White Paper

# Unlocking Edge Performance with the Capgemini Engineering ENSCONCE/Intel® Smart Edge Open Platform

**In this white paper we document the increased performance that results when the Intel® Smart Edge Open toolkit is integrated into the Capgemini ENSCONCE multi-edge compute platform. Benchmarking data shows increased throughput and reduced latency and jitter in typical edge computing use-cases when utilizing Intel Smart Edge Open optimizations that leverage CPU pinning and I/O acceleration features found within Intel® Xeon® Scalable processors.**

## Authors

**Nilanjan Samajdar**
Capgemini Engineering

**Anurag Ranjan**
Intel

### Table of Contents



## Introduction

The adoption of new technologies such as machine learning and artificial intelligence is driving advances across industries and transforming business models. These technologies are also driving changes in network architecture. Visual computing and machine learning require large data sets to be processed quickly, which has typically been addressed by cloud computing. Robotics and automation, however, require low-latency communications and processing to enable real-time decision making. This limits the effectiveness of cloud computing as a solution. The emergence of Multi-access Edge Computing (MEC), which brings the benefits of cloud computing to the edge of the network, results in lower latency and bandwidth while retaining the flexibility and scalability of a cloud architecture.

The Capgemini ENSCONCE was developed to deliver a cloud-native MEC platform that interfaces with multiple access networks. Capgemini and Intel have collaborated on a new version of the ENSCONCE platform that integrates the Intel® Smart Edge Open toolkit. The combination allows service providers to rapidly develop and deploy edge services that increase computing and I/O performance while also reducing latency to support emerging solutions like automation and machine learning.

This whitepaper reviews the architecture of the new Capgemini ENSCONCE platform with the Intel Smart Edge Open toolkit. It presents a sample workload requiring high throughput, low latency, and low jitter, while running other computing workloads. The study uses Intel Smart Edge Open building blocks to apply workload pinning to specific cores and to apply SR-IOV to the networking stack. The results show the Capgemini ENSCONCE platform with Intel Smart Edge Open delivers significant improvements in throughput and latency even under increased CPU compute loads.

## Capgemini ENSCONCE Platform in Emerging Use Cases

The ENSCONCE platform addresses a variety of use cases that require reliable, low-latency communication between an end device and an edge application, and high compute density at the edge to reduce communication to a central cloud. These use cases are found in a variety of domains, including manufacturing, government, and transportation, among others. Robotics, automation, machine learning, and visual computing require large amounts of data to be processed quickly to allow real-time solutions. The rollout of 5G, with its higher bandwidth and lower latency than previous wireless standards, is increasing the need for edge computing solutions that can take advantage of speed and latency benefits of 5G.

The ENSCONCE platform can be located either at the network edge, such as at a wireless base station, or on-premises at a customer location. By locating computing capabilities at the edge of the network, the ENSCONCE platform avoids the long latencies and high bandwidth costs when sending data to a faraway datacenter for processing. By leveraging Kubernetes containerization, applications can be easily deployed and quickly scaled to meet demand.

The ENSCONCE platform is also well-suited to serve Vehicle to Everything (V2X) and other use cases where devices change location, often at the high speeds associated with transportation. To provide consistent quality of service, the ENSCONCE platform monitors the end user location and the network load and migrates the edge application to edge platforms at the closest location as necessary.

## ENSCONCE Platform Overview

The ENSCONCE Platform consists of two key functions:

- **ENSCONCE-Central** provides orchestration functions both for applications running locally, as well as on remote edge platforms. ENSCONCE-Central utilizes control plane components from Intel Smart Edge Open to enable resource discovery and application scheduling. This allows ENSCONCE to run applications where it is most efficient.
- **ENSCONCE-Edge** provides local platform and application management, including application scheduling and hardware accelerator management.

The ENSCONCE platform can run on bare-metal servers as well as virtual machines, depending on customer preference and design decisions. The ENSCONCE platform supports standard services, such as:

- Firewall, NAT, DNS
- Traffic management and load balancing
- Kubernetes and Docker for standardized application containerization and orchestration
- Host management services including logging, application tracing, and inventory
- Identity and Access Management
- Prometheus monitoring
- Logging framework

A high-level visualization of the ENSCONCE platform architecture is shown in Figure 1.

## Intel Smart Edge Open

Intel Smart Edge Open is a royalty-free edge computing software toolkit that enables highly optimized and performant edge platforms to on-board and manage applications and network functions with cloud-like agility across any type of network. It is built on standardized APIs and open source software tools. Intel Smart Edge Open enables developers using the ENSCONCE platform to:

- easily migrate applications from the cloud to the edge by abstracting network complexity.
- enable secure on-boarding and management of applications with an intuitive web-based GUI.
- leverage standards-based building blocks for functions such as access termination, traffic steering, multi-tenancy, authentication, telemetry, and appliance discovery and control.

The ENSCONCE Platform leverages performance-enhancing capabilities available in Intel Smart Edge Open, including:

**Single Root I/O Virtualization (SR-IOV) for Accelerated Networking**
SR-IOV allows a single physical PCI resource, such as an Ethernet controller, to be virtualized and allocated to specific application and network functions. This allows traffic to bypass the software switch layer of the hypervisor and to be delivered directly to the appropriate virtual environment, reducing software overhead and delivering network performance comparable to nonvirtualized environments.

**CPU Manager for Kubernetes (CMK) and Huge Pages**
CMK allows Kubernetes containers to be pinned to specific processor cores on Intel® Xeon® Scalable processors, which improves performance by keeping recently used data in the on-chip cache and reducing cache misses that degrade performance. Huge Pages improves memory allocation and management in virtualized applications, resulting in better performance for memory intensive edge applications.

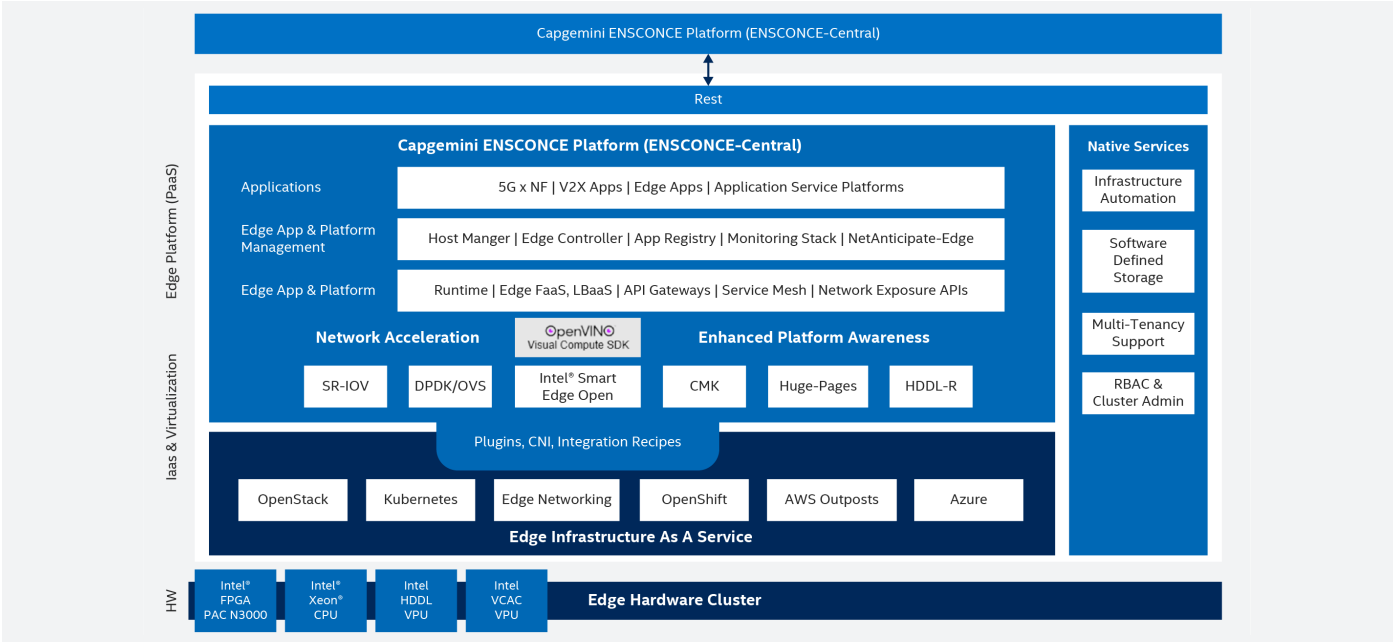


**Figure 1.** ENSCONCE platform architecture

**Open Virtual Switch (OVS) for Accelerated Networking**
Intel Smart Edge Open uses Data Plane Development Kit (DPDK), which provides a set of libraries and drivers for offloading packet processing from the kernel. By leveraging hardware features in Intel Xeon Scalable processors, DPDK results in higher computing efficiency and higher packet throughput when running OVS.

**Hardware Accelerators**
Intel Smart Edge Open includes plug-ins to support Intel® FPGA Programmable Acceleration Card N3000 (Intel® FPGA PAC), High Density Deep Learning (HDDL) accelerator, and Visual Cloud Accelerator Card for Analytics (VCAC-A). These plugins allow a Kube-native environment in which Kubernetes manages the hardware accelerators along with other hardware resources.

**OpenVINO™ Toolkit**
Intel Smart Edge Open enables tight integration between hardware accelerators and the OpenVINO™ toolkit, allowing high-performance visual analytics and natural language processing on the ENSCONCE platform.

**Mobile Network Integration**
Intel Smart Edge Open provides reference implementations for certain 5G core network functions. Network Exposure Function (NEF) and Application Function (AF) help steer traffic from an end user to an edge application. Intel Smart Edge Open provides a Core Network Configuration Agent (CNCA) function to interact with the NEF and AF. Capgemini Engineering works with mobile network operators to integrate these reference implementations into their core network.

## Performance Optimization Case Study

Edge-edge applications can vary from use-case to use-case; however, they can be represented by a simplified model: transfer data, process data, and return a response or forward the data to a different service. The crucial factors of this use case are:

1. The delay of the network path between the device and the edge platform.
2. The delay caused by traffic routing within the edge platform.
3. The delay caused by processing the data in the edge application.
4. The delay caused by other system loads in the edge platform caused by multiple applications competing for hardware resources. Because an edge platform must provide high compute density to be economical, a heavy system load is to be expected during normal operation. Applications that are sensitive to latency and jitter will be negatively impacted during heavy system loading without appropriate solutions.

Factor 1 is determined by the access network architecture and will not be covered in this analysis. Factors 2–4 are influenced by Capgemini ENSCONCE platform architecture, and the performance impact will be analyzed.

### Key Metrics for Observation

**Round Trip Time (RTT)**
RTT (in milliseconds) refers to the average time for a message from the client device to travel to the edge platform and back. The test uses a simple echo application so that data processing time is negligible.

**Throughput**
Throughput (in Mbps) refers to the rate at which data passes through the system from client to server. Because data processing time is negligible in this test, throughput measures the end-to-end performance of the system.

### System under Test

We evaluate the performance of the ENSCONCE platform using an edge cluster deployment as described below and depicted in Figure 2 and Table 1.

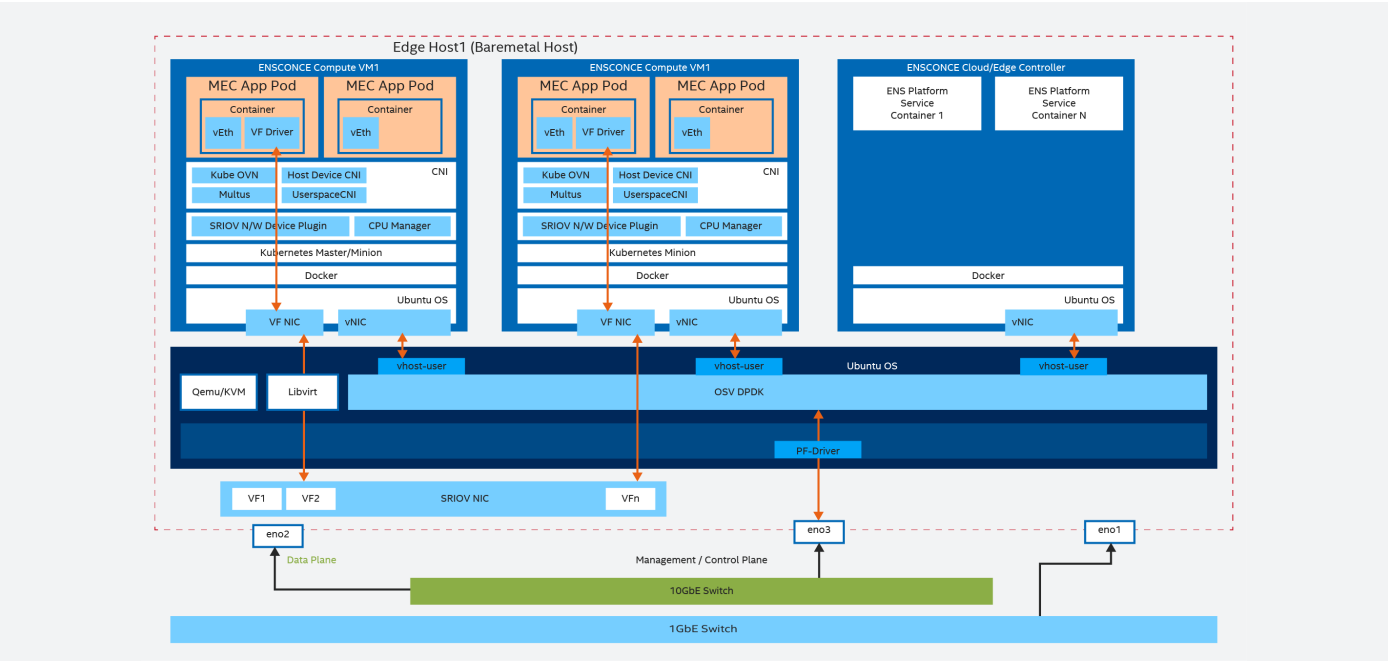


**Figure 2.** Reference deployment of ENSCONCE Platform as the System Under Test

| Hardware | Software |
|---|---|
| Platform:<br>2nd Gen Intel® Xeon® Scalable Processor Family Processors<br>2 x Intel® Xeon® Gold 6230 Processor<br><br>(24 cores, 48 threads, >=2.1 GHz per CPU, microcode: 0x5003003) | Host OS<br><br>Ubuntu 18.04 (kernel 4.15.0)<br><br>OpenStack (Rocky)<br><br>Docker 18.06.3-ce<br><br>Kubernetes v1.18 |
| Memory<br>192GB | ENSCONCE Platform 5.6.2.0<br><br>OpenNESS 20.06<br><br>Intel CPU Manager for Kubernetes v1.4.1<br><br>DPDK 19.11<br><br>OVS-DPDK v2.13.0 |
| Networking:<br><br>2 x Intel® Ethernet Network Adapter X710 | Multus CNI v3.3<br><br>SR-IOV CNI v2.0.0<br><br>SR-IOV plugin v3.1<br><br>Userspace CNI v1.2<br><br>HostDevice CNI 0.8.6<br><br>KubeOVN CNI 1.2.1 |
| Local Storage<br><br>1 x 1TB SSD SATA or Equivalent Boot Drive. | |

**Table 1.** Reference Hardware and Software Configuration for ENSCONCE Platform System Under Test

The ENSCONCE Platform is deployed on an OpenStack VM running Ubuntu, configured with:

- 8 virtual cores, with 0-4 cores reserved for CPU pinning
- 16 GB DRAM
- ENSCONCE Platform 5.6, incorporating OpenNESS 20.06
- Ubuntu VMs on servers running Intel® Xeon® Gold processors.

The edge client connects to the edge application running on the ENSCONCE Platform through a connected edge router.

### Network Configurations

Delays caused by packet routing within the edge platform (Factor 2) can be improved via multiple approaches. The following network variations were tested in this analysis:

- OVS with kernel mode
- OVS with DPDK mode
- SR-IOV

Delays caused by application processing (Factor 3) vary widely depending on the application. In this analysis, we used a simple application that echoes the input to the output.

Delays caused by other applications on the platform competing for resources (Factor 4) can be tested by pinning the test application to sequestered cores, and by varying the number and compute intensity of "noisy neighbor" applications. Noisy neighbor workloads are generated via the Linux stress tool, which loads all available cores by running multiple threads doing sqrt() and malloc() operations.

### Test Applications

iPerf server and client applications, L3-Forwarder, and Packet Generator were used to measure performance. Tests were conducted over a span of five minutes, with multiple iterations of iPerf performing TCP upload and download tests with 128 kB and larger packet sizes.

These test applications were provided the following resources:

- 1 vCPU core
- 1024 MB RAM

## Test Scenario

We evaluated platform performance of traffic representing a client connecting to an edge application. This scenario represents common dataflows occurring on an edge platform.

The following configurations were tested to measure impact on delays caused by packet processing within the platform and delays caused by noisy neighbor applications:

1. Kernel mode OVS Layer-2 Switch as the edge networking layer
2. Core pinning via Intel CPU Manager for Kubernetes
3. Kernel mode OVS with DPDK/OVS and using Kube-OVN plugin.
4. Kernel mode OVS with SR-IOV

### Base Configurations

**"Quiet"** consists of an ENSCONCE Platform, configured with a vanilla OpenStack VM based edge deployment, with the VMs connected through kernel mode OVS bridging. iPerf-client traffic is sent from an external-client to an iPerf-server edge application.

**"Noisy"** consists of the "Quiet" configuration with additional workloads competing for CPU, memory, and network resources in a noisy neighbor scenario.

### Optimizations on Base Configuration

**"Noisy" + Core Pinning** modifies "Noisy" by adding exclusive cores to the edge application using the CMK Scheduler built into Intel Smart Edge Open. This scenario is implemented with extra CPU workloads generating a noisy neighbor scenario.

**"Noisy" + SR-IOV & Core Pinning** augments Core Pinning by adding SR-IOV Virtual Functions (VFs) to the edge application using the Intel Smart Edge Open SR-IOV Plugin and CNI. This scenario is also implemented with extra CPU workloads generating a 'noisy neighbor' scenario.

These different configurations were benchmarked with the iPerf traffic test and relative performance was measured.

| Test Scenario | Test Configuration |
|---|---|
| "Quiet" | • KubeOVN with kernel mode OVS<br>• Without noisy neighbour |
| "Noisy" | • KubeOVN with kernel mode OVS<br>• With noisy neighbour |
| "Noisy" + Core Pinning | • KubeOVN with kernel mode OVS<br>• With CPU pinning<br>• With noisy neighbour |
| "Noisy" + SR-IOV & Core Pinning | • SR-IOV based interface<br>• With CPU pinning<br>• With noisy neighbour |

**Table 2.** Test Scenarios

## Benchmarking Observations and Analysis

### Throughput

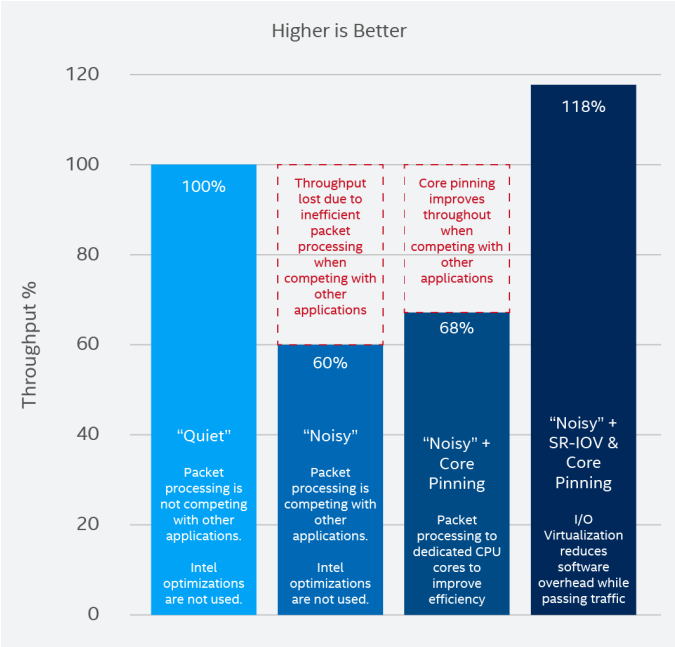


**Figure 3.** Impact of Intel optimizations on Throughput

The "Quiet" configuration with no competing application running is normalized to 100%.

When competing applications are introduced, "Noisy" throughput drops to about 60% of baseline as competition for resources limits packet processing performance.

When core pinning is introduced via Intel CMK, throughput increases to 68% of the baseline. This is due to efficiencies gained by allowing packet processing data to remain in the cache of the pinned core, rather than being flushed when other applications try to use the same core.

When SR-IOV and core pinning are used together, throughput increases to 118% of the baseline. By combining core pinning with the I/O virtualization options, packet processing efficiency in the presence of noisy neighbors is even better than an unoptimized system with any other applications competing for resources.

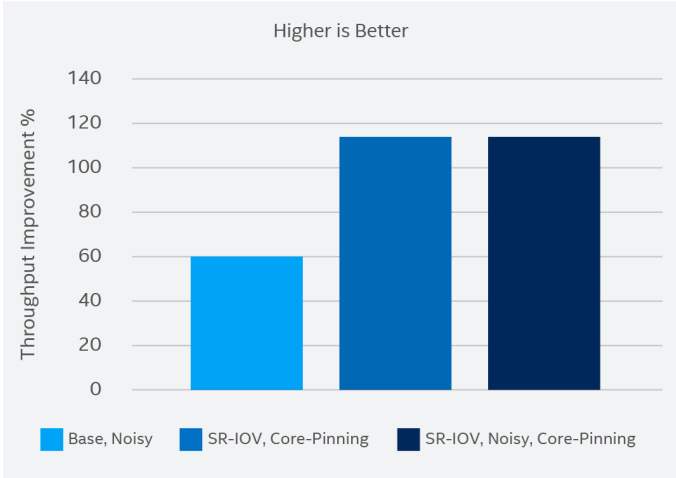


**Figure 4.** Throughput improvement as a percentage value of baseline

Further testing revealed that this test's throughput was limited by the 10 Gb network interface. When running the SR-IOV and core pinning scenario both with and without noisy neighbors, throughput was the same, indicating that with the Intel optimizations packet processing throughput is not impacted by other applications on the platform.

### Round Trip Time

In Figure 5, Round Trip Time (RTT) for "Quiet" is normalized to 100%.

When competing applications are added, "Noisy" RTT increases to 160% of baseline as packet processing is negatively impacted by competition for resources.

Core pinning reduces the impact on RTT somewhat, reducing RTT to 120% of baseline.

Adding SR-IOV combined with core pinning reduces RTT to 98-99% of "Quiet" despite having other applications competing for platform resources.

Not only does the combination of core pinning and SR-IOV reduce average RTT, it also reduces RTT variability as shown in Figure 6.

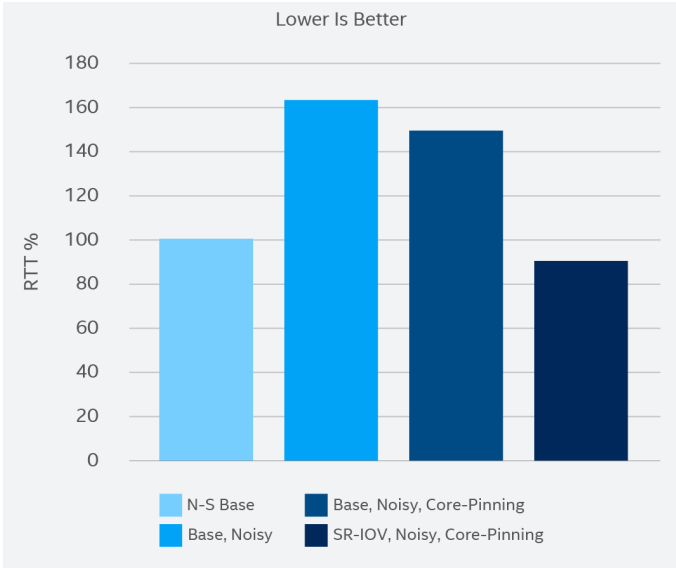


**Figure 5.** Packet RTT (Round Trip Time) as a percentage value of baseline

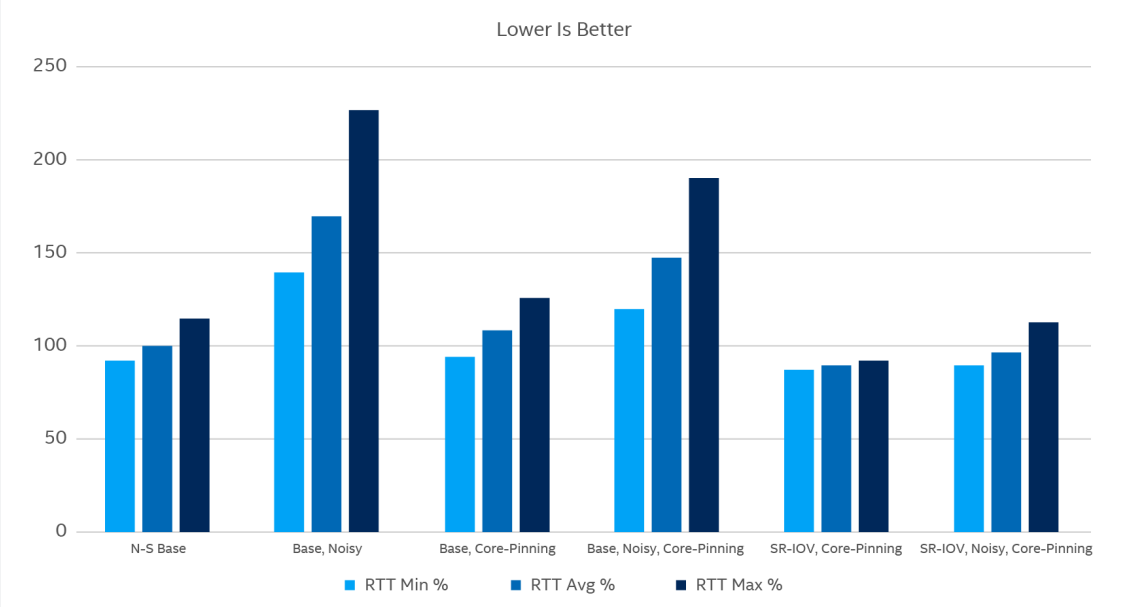


**Figure 6.** Minimum, mean, and maximum RTT across configurations

## Conclusions

Intel Smart Edge Open performance optimizations, including core pinning and SR-IOV, significantly boost performance for edge applications, especially when the edge platform is heavily loaded. By integrating these optimizations, Capgemini ENSCONCE platform delivers more deterministic behavior, with better throughput and lower RTT, to help edge applications meet expected QoS levels even when heavily loaded with other applications. This combination enhances the ability to deploy and manage applications in an edge network, and to optimize the performance of those applications for emerging technology solutions.



Printed in USA 347107-001 EN ♻ Please Recycle 0621/FZ/ICKK/PDF