14: ntdll.dll`LdrLoadDll + 228
frame #15: KernelBase.dll`LoadLibraryExW + 368
frame #16: KernelBase.dll`LoadLibraryExA + 49
frame #17: KernelBase.dll`LoadLibraryA + 63
frame #18: profiler.exe`foreverLoadDllLoop
```

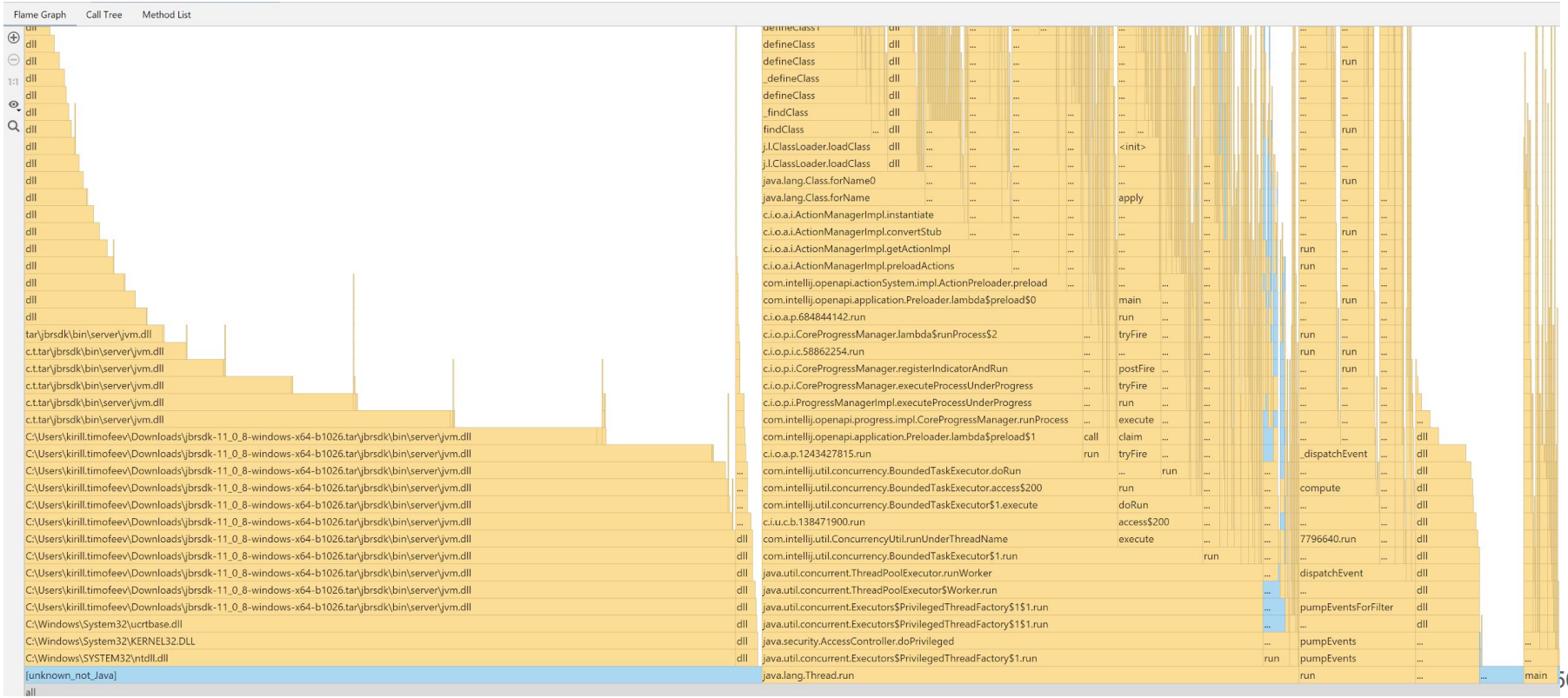
Windows x64 data-based unwinding [4]

Чёртовы ~~гуки~~ локи, они научились прятаться даже тут!

```
thread #5, name = 'sampler thread'
frame #0: ntdll.dll`ZwWaitForAlertByThreadId + 20
frame #1: ntdll.dll`RtlLookupFunctionEntry + 1728
frame #2: ntdll.dll`RtlLookupFunctionEntry + 73
frame #3: profiler.exe`getNativeTrace
frame #4: profiler.exe`sampler
```

```
thread #6, name = 'dll loader', state = SUSPENDED
frame #0:  ntdll.dll`RtlInitializeCriticalSectionEx+721
frame #1:  ntdll.dll`RtlGetExtendedContextLength + 249
frame #2:  ntdll.dll`RtlPcToFileHeader + 5037
frame #3:  ntdll.dll`RtlFreeUnicodeString + 1179
frame #4:  ntdll.dll`RtlDosPathNameTo... + 806
frame #5:  ntdll.dll`LdrShutdownThread + 5120
frame #6:  ntdll.dll`LdrShutdownThread + 1685
frame #7:  ntdll.dll`RtlFreeUnicodeString + 1248
frame #8:  ntdll.dll`RtlGetVersion + 775
frame #9:  ntdll.dll`LdrGetDllHandleByMapping + 2136
.................
frame #14: ntdll.dll`LdrLoadDll + 228
frame #15: KernelBase.dll`LoadLibraryExW + 368
frame #16: KernelBase.dll`LoadLibraryExA + 49
frame #17: KernelBase.dll`LoadLibraryA + 63
frame #18: profiler.exe`foreverLoadDllLoop
```

Windows x64 data-based unwinding: работает



Windows и её дебажные символы (pdb)

- Обычно надо качать, а для OpenJDK ещё и самим генерить

Windows и её дебажные символы (pdb)

- Обычно надо качать, а для OpenJDK ещё и самим генерить
- У нас уже есть возможность скачивать сами JDK и индексы для них прямо из IDEA

Windows и её дебажные символы (pdb)

- Обычно надо качать, а для OpenJDK ещё и самим генерить
- У нас уже есть возможность скачивать сами JDK и индексы для них прямо из IDEA
- В будущем научимся генерить и публиковать символы для себя (для JBR) и, вероятно, для других JDK тоже

Слайд с благодарностями

Спасибо всем моим прекрасным коллегам за помощь!

И лично:

- Роману Артемьеву – https://youtu.be/PYzsn8Mt7mE
- Семёну Огороднику – https://cutt.ly/khi1OCr
- Михаилу Пилину – https://pilin.name

Выводы:

- Windows – интересная операционная система
- Почти всегда есть план Б
- Async-profiler на Windows – уже скоро в IntellIJ IDEA

Всем спасибо, вопросики????

По ту сторону монитора сидел:

- Кирилл Тимофеев
- kirill.timofeev@jetbrains.com
- https://twitter.com/Kirill_Tim_