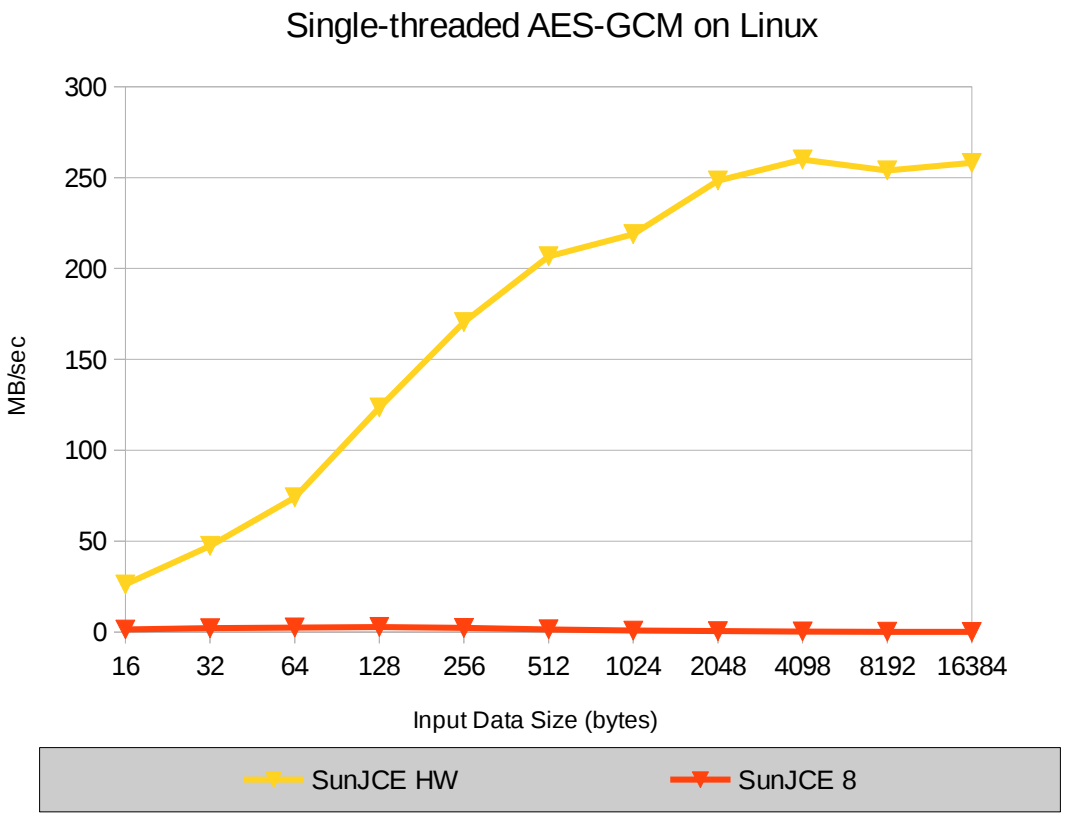
GCM Performance Analysis

JEP JDK-8046943
JVM Hardware Crypto Acceleration

The enclosed charts show the performance improvements to AES/GCM/NoPadding executing Galois Field Multiplication via HotSpot intrinsics. Hardware acceleration from both x86 and SPARC are used on the most time-consuming part of GHASH, carry-less multiplication, which is part of GCM mode. On x86 ISA instruction PCLMULQDQ and on SPARC ISA instructions xmul/xmulhi.

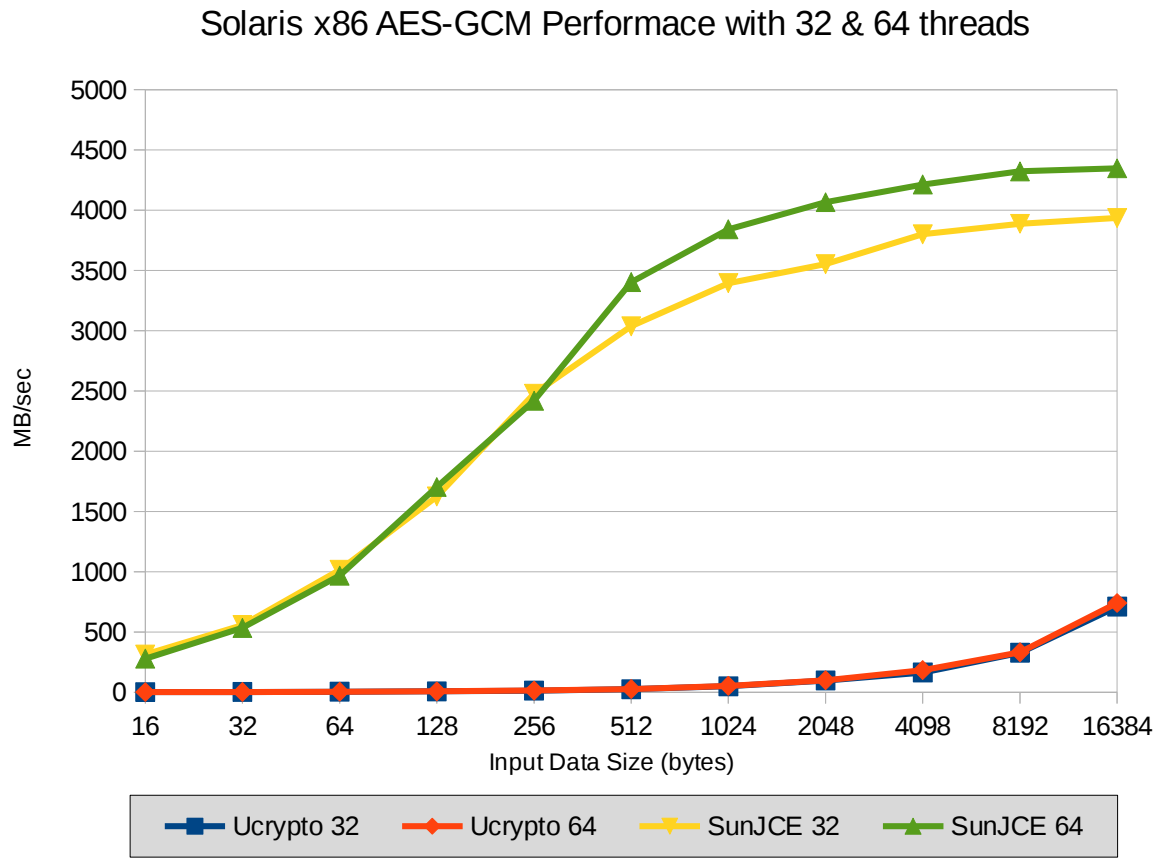
In the charts below, SunJCE HW denotes the intrinsified version. SunJCE 8 is the current shipping pure software implementation.

This first graph shows the dramatic gains from the SunJCE 8 to SunJCE HW using Linux. A 16 bytes encryption can perform at a rate of 26MB/sec with intrinsics, while the same 16 bytes in software 880KB/sec, over a 29x improvement. The numbers only improve as at 16K encryption, the performance is up to 258 MB/sec while can only manage 4MB/sec, a 61x improvement .



Multi-threaded Performance

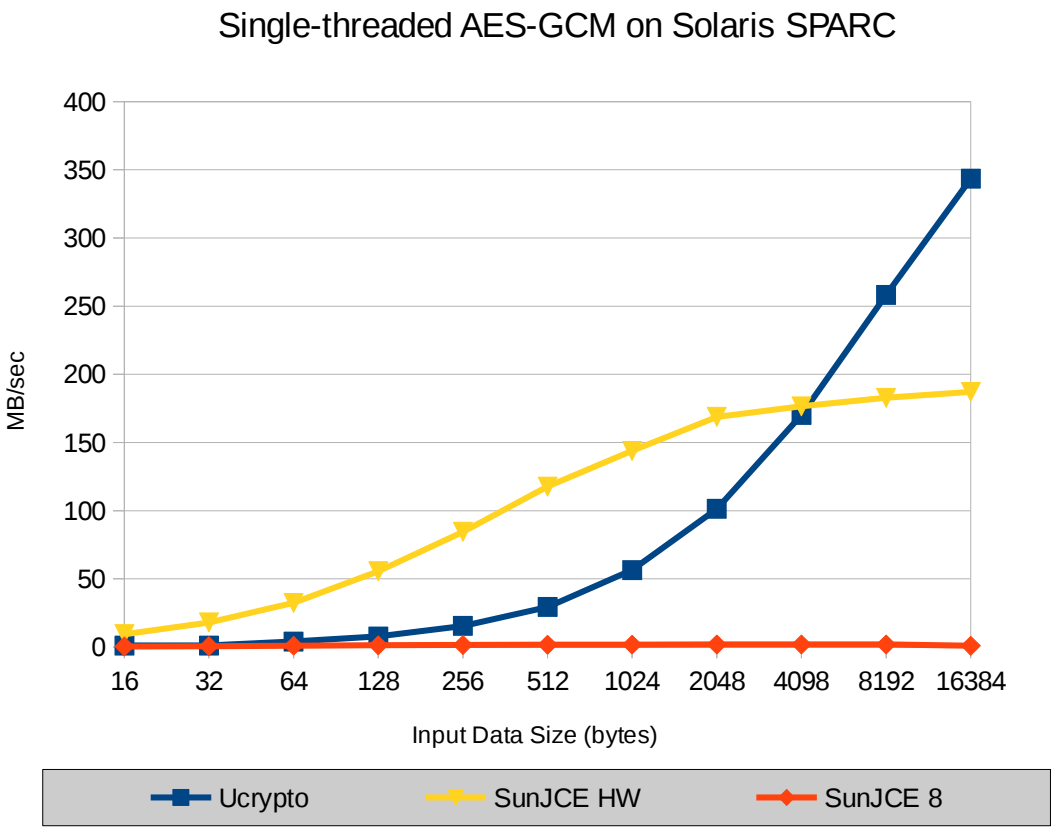
Multi-threaded performance also increased dramatically compared to JDK 8 GA. Looking at 32 and 64 threaded performance comparing SunJCE HW to the current shipping OracleUcrypto provider in 8. It shows how that 16 bytes data size are roughly 375MB/secs for both 32 and 64 threaded tests, reaching 5.8GB/secs for 32 threads and 7.6GB/secs for 64 threads at 16K data sizes. Ucrypto shows performance under 1MB/secs for 16 bytes data sizes and up to 800MB/secs for 16K data sizes. It is believed this low performance is because of problems in the OracleUcrypto provider and not because of OS, JCE, or HotSpot issues.



SPARC Performance

When looking at the SPARC results, we find performance differences. While multi-threaded shows similar numbers as x86, the single-threaded performance plateaus around 2K data sizes for SunJCE HW; meanwhile, OracleUcrypto continues on an upward path. Preliminary evaluation shows the performance is not related to the GCM changes.

Nevertheless, this performance is still dramatically better than JDK 8 GA.



Conclusion

The performance increase is large compared to JDK 8 GA, ranging from 34x to 150x. An upcoming bug fix (JDK-8069072) will improve the software-only version by 10x, but this doesn't diminish the MB/sec and GB/sec throughput we will get from hardware acceleration and the better CPU utilization. Bugs will be filed against OracleUcrypto for it's multi-threaded problems, and fixes will probably look more like the single-threaded graphs.

The goal of this GCM work is to close and surpass native library performance. In many cases that has

been met. More work can be done given some places where native libraries were still faster, but that will require further analysis. Additional enhancements of parallelizing algorithms can help performance and will be considered for future enhancements. Plans to move AES-GCM from SunJCE to the top of the java.security provider list is a “go”, as the overall performance is better.