PERFORMANCE CHARACTERIZATION AND OPTIMIZATIONS IN GRAAL AT INTEL

Jean-Philippe Halimi - jean-philippe.halimi@intel.com

Code Generation and Optimization 2019 – Washington DC

February 16th, 2019

Legal Disclaimer & Optimization Notice

The benchmark results reported above may need to be revised as additional testing is conducted. The results depend on the specific platform configurations and workloads utilized in the testing, and may not be applicable to any particular user's components, computer system or workloads. The results are not necessarily representative of other benchmarks and other benchmark results may show greater or lesser impact from mitigations.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more complete information visit www.intel.com/benchmarks.

INFORMATION IN THIS DOCUMENT IS PROVIDED "AS IS". NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO THIS INFORMATION INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.
Notice revision #20110804

Introduction

- **About me**
  - Software Engineer at Intel since 2014
  - Contributed to several compilers and runtime systems
  - Involved in Java* optimizations since early 2017

- **What we will present**
  - Performance comparison: C2* vs. Graal* CE
  - Initial analysis and contributions

Our vision

“As Graal moves forward, our goal is to leverage its performance to be at least on par with C2 on Intel platforms.”

Optimization Notice

Performance comparison: Graal CE vs. C2

- **Initial analysis on selected benchmarks**
  - SPECjvm2008*
  - JDK* micro-benchmarks suite

- **Workloads of interest moving forward**
  - Java frameworks (Apache Hadoop*, Apache Kafka*…)
  - Cloud and Serverless
  - Real-world Java applications

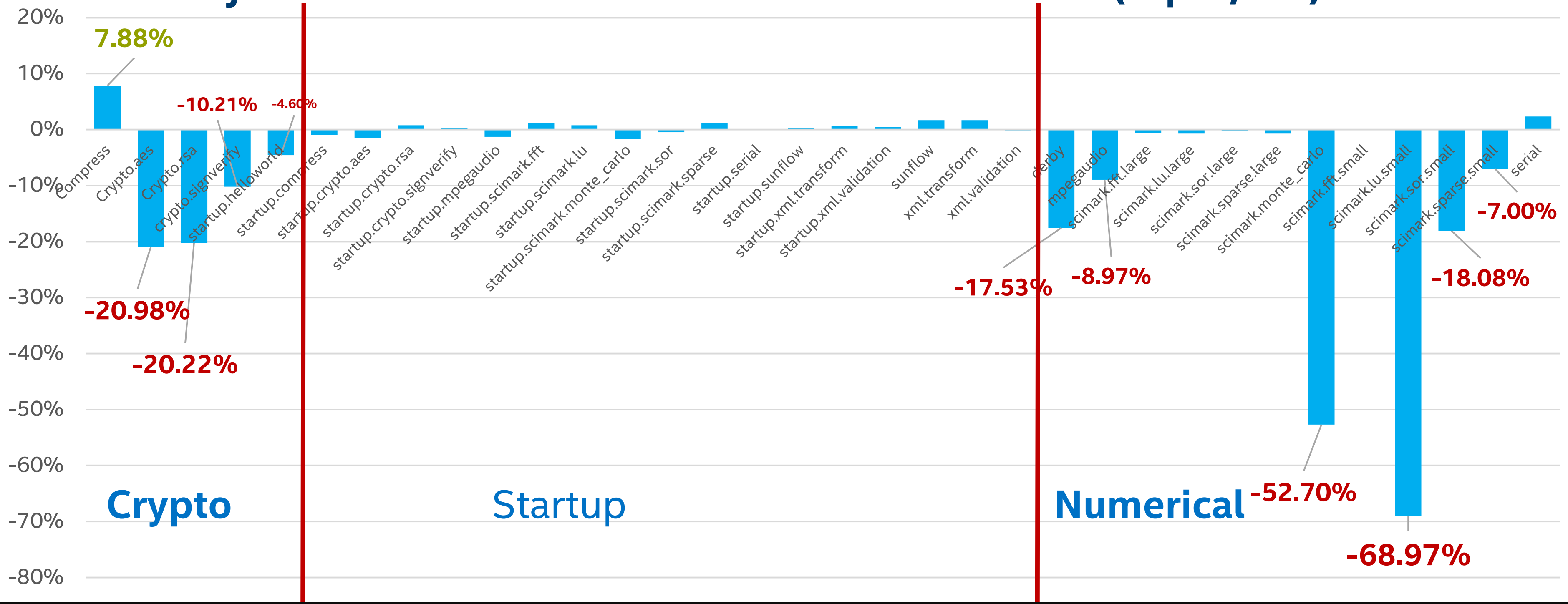
Setup

Linux* Kernel	4.15.6-300
Sockets	2S Intel® Xeon® Platinum 8180
OpenJDK*	11.0.1

Java Options	`-Xmx25G -Xms25G -Xmn15G –XX:+UseParallelGC`
	`-XX:+UnlockExperimentalVMOptions`

SPECjvm2008	`-ict -ivk -wt 60 -i 5 -bt 56`
NUMA control	`--cpunodebind=1 --localalloc`
C2	Default 11.0.1
Graal	Graal CE tip (Mid-December 2018)

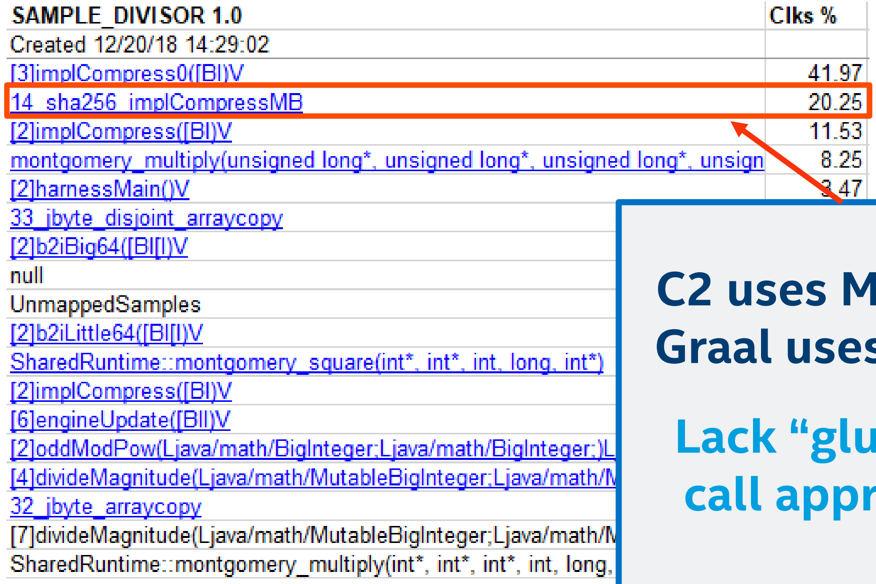
SPECjvm2008: Graal CE vs. C2 score (ops/m)



Source: Intel, as of February 16th, 2019. The benchmark results reported above may need to be revised as additional testing is conducted. The results depend on the specific platform configurations and workloads utilized in the testing, and may not be applicable to any particular user's components, computer system or workloads. The results are not necessarily representative of other benchmarks and other benchmark results may show greater or lesser impact from mitigations Results based on Intel measurements and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance/datacenter. Configuration: See slide 4.

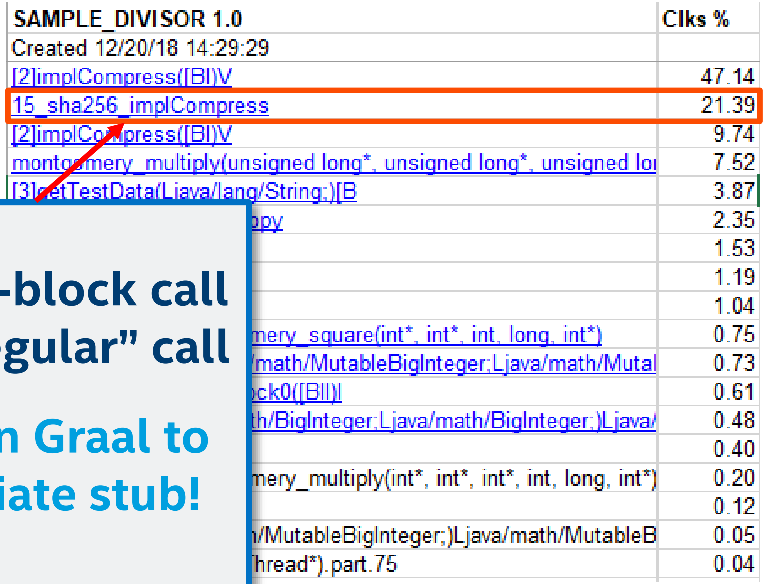
Crypto regressions: Signverify as example (-10%)

C2 hotspots



SAMPLE_DIVISOR 1.0	Clks %
Created 12/20/18 14:29:02	
[3]implCompress0([BI)V	41.97
14_sha256_implCompressMB	20.25
[2]implCompress([BI)V	11.53
montgomery_multiply(unsigned long*, unsigned long*, unsigned long*, unsign	8.25
[2]harnessMain()V	3.47
33_jbyte_disjoint_arraycopy	
[2]b2iBig64([BI[I)V	
null	
UnmappedSamples	
[2]b2iLittle64([BI[I)V	
SharedRuntime::montgomery_square(int*, int*, int, long, int*)	
[2]implCompress([BI)V	
[6]engineUpdate([BII)V	
[2]oddModPow(Ljava/math/BigInteger;Ljava/math/BigInteger;)L	
[4]divideMagnitude(Ljava/math/MutableBigInteger;Ljava/math/M	
32_jbyte_arraycopy	
[7]divideMagnitude(Ljava/math/MutableBigInteger;Ljava/math/M	
SharedRuntime::montgomery_multiply(int*, int*, int*, int, long,	

Graal CE hotspots



SAMPLE_DIVISOR 1.0	Clks %
Created 12/20/18 14:29:29	
[2]implCompress([BI)V	47.14
15_sha256_implCompress	21.39
[2]implCompress([BI)V	9.74
montgomery_multiply(unsigned long*, unsigned long*, unsigned lo	7.52
[3]getTestData(Ljava/lang/String;)[B	3.87
ppy	2.35
	1.53
	1.19
	1.04
mery_square(int*, int*, int, long, int*)	0.75
/math/MutableBigInteger;Ljava/math/Mutal	0.73
ck0([BII)I	0.61
th/BigInteger;Ljava/math/BigInteger;)Ljava/	0.48
	0.40
mery_multiply(int*, int*, int*, int, long, int*)	0.20
	0.12
/MutableBigInteger;)Ljava/math/MutableB	0.05
hread*).part.75	0.04

C2 uses Multi-block call
Graal uses "regular" call

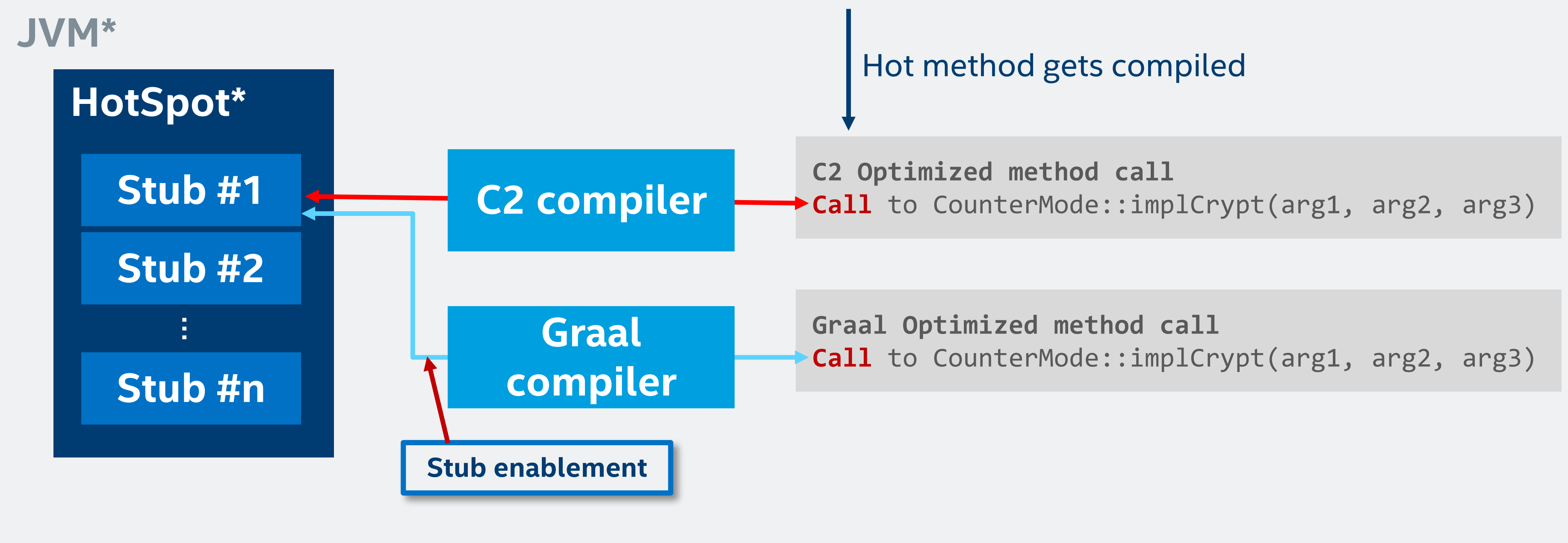
Lack "glue" in Graal to call appropriate stub!

Source: Intel, as of February 16th, 2019. The benchmark results reported above may need to be revised as additional testing is conducted. The results depend on the specific platform configurations and workloads utilized in the testing, and may not be applicable to any particular user's components, computer system or workloads. The results are not necessarily representative of other benchmarks and other benchmark results may show greater or lesser impact from mitigations Results based on Intel measurements and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance/datacenter. Configuration: See slide 4.

Stub enablement

- Stubs are highly optimized routines often written in assembly that replace regular compiled method calls.

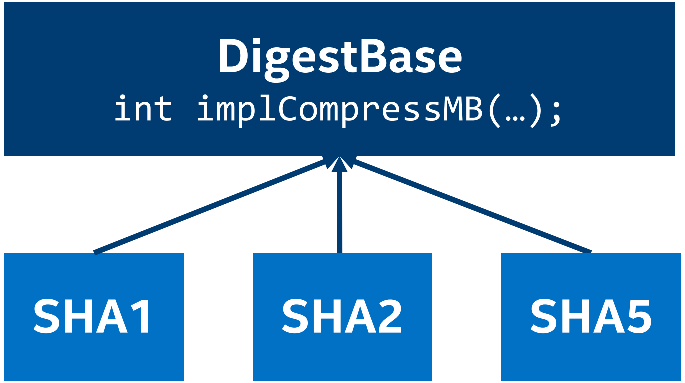


```
C2 Optimized method call
Call to CounterMode::implCrypt(arg1, arg2, arg3)
```

```
Graal Optimized method call
Call to CounterMode::implCrypt(arg1, arg2, arg3)
```

Crypto regressions: Signverify as example

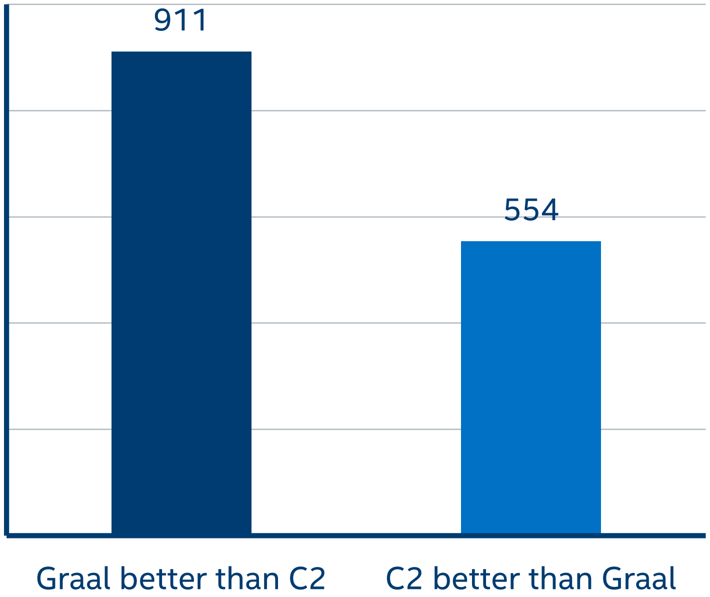
- **PR #924 enables DigestBase::implCompressMB for crypto.signverify**
  - Intrinsifies base class method
  - Retrieves run-time derived class
  - Converts implCompressMB arguments types to the corresponding Hotspot stubs
  - Returns the appropriate value
    - ➢ **163 lines** of code
    - ➢ Currently being debugged…



https://github.com/oracle/graal/pull/924

JDK micro-benchmarks suite

- Measures performance of specific features of the Java VM
- ~1500 tests (as of November 2018)
- **C2 vs. Graal reveals regressions in several benchmark categories**
    - Crypto operations
    - XML parsing
    - Stream operations



http://hg.openjdk.java.net/code-tools/jmh-jdk-microbenchmarks

Source: Intel, as of February 16th, 2019. The benchmark results reported above may need to be revised as additional testing is conducted. The results depend on the specific platform configurations and workloads utilized in the testing, and may not be applicable to any particular user's components, computer system or workloads. The results are not necessarily representative of other benchmarks and other benchmark results may show greater or lesser impact from mitigations Results based on Intel measurements and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance/datacenter. Configuration: See slide 4.

AESGCMBench.decrypt micro-benchmark

- 95% regression over C2 (JDK-8213264)
- **Two stubs** are not applied in Graal
  - GHASH::processBlocks(…) (PR #857), CounterMode::implCrypt(…) (PR#887)

C2 profile

```
....[Hottest Regions]..............................
 39.95% runtime stub StubRoutines::ghash_processBlocks
 18.89% runtime stub StubRoutines::counterMode_AESCrypt
  4.23% [kernel.kallsyms] [unknown]
  4.08% runtime stub StubRoutines::jbyte_disjoint_arr…
  3.32% runtime stub StubRoutines::jlong_disjoint_arra…
  2.72% c2, level 4
com.sun.crypto.provider.GaloisCounterMode::decryptFinal
  1.82% c2, level 4
com.sun.crypto.provider.GaloisCounterMode::decryptFinal
  1.22% runtime stub
StubRoutines::aescrypt_encryptBlock (181 bytes)
```

Graal profile

```
....[Hottest Regions]...................................
 25.03% jvmci, level 4 ….crypto.provider.GHASH::blockMult
  4.65% [kernel.kallsyms] [unknown]
  2.20% [kernel.kallsyms] [unknown]
  1.79% jvmci, level 4
com.sun.crypto.provider.GaloisCounterMode::doLastBlock
  1.59% runtime stub StubRoutines::aescrypt_encryptBlock
  1.54% [kernel.kallsyms] [unknown]
  1.34% jvmci, level 4 …crypto.provider.GHASH::processBlocks
  1.32% [kernel.kallsyms] [unknown]
  1.26% [kernel.kallsyms] [unknown]
  1.03% libjvm.so
_ZN14ElfSymbolTable6lookupEPhPiS1_S1_P16ElfFuncDescTable
```

Source: Intel, as of February 16th, 2019. The benchmark results reported above may need to be revised as additional testing is conducted. The results depend on the specific platform configurations and workloads utilized in the testing, and may not be applicable to any particular user's components, computer system or workloads. The results are not necessarily representative of other benchmarks and other benchmark results may show greater or lesser impact from mitigations Results based on Intel measurements and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

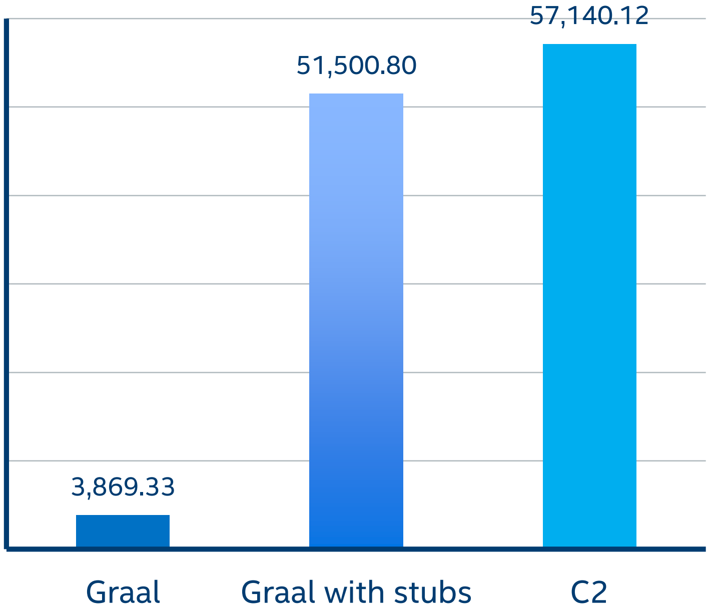
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance/datacenter. Configuration: See slide 4.

Optimization Notice

AESGCMBench.decrypt micro-benchmark

- **Implemented and merged two patch enabling the stubs**
  - Improved performance by >13x
  - ~ [50 – 150] lines
  - Still ~11% regression remaining over C2

https://github.com/oracle/graal/pull/857
https://github.com/oracle/graal/pull/887



Source: Intel, as of February 16th, 2019. The benchmark results reported above may need to be revised as additional testing is conducted. The results depend on the specific platform configurations and workloads utilized in the testing, and may not be applicable to any particular user's components, computer system or workloads. The results are not necessarily representative of other benchmarks and other benchmark results may show greater or lesser impact from mitigations Results based on Intel measurements and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance/datacenter. Configuration: See slide 4.

Optimization Notice

Stubs analysis: C2 vs. Graal

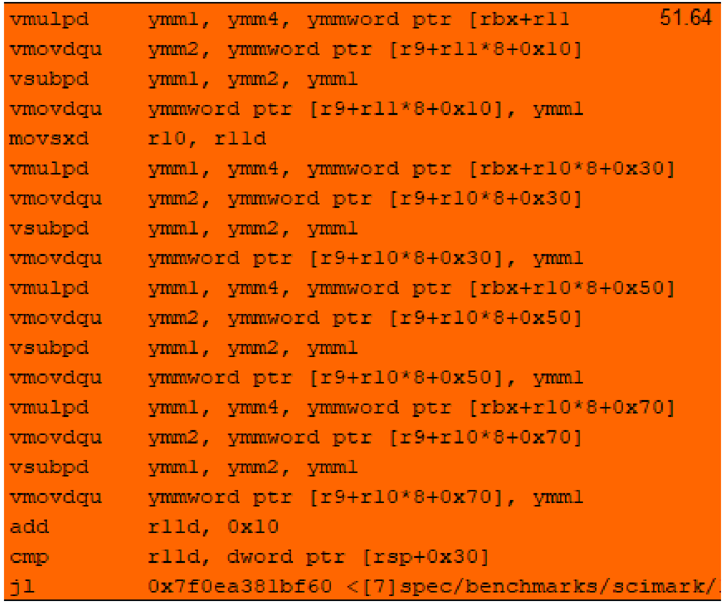
This stub is not the only example of missed optimized stubs in Graal.

- **Support missing for several methods in following classes**
  - CRC32, SHA, Adler (Crypto)
  - Math, Array, StringBuffer
  - Unsafe, MethodHandle, Reference

“Stub enablement is the easiest way to achieve impactful performance improvements in Graal.”

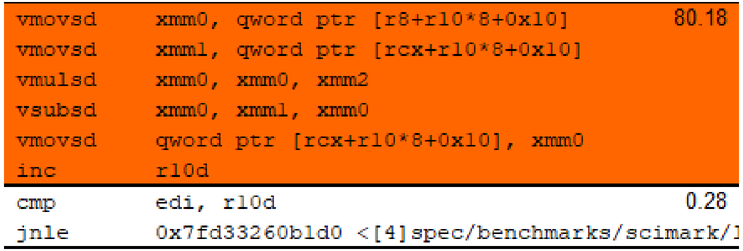
Numerical regressions: -68% scimark.lu.small

C2 LU.factor (79.8% ticks)



```
vmulpd      ymm1, ymm4, ymmword ptr [rbx+r11          51.64
vmovdqu     ymm2, ymmword ptr [r9+r11*8+0x10]
vsubpd      ymm1, ymm2, ymm1
vmovdqu     ymmword ptr [r9+r11*8+0x10], ymm1
movsxd      r10, r11d
vmulpd      ymm1, ymm4, ymmword ptr [rbx+r10*8+0x30]
vmovdqu     ymm2, ymmword ptr [r9+r10*8+0x30]
vsubpd      ymm1, ymm2, ymm1
vmovdqu     ymmword ptr [r9+r10*8+0x30], ymm1
vmulpd      ymm1, ymm4, ymmword ptr [rbx+r10*8+0x50]
vmovdqu     ymm2, ymmword ptr [r9+r10*8+0x50]
vsubpd      ymm1, ymm2, ymm1
vmovdqu     ymmword ptr [r9+r10*8+0x50], ymm1
vmulpd      ymm1, ymm4, ymmword ptr [rbx+r10*8+0x70]
vmovdqu     ymm2, ymmword ptr [r9+r10*8+0x70]
vsubpd      ymm1, ymm2, ymm1
vmovdqu     ymmword ptr [r9+r10*8+0x70], ymm1
add         r11d, 0x10
cmp         r11d, dword ptr [rsp+0x30]
jl          0x7f0ea381bf60 <[7]spec/benchmarks/scimark/
```

Graal CE LU.factor (91.7% ticks)



```
vmovsd      xmm0, qword ptr [r8+r10*8+0x10]            80.18
vmovsd      xmm1, qword ptr [rcx+r10*8+0x10]
vmulsd      xmm0, xmm0, xmm2
vsubsd      xmm0, xmm1, xmm0
vmovsd      qword ptr [rcx+r10*8+0x10], xmm0
inc         r10d
cmp         edi, r10d                                   0.28
jnle        0x7fd33260b1d0 <[4]spec/benchmarks/scimark/
```

- No Vectorizer (Graal CE)
- No Unrolling (in this case)

Source: Intel, as of February 16th, 2019. The benchmark results reported above may need to be revised as additional testing is conducted. The results depend on the specific platform configurations and workloads utilized in the testing, and may not be applicable to any particular user's components, computer system or workloads. The results are not necessarily representative of other benchmarks and other benchmark results may show greater or lesser impact from mitigations Results based on Intel measurements and are provided for informational purposes only. Any difference in system hardware or software design or configuration may affect actual performance.

Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance/datacenter. Configuration: See slide 4.

Optimization Notice

Initial analysis summary

- **Graal CE shows significant regressions over C2**
  - Some stubs are not present
    - We already merged 2, implCompressMB in progress (PR #924)
  - Core compiler optimizations are missing in CE
    - Vectorizer is #1

Next few steps

- **Contribution effort**
  - Pursue stub enablement contributions
- **Analysis**
  - Identify next optimization opportunities
  - AOT performance, startup time
  - Diversify workload portfolio

Call to action

- **We are looking for community support**
  - Participate to port stubs and intrinsics from C2 to Graal
  - Analysis effort
  - Collaborate on real-world Java applications

intel®

experience
what's inside™