White Paper

Healthcare and Life Sciences
Image Compression

Low-Latency Lossless Compression of Monochrome Medical Images

This paper surveys potential codecs for the lossless compression of 16-bit RAW monochrome medical images, assesses the strengths and shortcomings of each in that context and discusses optimization approaches to improve compression ratio and speed. Based on research and performance testing, the paper recommends High Throughput JPEG 2000 (HTJ2K) for this purpose. To extend the value of that recommendation, the paper also explores potential optimizations of HTJ2K to reduce encoding time.

GE Healthcare

Authors

Beenish Zia*, Palanivel Guruvareddiar*, Yong Tong Chua*, Xiang Guo*, Sun Jian*, Livio Lattanzio*, Jerome Knoplioch§

* Intel Corporation
§ GE Healthcare

Table of Contents

Modern clinical practices rely on the ability to efficiently and cost-effectively capture, store and manipulate growing bodies of patient image data such as ultrasound, CT and MRI studies. As essential building blocks for that effort, low-latency codecs that work across cloud and edge infrastructure play a fundamental role in reducing the storage space and bandwidth requirements for a given set of images, as well as the computing resources needed. Adding to the solution challenges, practical and regulatory considerations require this compression to be lossless.

Efficiency is measured in terms of both compression ratio and encode/decode time. In this context, encoding and decoding time act in part as proxies for resource requirements, meaning that faster encode and decode times equate to handling a given workload with a smaller amount of computing hardware, which reduces both the CapEx and the OpEx associated with supporting that work.

A common approach for enabling remote radiologists to view medical images is to provide them with secure access to a cloud instance using a web-based application. This approach has the advantage of flexibility in providing access from any geographic location, using any type of device, as illustrated in Figure 1. It can easily be adapted to a variety of use cases, including efficient access to patient studies from anywhere within a clinical network and beyond, cost-effective retrieval from long-term storage and image sharing for collaborative studies.

In this type of topology, abundant upload bandwidth is typically available, and the cost of that bandwidth is typically covered by the cloud service provider (CSP), as opposed to the solution provider. Download bandwidth resources may be less assured, as they are outside the control of the solution provider, as is the client user equipment (UE). Nevertheless, solutions must transcode input images or videos to low latency, lossless bit streams and make them available to remote users. These considerations require that solution providers select efficient codecs that are independent of client hardware.

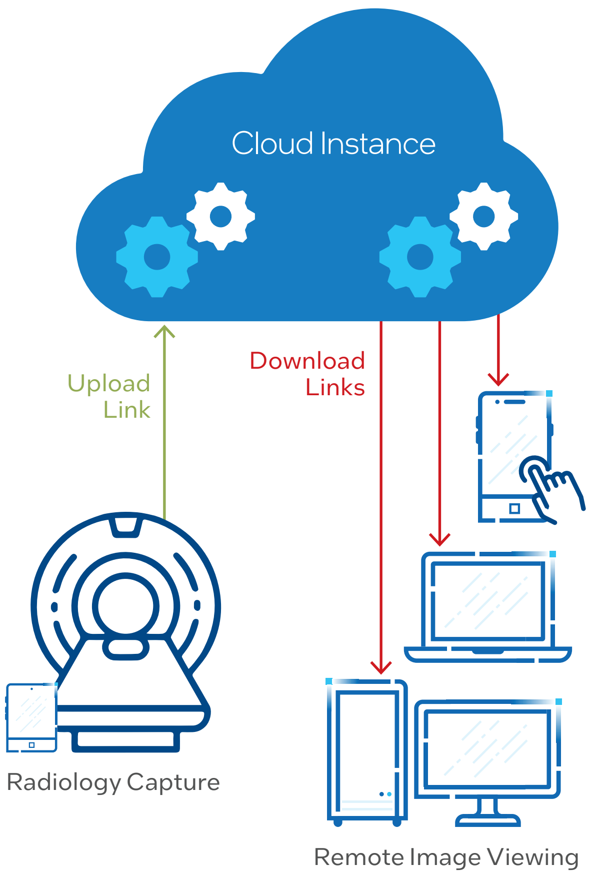

Figure 1. High-level typical transmission paths for medical image data.

To enable that effort, engineering teams from Intel and GE Healthcare have collaborated to analyze, test and optimize common codecs for use in the processing and delivery of medical images. As a basis for assessing the strengths and shortcomings of each codec, the teams established a set of four experimental workloads, each consisting of monochrome still images or video streams. These four test cases (named Ct1, Mg1, Mg2 and 01200L1c) are used as the standard of comparison throughout this paper. Each of the raw images used in the test case are arrays of short, signed values (signed 16 bits) that, after using linear transfer functions, could be viewed as shown in Figure 2.

The dimensions for each of these test cases were different, providing a good comparison on how various file sizes with different alignments are processed by the different compression codecs analyzed in this work. The nominative overall **objective of this study was to provide the basis for encoding a 512x512 16-bit** monochrome image from RAW format **with approximately 1 millisecond (ms) latency or less and a lossless compression ratio of greater than 2**.

At the same time, the approach used must meet regulatory requirements for compression of medical images to be lossless at the bit level. This mandate is in contrast to many usages that can use "visually lossless" modes, defined as having image deterioration that is imperceptible to humans. The higher computational requirements for lossless compared to lossy compression contribute to the challenges of delivering high compression ratios with fast encoding and decoding times.

In presenting the teams' research and testing results, this paper first establishes baseline performance for H.264 and H.265 on the four test cases, as the basis for comparison with other codecs. It then evaluates JPEG 2000, JPEG XL and HTJ2K for use in web-delivered solutions for access to medical images. Based on the assessment that HTJ2K is the codec of choice for these solutions, the paper then introduces the process of optimizing HTJ2K for Intel® Xeon® Scalable processors.

Baseline Assessment: H.264 and H.265

Initial testing with H.264 and H.265 for medical image compression yielded poor results, as shown in Figure 3. The team identified that the maximum bit depth supported by both H.264 and H.265 is 14 bits, compared to the 16-bit monochrome RAW images used in testing. Further, lossless encoding is supported only in YUV 4:2:0/4:2:2/4:4:4 models, and not with monochrome. To circumvent these limitations, the team treated the original 16-bit monochrome data as YUV 4:2:0, 8-bit data. In the initial testing, H.264 generated compressed size results that were larger than the corresponding uncompressed sizes in three out of four test cases. H.265 meanwhile provided just 10% to 20% space savings in three test cases and approximately 40% in the fourth case. Here and throughout this paper, compression ratio is defined as **(original size/compressed size)**, and space savings is defined as **(1-(compressed size/ original size))**. Encoder settings were lossless, zero latency and superfast.

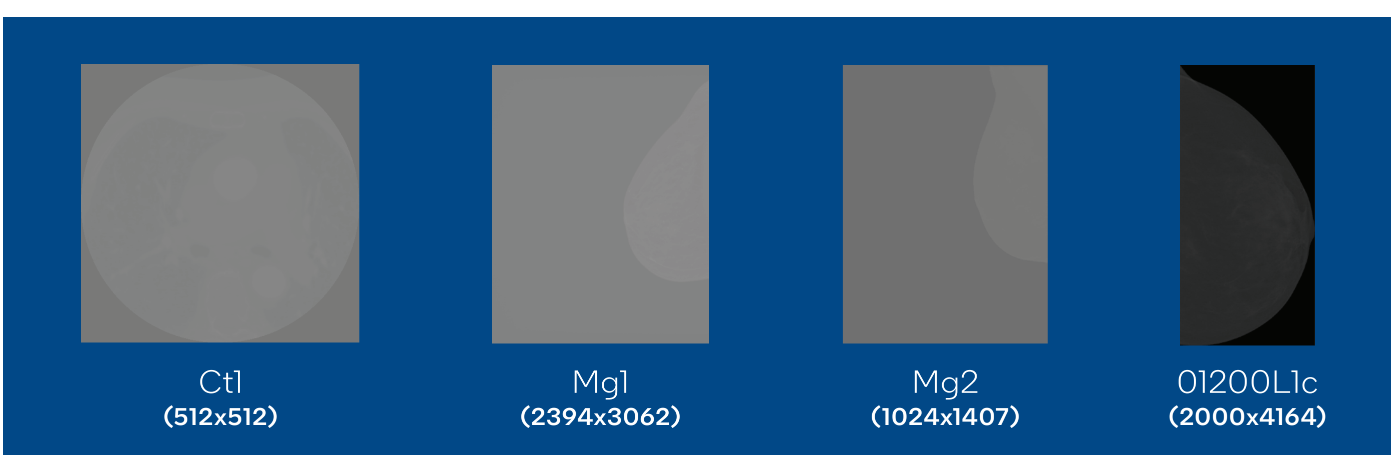

Figure 2. Representations of test cases used in this paper.

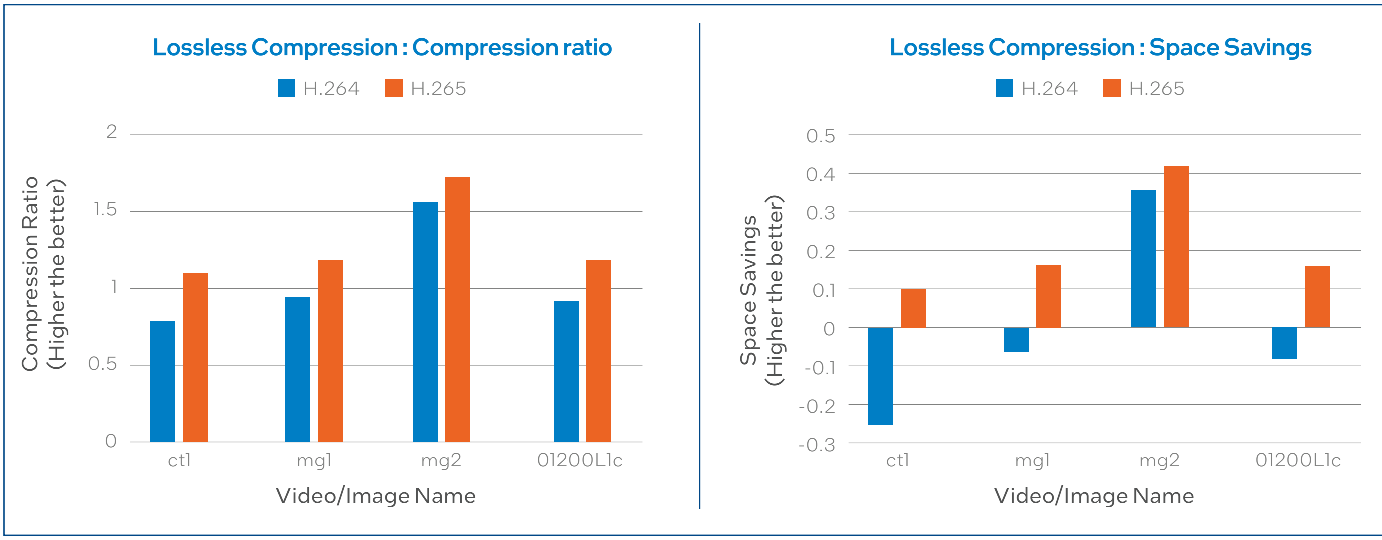

Figure 3. Preliminary 16-bit lossless compression testing with H.264 and H.265.[1]

Investigation of these results and the assessment of data revealed high correlation among the most significant bits (MSBs) and high similarity among the least significant bits (LSBs) of the 16-bit representations, compared to the similarity between MSB and LSB, which a typical codec would try to take advantage of. The similarity between adjacent frames could be exploited, potentially providing for interframe redundancy gains. In this context, frames can be adjacent spatially, as in a 3D model, or temporally, as in a video feed.

The team developed a novel approach: Before encoding, bits are realigned to treat the MSB as the Y plane, half the LSB as the Cb plane and the remaining half of the LSB as the Cr plane, and all planes have 8-bit data. After decoding, the transfer function realigns the data back to its original 16-bit form. This process optimization substantially improved compression testing results, as shown in Figure 4. Here, both H.264 and H.265 generated compression ratios of 40% to 60% for three of the four test cases and approximately 90% for the fourth case. Results using AV1 are provided as a further point of comparison, which slightly underperformed both H.264 and H.265 in all four test cases.

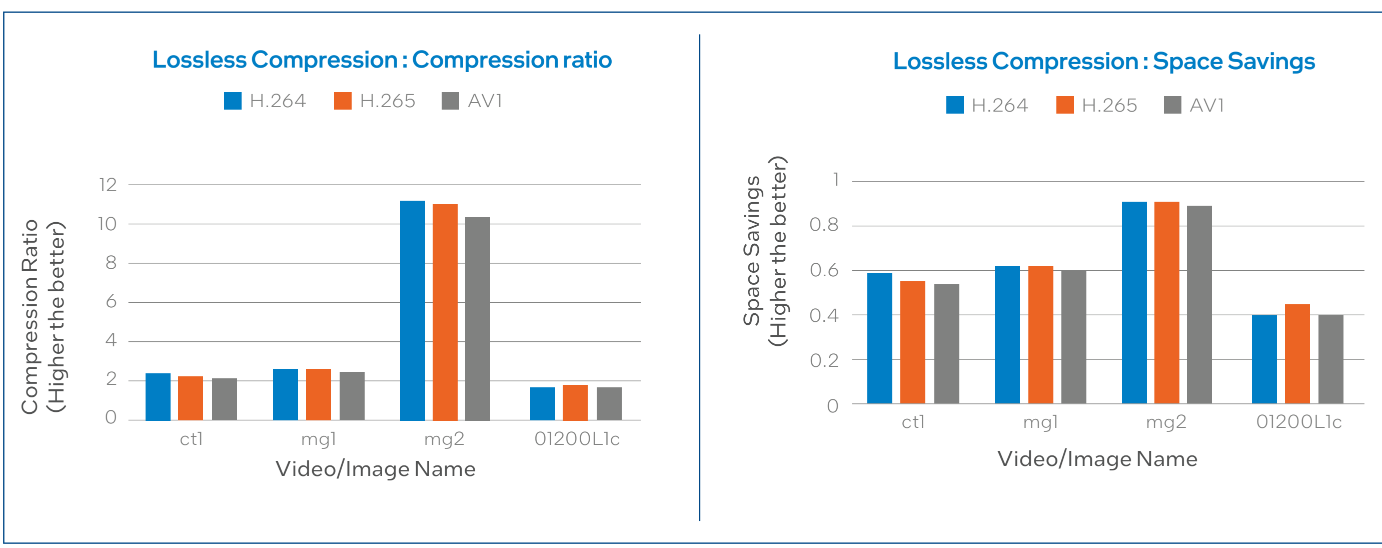

Figure 4. Optimized 16-bit lossless compression testing with H.264, H.265 and AV1.[1]

Survey of Codec Suitability

The team assessed various codecs for compressing the defined RAW 16-bit monochrome images and video used in this study, including the following:

- **JPEG 2000** was finalized in 2000 as a more flexible alternative to the older JPEG (Joint Photographic Expert Group) standard; it is widely used for diagnostic medical images.
- **JPEG XL** is a successor to JPEG 2000 that is optimized for delivery to a wide variety of end-user devices over the internet, including computationally efficient software encoding and decoding. The L in JPEG XL stands for long-term, which reflects the authors' intention for JPEG XL to act as a long-term replacement for legacy JPEG. JPEG XL is attractive for the healthcare industry because of it becoming officially part of the DICOM standard.
- **JPEG-LS** provides high-efficiency lossless compression with low computational complexity, enabling the standard to be implemented with low hardware footprint requirements. The LS in JPEG-LS stands for Lossless Standard.
- **High-Throughput JPEG 2000 (HTJ2K)** offers an order of magnitude or more improvement[2] over JPEG 2000 on encoding throughput by means of a new drop-in replacement encoder. Though currently not a DICOM standard, the DICOM committee is considering HTJ2K.

This work used FFmpeg (Fast Forward Moving Picture Experts Group), an open source set of tools and libraries for processing media files, including transcoding and basic editing. It also drew on OpenJPEG, an open source library for encoding and decoding JPEG 2000 images.

JPEG 2000

Both lossless and lossy compression are provided by a single architecture in the JPEG 2000 standard, which supports a maximum bit depth of 38 bits per component. Lossless compression is made possible by a reversible integer wavelet transform. Limitations arise when applying JPEG 2000 compression to the test workloads. In addition to the codec being computationally intensive, adding the OpenJPEG library to FFmpeg provides support only for JPEG 2000-supported file formats. The team was unable to successfully use the available patchworks to add the jpegxl library to FFmpeg. Because OpenJPEG for JPEG 2000 compression does not support RAW format, JPEG 2000 was unable to meet the test requirements.

JPEG XL

The maximum bit depth for JPEG XL can push up to 24-bit integer true color or 32-bit floats. Contrary to the test requirements, the current GitHub JPEG XL executables do not support RAW format as input files. The team used the FFmpeg framework, which accepts RAW format files, and libjxl as the processing library to feed the grayscale 16-bit files, producing the JPEG XL lossless compression results given in Table 1. For this testing, Effort was configured at 3, with Rendering set to Progressive. Implementation of JPEG XL is enabled by native support across modern browsers, including Chrome, Firefox, Edge, Opera, Brave and Vivaldi. That support helps streamline the use of JPEG XL in web apps, enabling it across client devices. The team set goals for this testing for encode and decode times of less than 500 ms and compression ratios of at least 2. JPEG XL only partially meets these goals, with Mg1 and 01200L1c decoding times above 500 ms and encoding times approaching that level as well.

HTJ2K

Like JPEG 2000, HTJ2K has a mathematically lossless mode, and it provides a new high-throughput block coder drop-in replacement for the original JPEG 2000 Part 1 block coder. This new component may provide an order of magnitude improvement or more in lossless encoding and decoding times in some cases, compared to JPEG XL. Equally important, it uses the same encoding as JPEG 2000, which provides for reversible transcoding between the two standards. This capability reduces the cost and complexity of implementation while retaining flexibility for different usages. Results for lossless compression of the standardized medical images in this report, using 16-bit signed data and the 4:0:0 model, are given in Table 2. For these results, the team optimized the workload by cutting the image into several tiles; Intel AVX-512 instructions are used for the aligned tiles and Intel AVX2 instructions are used for the unaligned tiles.

Table 1. JPEG XL compression performance.[3]

Image	Width	Height	Original Size (Bytes)	JPEG-XL Size (Bytes)	Compression Ratio§	Encoding Time (ms)*	Decoding Time (ms)*
Ct1	512	512	524,288	124,188	4.22	81	98
Mg1	2394	3062	14,660,856	4,626,030	3.16	462	661
Mg2	1024	1407	2,881,536	202,399	14.23	127	179
01200L1c	2000	4164	16,656,000	6,948,239	2.39	411	746

§ Compression ratio = (original size/compressed size); higher is better.
* Single-thread performance.

Table 2. HTJ2K compression performance before optimizations.[4]

Image	Width	Height	Original Size (Bytes)	HTJ2K Size (Bytes)	Compression Ratio§	Encoding Time (ms)	Decoding Time (ms)
Ct1	512	512	524,288	159,713	3.28	15.63	11.45
Mg1	2394	3062	14,660,856	4,977,313	2.94	210.21	144.73
Mg2	1024	1407	2,881,536	229,863	12.53	23.10	19.67
01200L1c	2000	4164	16,656,000	7,332,627	2.27	179.88	131.85

§ Compression ratio = (original size/compressed size); higher is better.

While these results show slightly lower compression ratios across all four test workloads than with JPEG XL, all remain substantially above the test objective of 2 for this value. At the same time, all of the encoding and decoding times with HTJ2K are markedly lower than the results from the JPEG testing, with the improvement approaching an order of magnitude in some cases. These results support the objective of reducing the time and computing resources needed to handle medical images for end-user applications. They demonstrate the potential for cost-effectively delivering an improved end-user experience.

Unlike JPEG XL, HTJ2K does not enjoy native browser support, potentially complicating its use in web-based applications. This limitation can be addressed through the use of OpenJPH, an open-source implementation of the HTJ2K standard, that includes both encoder and encoder tools. The code is written in C++ and includes a lightweight JavaScript decoder to simplify adoption on web-based platforms. The color and wavelet transform steps can employ SIMD instructions on Intel platforms. More information is available at https://github.com/aous72/OpenJPH. The DICOM committee is considering HTJ2K as a DICOM standard (a status that JPEG XL already has). HTJ2K's potential approval would provide another compelling reason for its wide adoption by the healthcare industry, alongside its fast encoding and decoding with favorable compression ratios.

Optimization Opportunities for HTJ2K

High efficiency for lossless usages (including streaming) make HTJ2K the best choice considered in this study for use in medical image compression applications. To realize additional efficiency advantages and achieve the encoding time of ~1 ms for a 512x512 16-bit monochrome image, the teams are investigating optimizations for the HTJ2K codec on Intel Xeon Scalable processors. In particular, encode and decode operations are both serial applications, meaning that they can run only on a single core.

Parallelizing the code to enable multicore operation incurs substantial thread synchronization overhead, limiting the value of this approach. As image sizes continue to grow, however, the advantages of parallel execution may eventually outweigh that overhead, making this a more viable option. Independent of that, vectorizing calculations using SIMD instruction sets such as SSE and AVX offer an alternative approach to parallelization.

In cases of unaligned images — those where the edge of the image data doesn't line up with instruction set requirements — best practices call for the remaining width to be padded with zero-valued pixels to ensure 512-bit alignment. The critical metrics in this testing are encoding and decoding time. Ongoing optimization continues to work toward the goal of approximately 1 ms encoding for lossless compression of the Ct1 512x512 image. In an effort to improve results, two optimizations were investigated, the test results of which are given in Table 3:

- **Solution 1: The padding data is saved to the file and removed on the decode side.** This approach reduces encoding time by more than 84% for the 01200L1c workload, up to more than 91% for Mg1, with compression ratios nearly identical to the unoptimized case.
- **Solution 2: The padding data is removed by the encoder before the data is saved to the file.** This approach results in very slightly higher encoding times than solution 1, while retaining the nearly identical compression ratios to both Solution 1 and the unoptimized case.

Table 3. Optimized HTJ2K compression performance.[4]

Image	Width	Height	Solution 1		Solution 2	
			Compression Ratio§	Encoding Time (ms)	Compression Ratio§	Encoding Time (ms)
Ct1	512	512	3.28	1.3	3.28	1.3
Mg1	2394	3062	2.93	29.3	2.94	30.5
Mg2	1024	1407	12.51	2.9	12.51	2.9
01200L1c	2000	4164	2.27	28	2.27	28.8

§ Compression ratio = (original size/compressed size); higher is better.

Ongoing optimization efforts focus on Intel AVX-512 instructions to increase the amount of data that can be processed per processor clock cycle, as well as optimization of bit operations, compile flags and the algorithm itself. Intel Speed Select technology on the Intel Xeon Scalable processor may be used in the future to explore performance improvement for serial encoding/decoding.

Conclusion

HTJ2K provides a compelling combination of high compression ratio and low encoding/decoding time for lossless compression of medical images. This report finds that the HTJ2K standard, including the open source OpenJPH implementation, is a promising basis for these applications, while noting that browser support is limited.

Vectorization of HTJ2K workloads using SIMD instructions is a viable approach to optimizing encoding time, as is padding images to align with 512-bit instruction set requirements.

JPEG-XL provides comparable compression efficiency for the compression of medical images and better browser support, making it suitable for playback-focused scenarios, but at the cost of higher encode and decode times. Common video compression standards including H.264, H.265 and AV1 offer comparable compression efficiency for the images only when the pixel values are split across the MSB/LSB channels. With this modified pixel arrangement, they are expected to provide better compression for medical videos as they exploit temporal redundancy.

Contact your Intel representative at health.lifesciences@intel.com

Learn More

Intel Healthcare and Life Sciences Technology Solutions:
https://www.intel.com/content/www/us/en/healthcare-it/healthcare-overview.html

Intel Solutions Marketplace for GE Healthcare:
https://marketplace.intel.com/s/partner/a5S3b0000016NmKEAU/ge-healthcare?language=en_US

GE Healthcare Systems: https://www.gehealthcare.com/

A Blazing-Fast JPEG 2000: What is HTJ2K?: https://htj2k.com/

OpenJPH: An Open Source Implementation of HTJ2K: https://github.com/aous72/OpenJPH

[1] Testing completed March 2022 by Intel. System under test: Intel® Core™ i7-1185G7 processor (4 cores @ 3 GHz), 64 GB DDR4-3200, OS: Ubuntu 20.04 LTS on WSL-2, Kernel: 5.10.16.3-microsoft-standard-WSL2, FFMPEG: N-106744-g8449fbdf8e.

[2] Joint Photographic Experts Group (JPEG) Document N87018. "High Throughput JPEG 2000 (HTJ2K) and the JPH file format: a primer." https://ds.jpeg.org/whitepapers/jpeg-htj2k-whitepaper.pdf.

[3] Testing completed 6/15/2022 by Intel. System under test: 2x Intel® Xeon® Gold 6252N processors (24 cores per socket @ 2.30 GHz), 192 GB DDR4, 1.92 TB storage, BIOS version: 5.14, OS: Ubuntu 18.0.4.

[4] Testing completed 8/3/2022 by Intel. System under test: 2x 4th Gen Intel® Xeon® Scalable processors, (48 cores per socket @ 3.6 GHz), CPU QDF=QY5E, D0 stepping, PCH WDF=QY0U, Intel® Speed Select Technology disabled, 64 GB DDR5-4800, OS: Red Hat 8.5.0-3 Linux version 4.18.0-348.el8.x86_64.

1122/ADS/MESH/350494-001US