



Host Power Management in VMware vSphere 7.0

Performance Study for Optimal Power Consumption

September 28, 2021



VMware, Inc. 3401 Hillview Avenue Palo Alto CA 94304 USA Tel 877-486-9273 Fax 650-427-5001 www.vmware.com

Copyright © 2021 VMware, Inc. All rights reserved. This product is protected by U.S. and international copyright and intellectual property laws. VMware products are covered by one or more patents listed at <http://www.vmware.com/go/patents>. VMware is a registered trademark or trademark of VMware, Inc. in the United States and/or other jurisdictions. All other marks and names mentioned herein may be trademarks of their respective companies.

Table of Contents

Executive Summary	3
Introduction	3
Background	4
C-States (Idle States/Power States)	4
P-State (Operational States/Performance State); Demand-Based Switching (DBS)	4
Power Management Settings	5
BIOS Settings	6
Host Power Management in vSphere 7.0	6
Defining Custom Power Policy in vSphere	8
Summary: vSphere Power Policies	8
Experimental Setup	9
Hardware	9
vSphere Power Management and esxtop	10
Results	12
Impact of P-States and C-States on Host Power Consumption	12
Observation 1: Only P-State Management is Enabled	13
Observation 2: Both P-State and C-State Management are Enabled	13
LoadGen	14
SPECpower	15
VMmark	17
View Planner (VDI)	20
Impact of vSphere Power Policies on Turbo Frequencies	23
Best Practices	25
References	26
About the Authors	27
Acknowledgments	27
Appendix A	28

Executive Summary

Modern datacenters, which are vital to the computing infrastructure, are expensive to build and operate. A significant part of the operational cost of a modern data center can be attributed to power consumption and cooling. In an ideal world, we would want to minimize power consumption without any impact on application performance. However, practical cases require a compromise. Finding the right balance entails considerable financial gains.

VMware vSphere® (ESXi) provides several pre-configured policies for power management, targeting applications to optimize for latency, throughput, and power consumption. Prior technical papers have described the available power policies in vSphere, their functionality, and present performance and power consumption data [1] [2]. This technical paper provides an updated guide to vSphere 7.0 users on different power policies' performance and power tradeoffs. In addition, the article uses experimental results to gain insights regarding the power management capabilities of VMware vSphere on the latest processors using modern workloads. Note that there are no algorithmic changes in the implementation of power management in vSphere 7.0. Instead, the article evaluates the vSphere power management on the latest processors with several recent workloads.

While observing the performance and power consumption on a range of workloads, the paper recommends the use of power policies as follows: (a) The “**Balanced**” policy is the default recommended policy in vSphere, because it maximizes performance-per-watt overall and provides an optimal point between performance and power consumption for a wide variety of application characteristics. (b) The “**High Performance**” profile is preferred in the case of latency-sensitive applications at the cost of higher power consumption, and (c) “**Low Power**” profile for low utilization servers, resulting in higher power savings at the cost of performance.

Introduction

The power used by servers and data centers accounts for ~1.5% of the total power consumption in the world [3]. CPUs in modern data centers consume ~30% of the overall system power. As CPU hardware complexity has grown significantly in recent years, computer architects have introduced several power-focused enhancements. These designs have enabled operating systems to capitalize on architectural tools to improve performance-per-watt on various applications. The introduction of demand-based switching (DBS) from Intel has been the critical factor behind power savings in modern processors.

The Advanced Configuration and Power Interface (ACPI) specification is an open standard initially developed by several hardware vendors and software developers [4]. It establishes standard interfaces that enable operating system-directed motherboard device configuration and power management. ACPI can apply to both individual hardware components and the entire system. In addition, it helps monitor the system's status and employs power management algorithms by changing the CPU operating frequency and putting unused components to sleep [5].

Background

The ACPI standard defines C-States (commonly known as idle states or power states) and P-States (widely known as operational states or performance states). Hardware vendors such as Intel and AMD have introduced hardware support for P-States and C-States, which are accessible through the on-chip power control unit (PCU) [6] [7].

C-States (Idle States/Power States)

C-States are power states that aid in saving power by turning off sub-sections of the CPU when not in use. CPUs are designed to support various power levels. C0-state is the operational state where all the components are active, and the processor can actively execute instructions. C1 is a shallow state where the clock is gated (switched off). However, all the modules remain active, and the processor can go back to the active C0 state instantaneously.

Furthermore, C2-Cn are sleep states where specific sections of the CPU are turned off. The higher the C-State, the deeper into sleep mode the CPU goes. Thus, higher C-States result in significant power savings. However, it takes more time for the CPU to return to the operational state from deeper sleep states. Therefore, changing the C-state in the BIOS setting doesn't have an impact on throughput. However, the latency to get back from a deeper sleep state would be higher.

P-State (Operational States/Performance State): Demand-Based Switching (DBS)

P-States correspond to different performance levels that are applied while the processor is actively executing instructions. P-States are relevant only when the processor is in the active C0-state. P-State is both a frequency and voltage operating point defined as performance states in the ACPI specification. Both frequency and voltage are scaled as the P-State increases. This process is referred to as dynamic voltage and frequency scaling (DVFS). Hardware vendors such as Intel incorporate several hardware-level P-States that divide the energy and frequency demands into several tiers. P0 is the highest frequency (with the highest voltage). It is often referred to as the 'turbo mode'. P1 is the nominal/base operating frequency. Higher P-States correspond to lower operating frequencies. P-States play a crucial role in saving CPU power when the workload does not fully load a CPU.

Figure 1 shows how P-States and C-States relate to each other. The P-States and C-States form the basis for power management in modern processors. The number of available P-States and C-States vary with vendors and processor generations. VMware vSphere utilizes the available P- and C-States in the power management algorithms to provide easy-to-use power profiles for end-users.

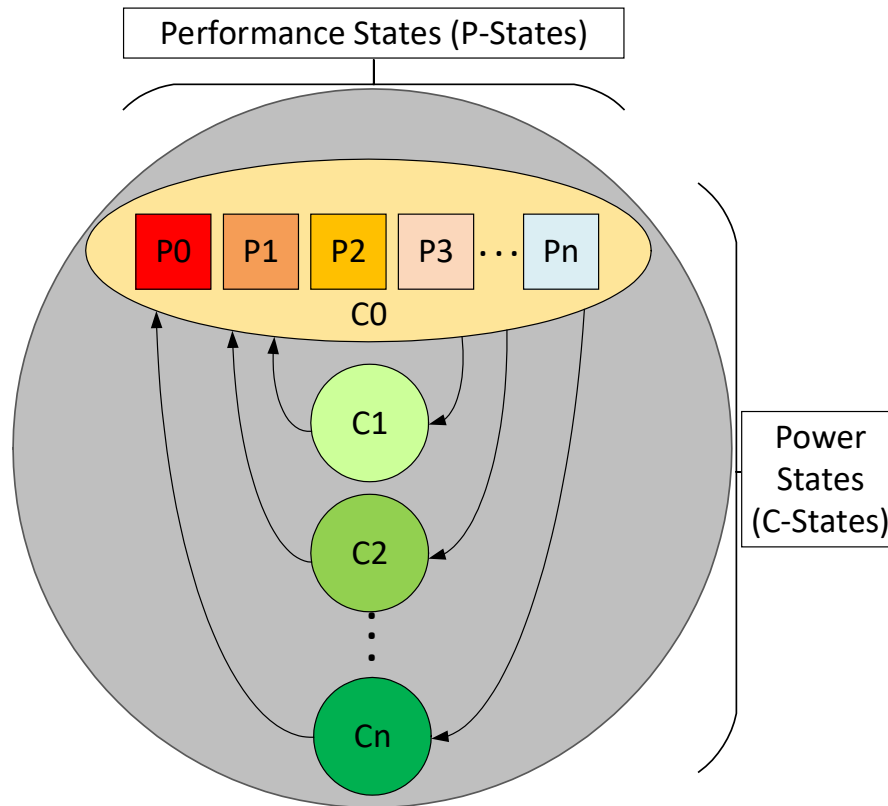


Figure 1: Representation of ACPI P-States and C-States

Power Management Settings

VMware vSphere includes a comprehensive set of power management capabilities. These are designed to save power during idle times and when the host is underutilized or is inactive. The user is recommended to configure the BIOS settings to **OS control mode** or the equivalent to allow vSphere the most flexibility to use available hardware features for optimal power consumption.

An example of how to select the BIOS settings to allow vSphere to control power management in a *Dell PowerEdge* system is shown below in [Figure 2](#). Similar options can be found in servers from other vendors as well.

BIOS Settings

To set the recommended BIOS settings, follow these steps before deployment.

1. Enter the System BIOS settings by pressing **F2** during bootup.
2. Select **System Setup > System Profile Settings**.
3. The default System Profile on Dell systems is **Performance Per Watt (DAPC)**. Change this to **Performance Per Watt (OS)** to transfer control to vSphere/ESXi.
4. All other settings are selected by default.

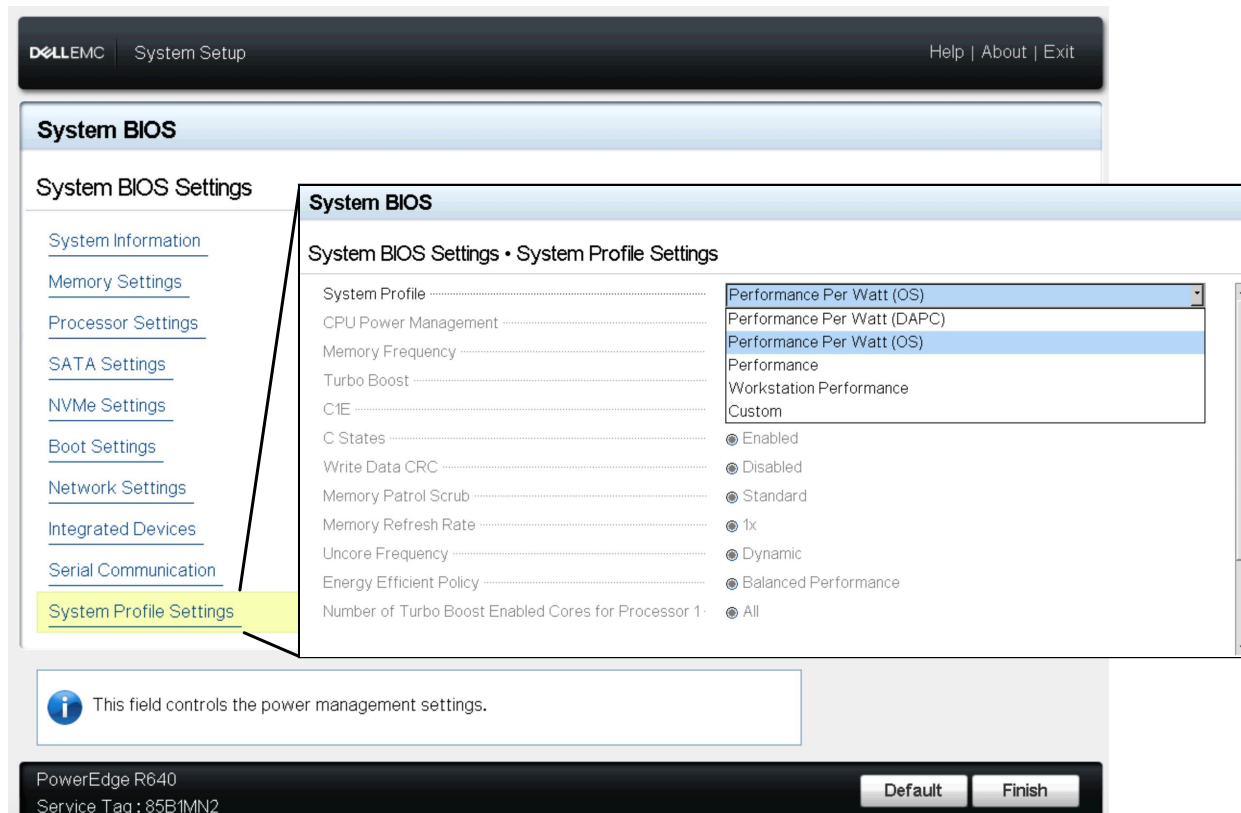


Figure 2: Sample BIOS setting on a Dell PowerEdge R640 host

Host Power Management in vSphere 7.0

vSphere 7.0 offers four different power management policies that utilize the ACPI P-States (Performance States) and ACPI C-States (Power States).

The available power policies are as follows:

- **High Performance:** This policy tries to maximize performance by disabling C-State and P-State management. It always keeps the CPU in the highest possible operating frequency (P0-State) and only uses the top two shallow C-States (C0 when running and C1 when idle). The High-Performance policy is geared toward latency-sensitive applications and provides predictable and consistent performance.

- **Balanced:** This policy is designed to minimize host power consumption while having little to no impact on performance. The balanced policy is the default power policy in vSphere 7.0. The policy determines the system load and selects the appropriate P-State. The Demand Based Switching (DBS) implementation here has little to no impact on application performance. vSphere chooses a suitable deep C-State (like C2) based on its prediction of when the CPU cores need to be active again.
- **Low Power:** This policy is designed to reduce power consumption compared to the other policies substantially. The P-State and C-State selection algorithms are more aggressive toward power saving. This policy, however, can impact the performance of latency-sensitive applications.
- **Custom:** This policy, by default, has the same settings as Balanced. However, it provides the user the ability to tune individual parameters for specific use cases.

Steps to change the power policies using the vSphere Web Client:

1. Select the host from the inventory and click the **Manage** tab and then select **Settings**.
2. In the left pane under **Hardware**, select **Power Management**.
3. Click **Change Policy** and select the policy of interest as shown in [Figure 3](#).

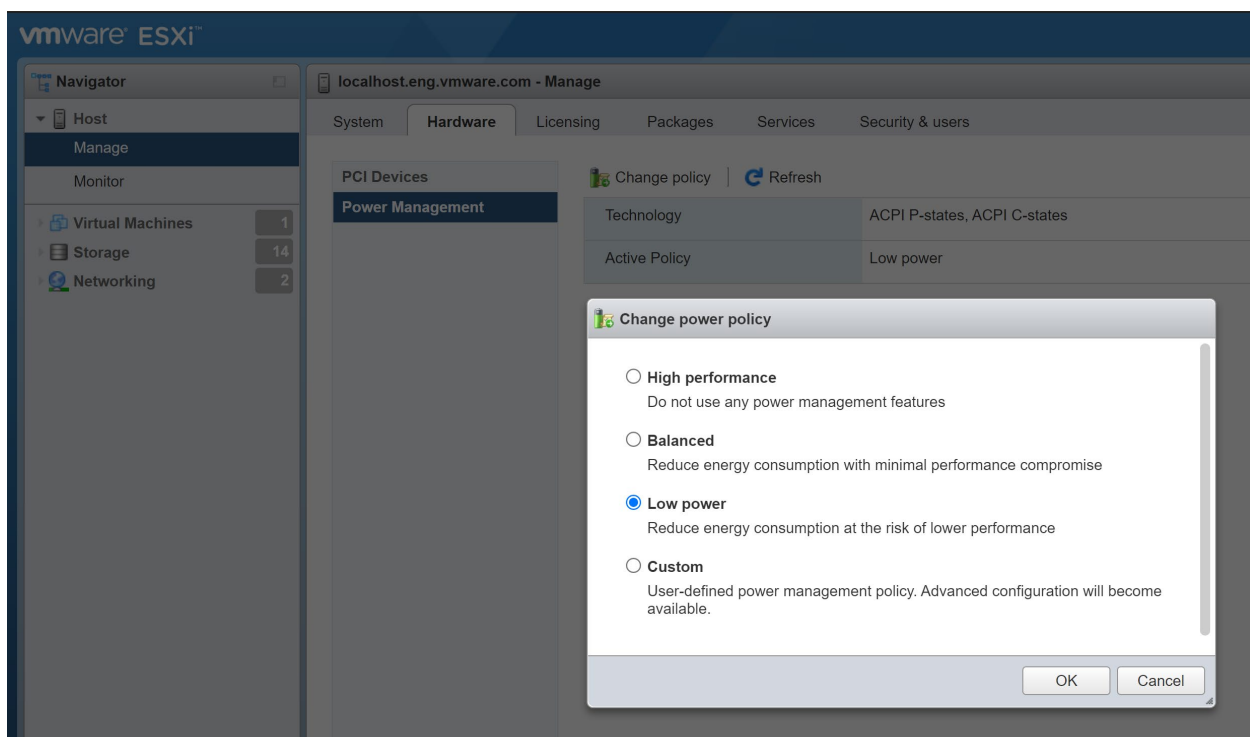
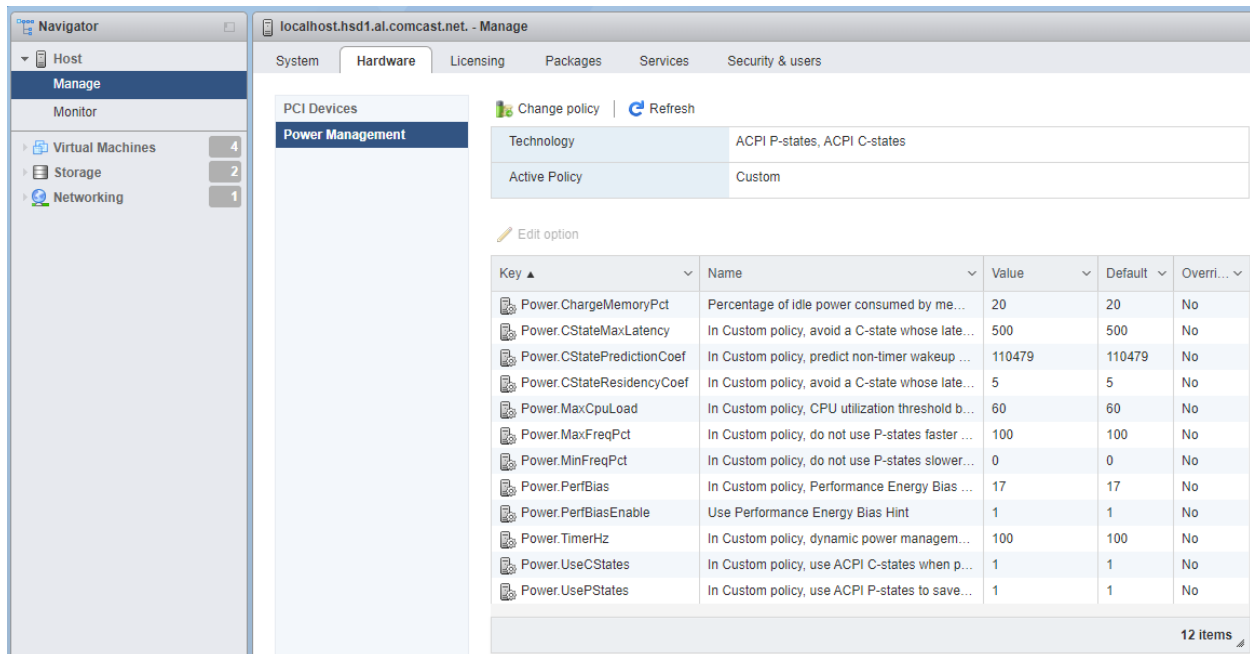


Figure 3: vSphere UI for power policy selection

Defining Custom Power Policy in vSphere

When the Custom policy is selected, vSphere provides various options to the user, as shown in Figure 4. Note that changing parameters in the Custom policy without a comprehensive understanding of the options may negatively impact performance and power consumption.



Key	Name	Value	Default	Overri...
Power.ChargeMemoryPct	Percentage of idle power consumed by me...	20	20	No
Power.CStateMaxLatency	In Custom policy, avoid a C-state whose late...	500	500	No
Power.CStatePredictionCoef	In Custom policy, predict non-timer wakeup ...	110479	110479	No
Power.CStateResidencyCoef	In Custom policy, avoid a C-state whose late...	5	5	No
Power.MaxCpuLoad	In Custom policy, CPU utilization threshold b...	60	60	No
Power.MaxFreqPct	In Custom policy, do not use P-states faster ...	100	100	No
Power.MinFreqPct	In Custom policy, do not use P-states slower...	0	0	No
Power.PerfBias	In Custom policy, Performance Energy Bias ...	17	17	No
Power.PerfBiasEnable	Use Performance Energy Bias Hint	1	1	No
Power.TimerHz	In Custom policy, dynamic power managem...	100	100	No
Power.UseCStates	In Custom policy, use ACPI C-states when p...	1	1	No
Power.UsePStates	In Custom policy, use ACPI P-states to save...	1	1	No

Figure 4: Custom policy selection and available options

Power management parameters that affect the custom policy have descriptions that begin with the phrase, **In custom policy**. All other power parameters affect all power management policies. Table 1A in Appendix A describes these parameters in detail.

Summary: vSphere Power Policies

Table 1 summarizes the use of P-States and C-States in each of the power policies in vSphere. To recap, the **High-Performance** policy always requests the highest operating frequency (P0) in hardware and does not use deep sleep states (C2). It is designed for latency-sensitive workloads. **Low Power** policy has P-State, and C-State management focuses on power savings; this generally comes at a performance cost. Finally, the **Balanced** policy is designed to be an optimal middle ground for various applications.

	High Performance	Balanced	Low Power
P-States	P0	P0-Pn	P0-Pn
C-States	C0 & C1	C0, C1, & C2	C0, C1, & C2

Table 1: P-State and C-State usage in vSphere policies

Experimental Setup

In this section, we describe the hardware and software setup used for evaluating the power management features of vSphere.

Hardware

Table 2 shows the test machine configurations used for this paper. The test system is a *Dell Power Edge R640* server housing an Intel Xeon Platinum 8260 from the Cascade Lake architecture. The System Profile in the BIOS settings was changed to **Performance Per Watt (OS)**, as shown in [Figure 2](#). All other BIOS options are left at their default values.

CPU	Processor	Intel Xeon Platinum 8260
	Codename	Cascade Lake
	# Sockets	2
	Physical Cores	24 * 2
	Logical Core	48 * 2
	Thermal Design Power (TDP)	165 W
CPU Frequency Information	Max. Turbo Frequency	3.90 GHz
	CPU Nominal/Base Frequency	2.40 GHz
	CPU Minimum Frequency	1.00 GHz
System Memory	DRAM on Machine	768 GB
	# Mem. Channels Per Socket	6

Table 2: Dell PowerEdge R640 test machine specifications [8]

vSphere Power Management and esxtop

Figure 5 shows the esxtop power screen (obtained by executing `esxtop` and pressing `p`) for the vSphere host. Please consider the following points while observing the output.

- **Power Usage:** The total host power reading is obtained from the internal sensors embedded in the host's power supply unit through the Intelligent Platform Management Interface (IPMI) driver. This reading includes CPU Power, Memory Power, and Other Power (PCIe devices, cooling fans, etc.). If a particular machine does not have a built-in sensor or vSphere cannot read the values, '0' is displayed.
- **P-State MHz:** Provides the available P-States and the respective frequency of operation in each P-States on the Intel Xeon Platinum 8260. This metric is specific to a particular CPU and can vary among different CPU families and vendors.
- **CPU:** The test system has 96 logical Cores (hyper-threads). Each row gives the power statistics for each of the logical cores.
- **C-State Selection:** The available C-States are shown as different columns. The available C-states are numbered consecutively without gaps (C2, in this case, corresponds to Intel's C6 deep C-state). Each row shows the percentage of time the logical CPU votes to stay in that particular C-State.
- **P-State Selection:** The screenshot shows 16 different P-States, P0 to P15. P0 has the highest clock frequency (2.401 GHz), and P15 has the lowest clock frequency (1.00 GHz). Each logical core on the physical core can vote for a particular P-State (frequency), but the P-State assignment happens at the physical core level. Thus, the lower P-State vote between the two requests is selected. On the other hand, if a processor has a socket-level P-State granularity, then the chosen operating P-State would be the lowest vote across all cores on the socket.
- **%Aperf/Mperf:** This column shows the ratio of Aperf to Mperf, which indicates the actual frequency at which the core is running. A value of 100 means the core is running at nominal/base frequency; anything higher indicates running in turbo and anything lower suggests the core is running at lower frequencies. Note that the %Aperf/Mperf counter is updated only when the core is in the running state (C0-state). So, the counter value is not valid when the core spends time in C1 or C2 idle sleep states.

```
Power Usage: 175W, Power Cap: N/A
PSTATE MHZ: 2401 2400 2300 2200 2100 2000 1900 1800 1700 1600 1500 1400 1300 1200 1100 1000
```

CPU	%USED	%UTIL	%C0	%C1	%C2	%P0	%P1	%P2	%P3	%P4	%P5	%P6	%P7	%P8	%P9	%P10	%P11	%P12	%P13	%P14	%P15	%A/MPERF	
0	0.1	0.4	0	1	99	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	98	42.2
1	0.1	0.1	0	2	97	0	0	0	0	0	0	0	3	0	0	0	0	0	0	0	0	97	60.4
2	0.1	0.2	0	0	100	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	96	42.9
3	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
4	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
5	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
6	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
7	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.6
8	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	46.8
9	0.0	0.1	0	0	100	91	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	9	51.9
10	0.0	0.1	0	5	95	14	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	86	43.9
11	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.9
12	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
13	0.0	0.1	0	2	98	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.6
14	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
15	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
16	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
17	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
18	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
19	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
20	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
21	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.6
22	0.1	0.5	0	0	100	61	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	39	46.5
23	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	49.3
24	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
25	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
26	0.2	0.5	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
27	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
28	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.6
29	0.0	0.1	0	2	98	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	41.7
30	0.1	0.4	0	0	100	43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	57	45.2
31	0.0	0.1	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	46.4

Figure 5: Test machine esxtop view of P-States and C-States

Some vsi nodes can also be queried to see the core's actual time in deep C-States through the C-state residency counters. Note that this depends on the architectural supports provided by the vendor. For example, the following command can be used:

```
vsish -e get /power/pcpu/<pcpu_num>/cres
```

Figure 6 shows the expanded view of the available P-States, the equivalent operating frequency, and the socket TDP. Note that the difference between P0 and P1 is shown to be only 1 MHz. The main difference between P0 and P1 is that P0 enables turbo mode, whereas P1 does not. Turbo frequency is hardware controlled (invisible to vSphere) and is based on the load, number of active threads, power, and the thermal budget. The OS can only request P0, and the hardware determines the actual frequency of operation.

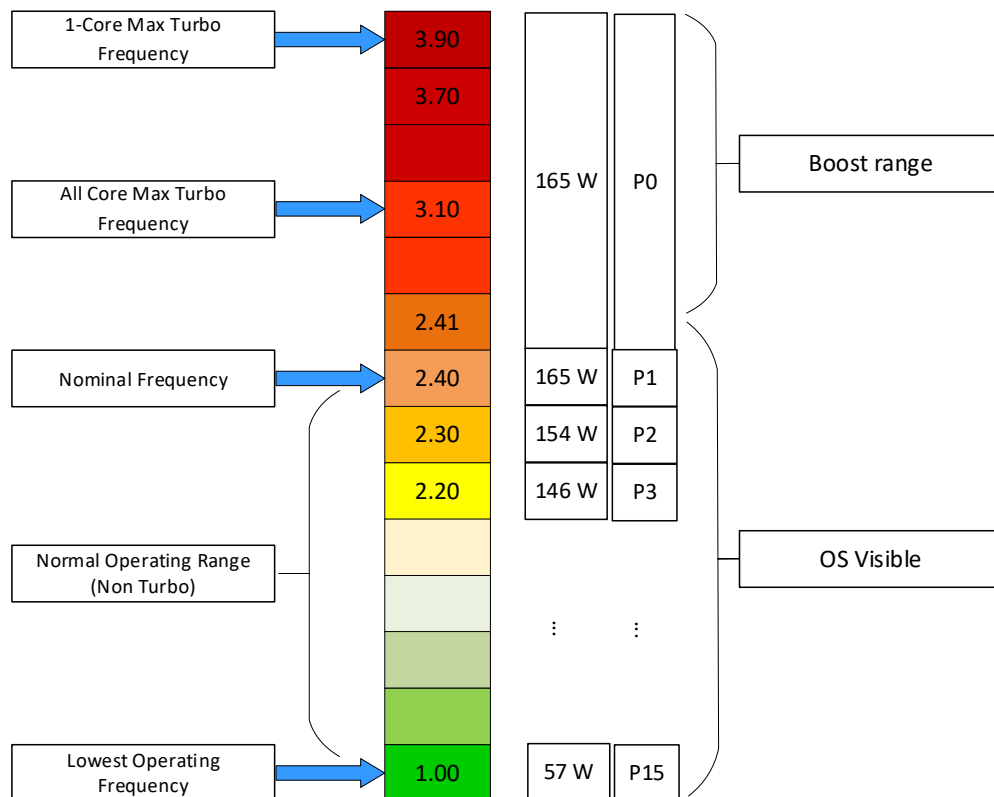


Figure 6: Intel Xeon Platinum 8260 operating states and equivalent power estimates

Results

This section walks through the experimental results of the test machine for several workloads. The study utilizes synthetic and three real-life workloads to demonstrate the impact of the vSphere power management policies on metrics such as latency, performance, and power consumption.

Impact of P-States and C-States on Host Power Consumption

It is vital to establish the impact of P-State and C-State management on total host power. This section shows the effect of enabling or disabling the P-State and C-State management on host power consumption at various load levels. We achieved this by selecting the Custom power policy and changing the parameters. We will show the performance impact in the next section. Figure 7 shows the Host Power (Y-Axis) sampled during different load levels (X-Axis) of SPECpower for the following cases:

- Power management disabled (P-States and C-States are disabled) – orange line
- Only P-state management enabled – blue line
- Both P-State and C-State management enabled – green line

When power management is disabled, both C-state and P-state management is disabled. The system will continuously operate in P0 while active and C1 when idle (C2 or higher is disabled). We use this as the baseline for the subsequent two cases.

Observation 1: Only P-State Management is Enabled

In this case, only P-state management is enabled. While active, the system can operate between P0 and P15, depending on the load. During idle times, the system will be in the C1 state (C2 is disabled). During high load levels (~100% load), the processor operates in P0, and consequently, we see that enabling P-state management saves only ~1% power. However, as the load decreases, enabling P-States provides an average power gain of ~6% (with a maximum of ~8%, which is the difference between the orange and blue lines). Note that enabling P-state management does not affect idle power consumption because it only uses C1 when idle.

Observation 2: Both P-State and C-State Management are Enabled

In this case, both C-State and P-State management are enabled. The system can operate between P0 to P15 while active and C1, and C2 are used during idle times. As the load reduces, we can observe a clear distinction between the orange and green lines. This is due to the system going into a deeper C2 state during idle times in low load levels. An average power gain of ~9% can be observed across load levels (with of maximum of ~13%). When the system is idle, ~13% gain can be observed when power management is enabled due to the use of the C2 state.

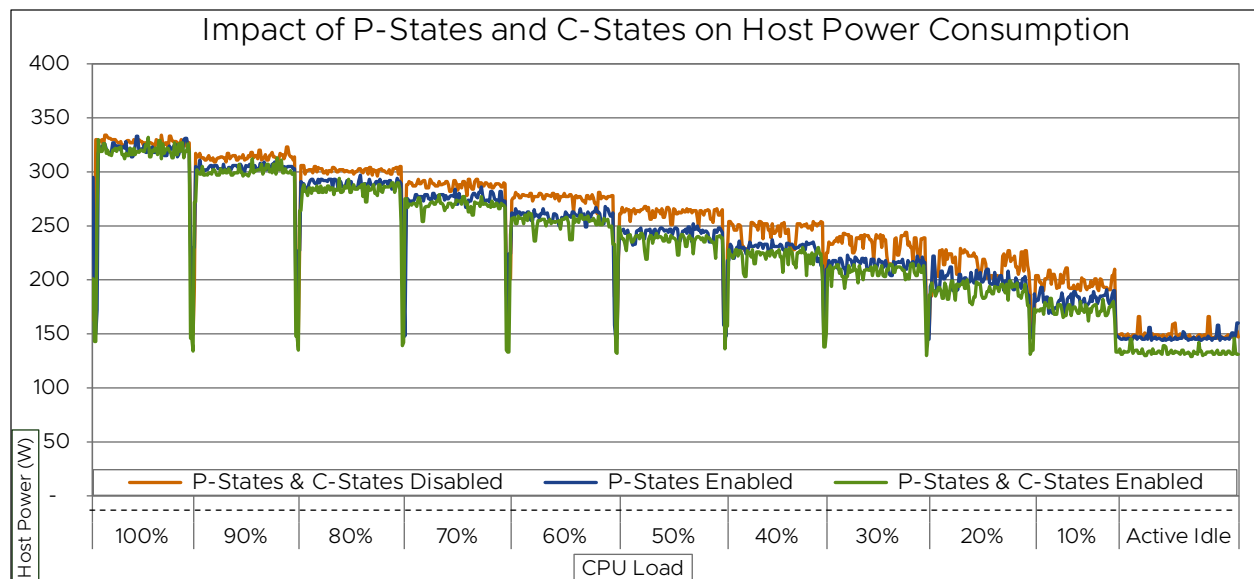


Figure 7: Impact of P-States & C-States on host power consumption

LoadGen

LoadGen is a user-world synthetic microbenchmark developed by VMware that can precisely control the CPU load and the number of active CPUs using pulse-width modulation. Thread affinity is built into the application. This micro-benchmark is a CPU-intensive application and does not stress memory. The objective here is to establish a baseline estimate of power consumption for various threads and intensity levels on the host.

Figure 8 shows the total host power consumed while executing the Load Gen microbenchmark at various threads and intensity levels. The X-Axis shows the number of threads, with ten intensity levels from 10% to 100% in each thread combination. The Y-Axis shows the total host power consumed. Again, the contrast between the three policies can be observed, indicating the impact of P-States and C-States. Note that Load Gen applies thread affinity, the first 48 threads are pinned to the CPUs in the first physical socket, and the subsequent threads are pinned to the cores in the second socket in that order.

We can draw a clear distinction between High Performance and Balanced/Low Power in the case of low thread count. This is precisely due to the use of the C2 state to save power on the idle cores. However, when the load increases, the power consumption in all three policies is comparable, signifying that the power budget is not hindered by policy selection. Another key difference is the increase in power consumption rate for the three policies as the intensity increases within each thread count. This is because each policy is tuned with different thresholds for P-State management. Finally, we can see that all three policies consume the same amount of power when the system is fully loaded.

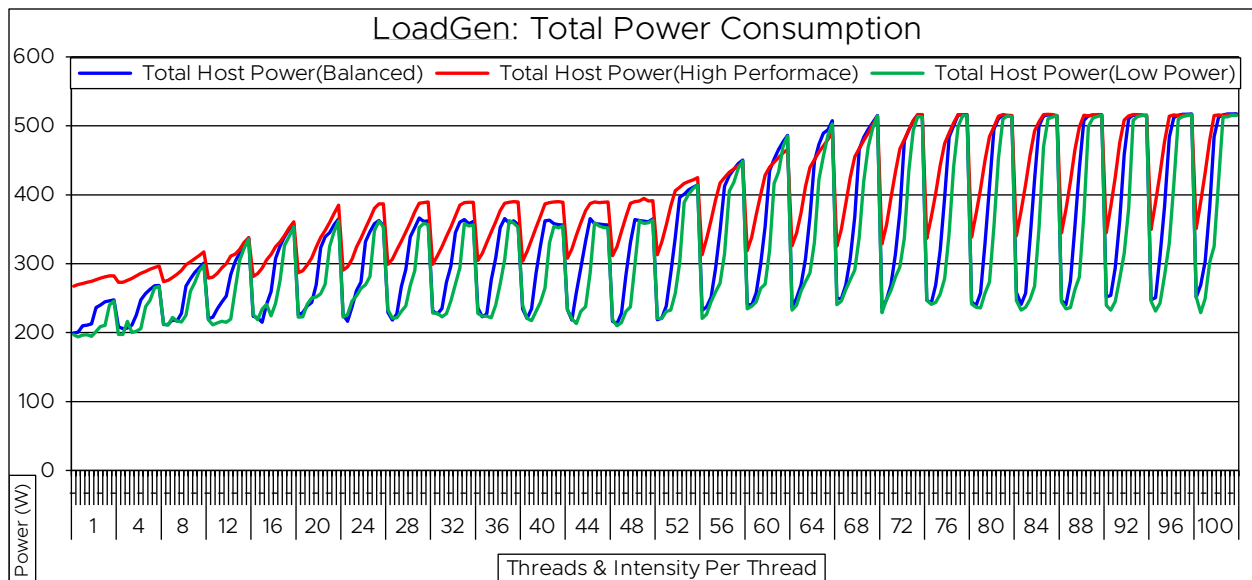


Figure 8: Host power consumption for various vSphere power policies for the LoadGen workload

SPECpower

The SPEC Power_ssj2008 was developed by The Standard Performance Evaluation Corporation (SPEC) to measure servers' power and performance characteristics and report the overall performance/watt metric [9]. SPEC power_ssj2008 (referred to hereafter as SPECpower) benchmarks are based on server-class Java workloads that are scalable (multithreaded), portable, and economical (in terms of time and money). The benchmark targets hardware vendors, IT providers, computer manufacturers, and governments [10].

The execution of the benchmark takes ~70 minutes. The benchmark goes through various phases of execution:

- **Calibration Phase:** The benchmark runs on the system with the maximum throughput possible, determined by running the workload unconstrained for at least three calibration levels. The maximum throughput is selected as the average of the throughput achieved during the final two calibration levels.
- **Execution Phase:** The workload is then run in a controlled manner, with delays inserted into the workload stream, to obtain total throughputs of 100%, 90%, 70%, 60%, 50%, 40%, 30%, 20%, and 10% of the maximum throughput calculated in the calibration phase. During each of these target loads, the power characteristics of the system under test (SUT) and the temperature are recorded.
- **Inference Phase:** Finally, the power characteristics and temperature are measured and recorded during an idle interval during which the SUT processes no Java transactions. When the benchmark completes, an efficiency score (performance per watt), maximum performance, and power consumption for each load level are generated.

We execute SPECpower on a RHEL 7.8 VM with 96 vCPUs and 764 GB vRAM while monitoring total host power consumption. Figure 9 shows the total host power consumed and the percentage power savings in different policies compared to High Performance. The primary Y-Axis on the graph shows the total host power consumed in Watts for each of the load levels on the system. The secondary Y-Axis represents the percentage savings in Balanced and Low Power policies compared to High Performance in each of those load levels.

We find that during higher load levels, the power consumption in all three policies is close. However, the difference in power consumption starts to manifest as the load decreases. Starting at 50% CPU load, the Balanced and Low Power policies start saving power compared to High Performance with a peak power savings of ~17% at 10% load and ~30% for the idle system.

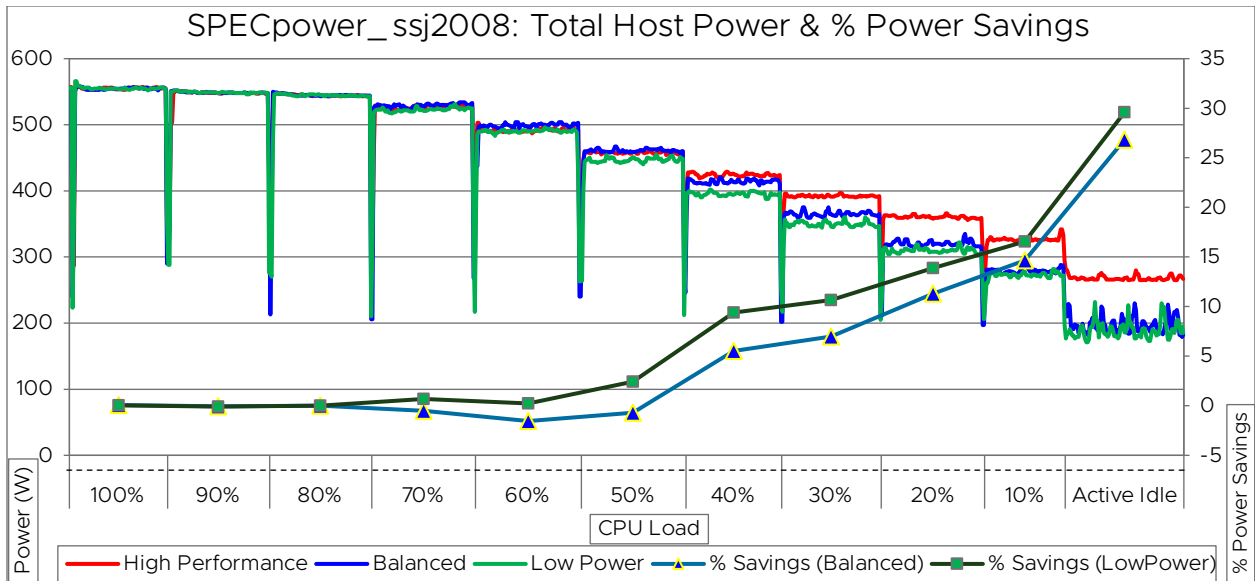


Figure 9: Host power consumption for various vSphere power policies during a SPECpower run

Figure 10 shows the SPECpower performance metric (ssj_ops) on the primary Y-Axis and the average host power consumed during each load level on the secondary Y-Axis. Note that SPECpower is a throughput-oriented workload, and all the three policies achieve similar performance. However, Balanced and Low Power policies could reduce the power consumption at lower load levels by using P-States and C-States.

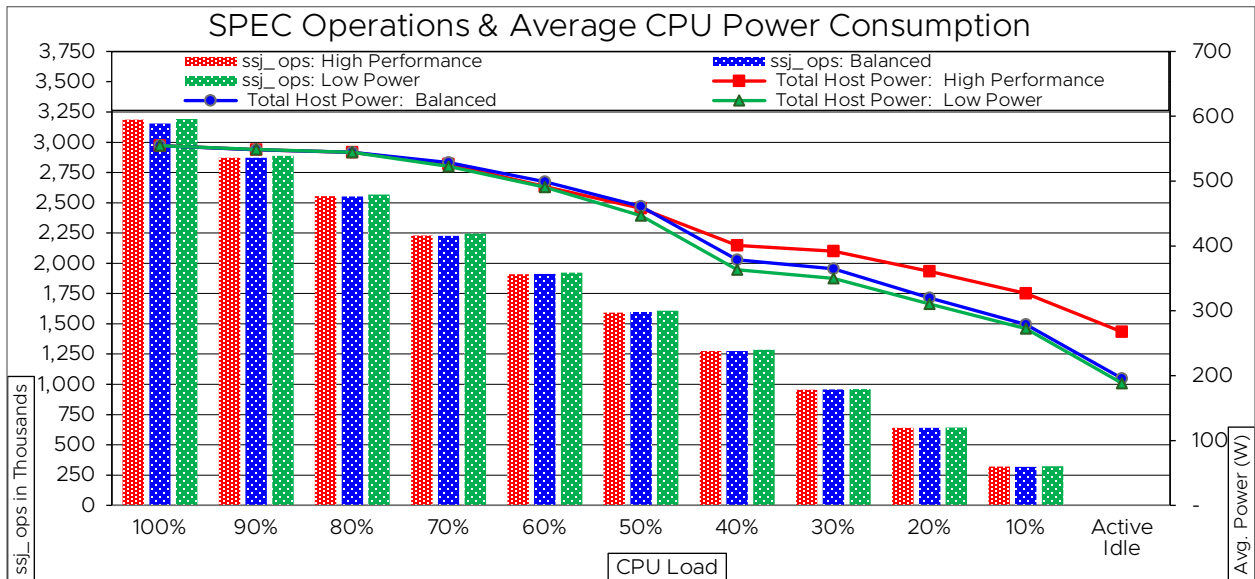


Figure 10: SPECpower performance & average host power consumption for vSphere power policies

Figure 11 shows the performance per watt for the SPECpower benchmark on the test machine. The higher performance per watt seen for Balanced and Low Power, especially for load levels less than 50%, is directly attributed to the extra power savings in those two policies.

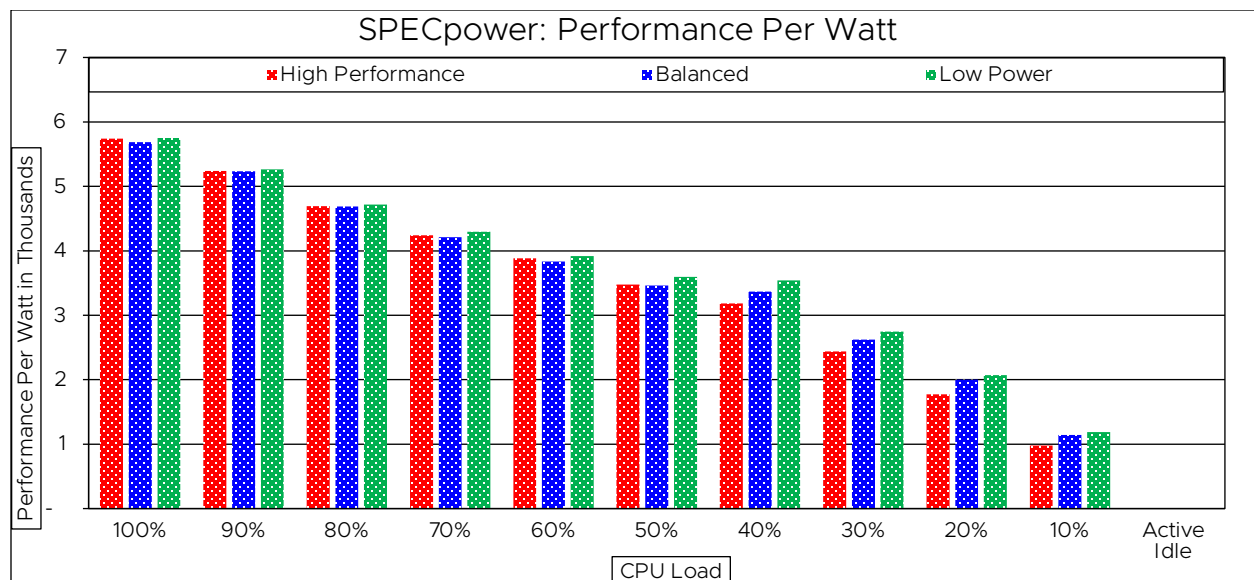


Figure 11: SPECpower performance per watt

Overall, we observe that using Balanced or Low Power policies for throughput-oriented applications like SPECpower can result in power savings with no loss in performance.

VMmark

A cloud environment typically comprises several diverse workloads on a virtualization platform — a collection of physical servers accessing shared storage and network resources. VMmark is an open-source benchmark suite from VMware used by hardware vendors and others to evaluate virtual platforms' performance, scalability, and power consumption [11] [12].

The VMmark benchmark suite uses a tiled approach. Each tile consists of 19 VMs requiring 47 vCPUs with ~190 GB of DRAM and ~800 GB of storage space.

VMmark workloads include:

- Scalable Web Simulation (Static & Elastic)
- E-Commerce Simulation
- Idle Server

We ran VMmark with 1-tile, 2-tiles, 3-tiles, and 4-tiles on the test systems. Figure 12 shows the VMmark score obtained for each of the runs. The secondary Y-Axis shows the PCPU percentage utilization time in each of the runs. The VMmark score depends on the latency observed in the different workloads. The High Performance policy can achieve lower latencies by not using a deeper C2 state and, consequently, a better overall score.

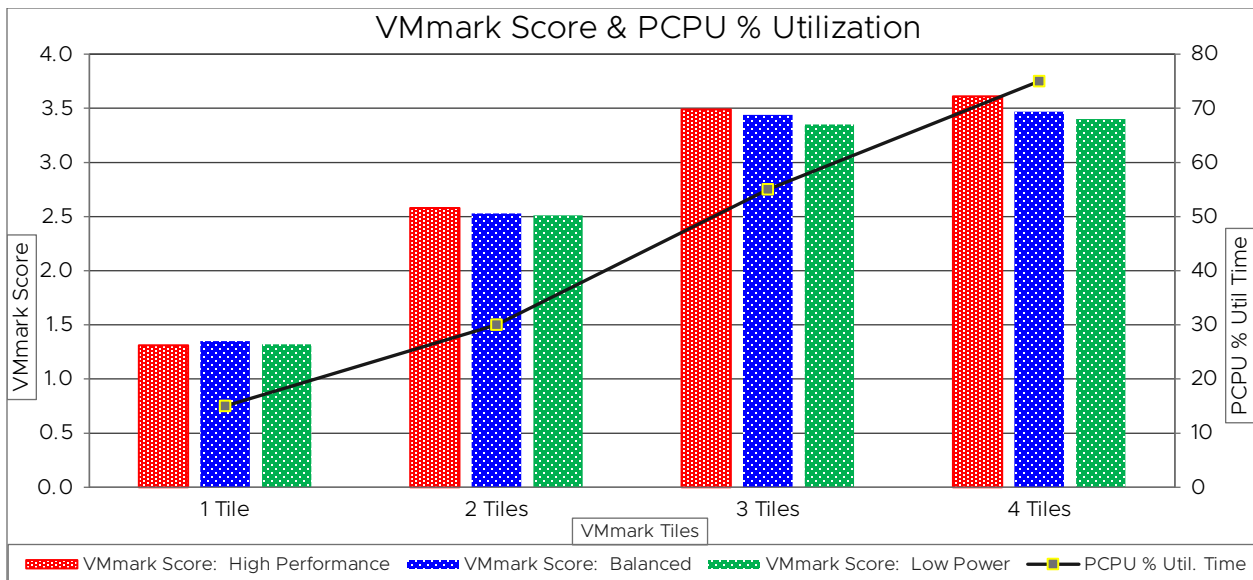


Figure 12: VMmark score and PCPU % utilization

Table 3 shows the individual average benchmark score and the overall VMmark score for a 4-tile execution. We notice that the web simulation benchmark (Weathervane) results are similar with no noticeable difference in performance. However, the e-commerce simulation benchmarks (DVD Store) show visible performance variations. For example, the High Performance profile has a performance improvement of ~5% over the Balanced profile and ~10% over the Low Power profile. As a result, the final VMmark score shows the expected behavior of the power policies, with High Performance providing the best performance result.

4-Tile Run	Weathervane Auction	Weathervane Elastic	DVDStoreA	DVDStoreB	DVDStoreC	VMmark Score
High Performance	3,549	565	671	419	282	3.61
Balanced	3,554	564	650	390	253	3.47
Low Power	3,558	562	611	374	250	3.4

Table 3: Benchmark Score for a 4-Tile VMmark Run

Table 4 shows the average host power consumed for three power policies during runs. Again, we observed very similar power consumption in all three policies. This is because even one tile of VMmark needs 47 vCPUs and can keep all the available physical cores (48) of the underlying system active.

VMmark Tiles	Avg. Power (W)		
	High Performance	Balanced	Low Power
1 Tile	366	360	362
2 Tiles	458	453	452
3 Tiles	530	530	528
4 Tiles	542	542	542

Table 4: Average host power for vSphere power policies for the VMmark run

Figure 13 provides a run-time power consumption view during the execution of a 2-tile VMmark run. The three different lines indicate the power consumed for the three different power policies. We observed that power consumption for all policies is very similar; however, we notice that the High Performance policy has higher peaks than Balanced and Low Power. We can also observe the difference in power consumption during ramp up or ramp down stages.

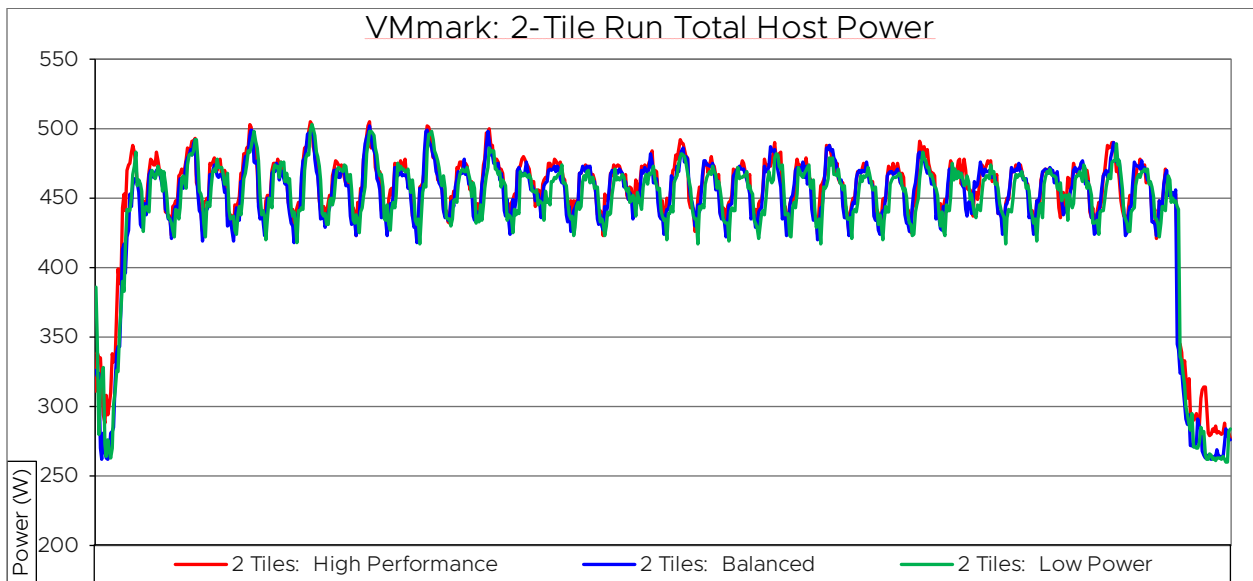


Figure 13: 2-Tile VMmark run for host power consumption for vSphere power policies

Figure 14 shows the host power consumption when running 1, 2, 3, and 4 tiles on the test system in the Balanced policy. We observed the machine reaches saturation at 4-Tiles as the host power consumption stays steady at its peak (550 W).

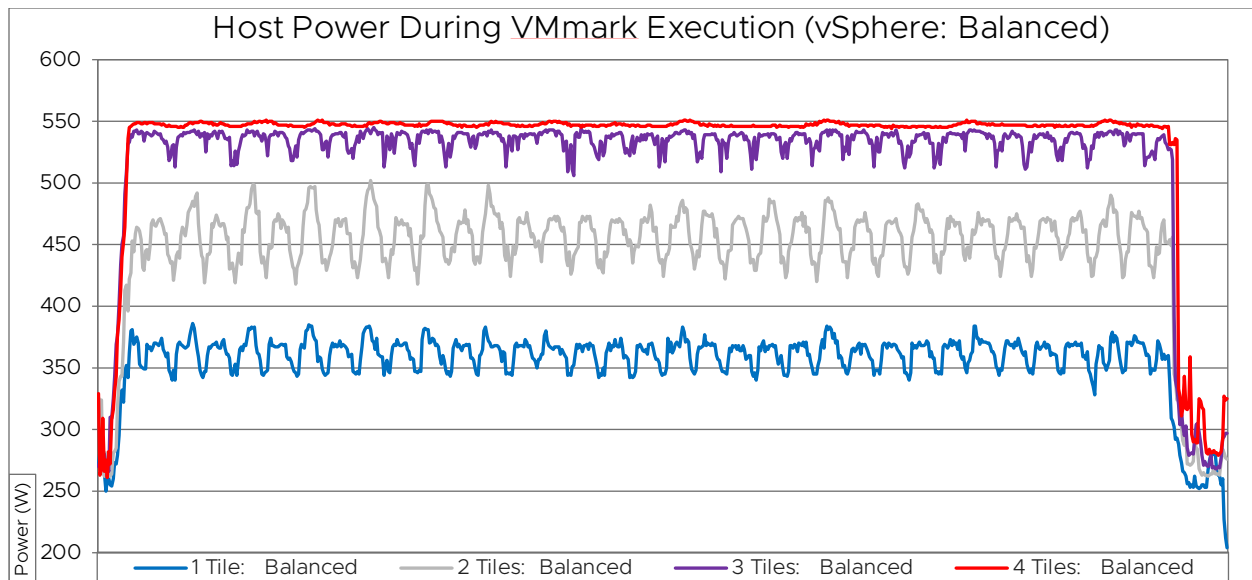


Figure 14: Host power consumption during VMmark tile execution on Balanced policy

View Planner (VDI)

VMware View Planner is a benchmark that is used to measure and compare virtual desktop deployment platforms. Using patented technology, View Planner generates a realistic measure of the client and server-side performance for all desktops being measured on the virtual desktop platform. The benchmark uses VDI and the hardware infrastructure of your choice and is scalable from a few virtual machines running on one host up to hundreds of virtual machines distributed across a cluster of hosts [13].

The results presented in the paper include five different loads by changing the number of VDI VMs from 48, 96, 144, 192, to 240, which represent 1x to 5x CPU overcommitment. View Planner was run in local mode with the standard profile workload. View Planner workload mix consists of multiple applications running in the desktop virtual machines and performing user operations. The quality of service (QoS) shown in milliseconds is the metric of interest for the benchmark.

Table 5 shows the quality of service for the 95th percentile latency in milliseconds. The first three columns are CPU sensitive (Group A), and the last three columns are storage sensitive (Group B). The default thresholds are 1000 ms for Group A and 6000 ms for Group B. In the case of CPU, we see a minor reduction in the latency using High Performance. Overall, the results are identical for all three profiles. While considering the sensitive storage group, we observe a slight increase in the latency as we change from High Performance to Low Power policy. However, this did not affect the overall system performance.

	QoS Summary (95% Latency group: CPU Sensitive in ms)			QoS Summary (95% Latency Group: Storage Sensitive in ms)		
# VMs	High Performance	Balanced	Low Power	High Performance	Balanced	Low Power
48	530.6	530.8	530.7	1906.8	1926.1	1968.9
96	530.7	530.7	530.8	2015.7	2021.5	2032.5
144	530.7	530.7	530.7	2134.7	2140.6	2146.1
192	530.8	530.8	530.8	2363.0	2367.1	2367.8
240	531.1	531.1	531.2	2424.5	2454.6	2455.0

Table 5: CPU and storage latencies during VDI runs

Figure 15 shows the average host power consumed in watts for various VM runs on the primary Y-Axis. The secondary Y-Axis represents the average CPU utilization and the maximum CPU utilization during each run. We observed that the High Performance policy is consuming higher power, especially for 48 and 96 VMs. This is in line with our prior observations because the C2 idle state is disabled in High Performance—when CPU utilization is low, idle power consumption is higher. As the CPU load increases, the C2 residency in Balanced and Low Power decreases significantly, eroding the differences between High Performance and the other two policies.

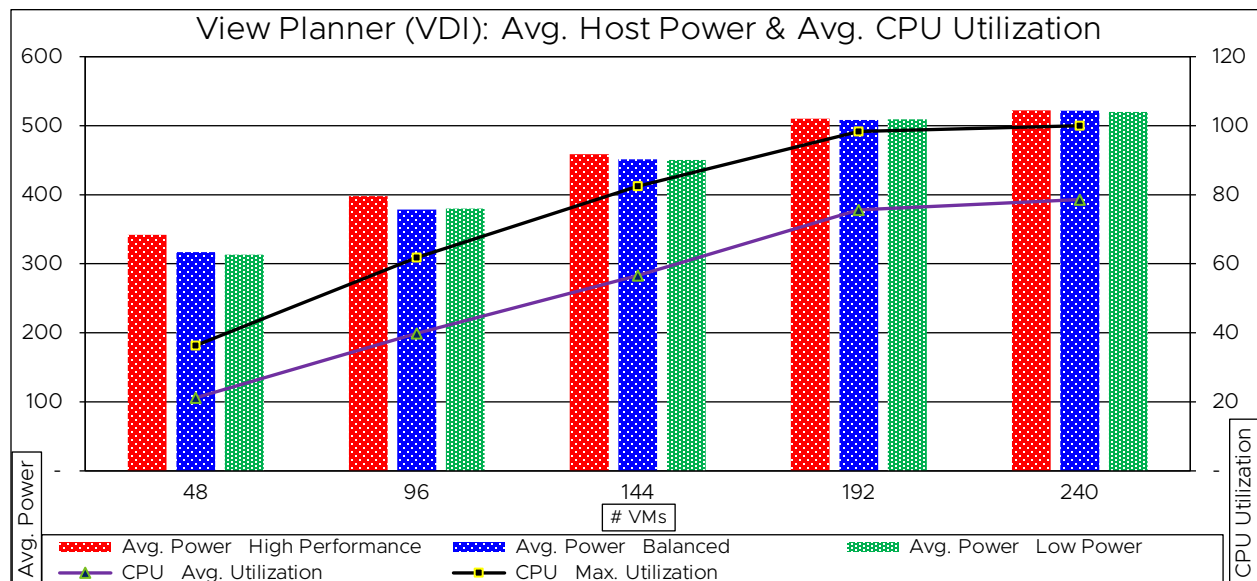


Figure 15: VDI host power consumption and CPU utilization for vSphere power policies

Figure 16 shows the run-time host power consumption for a 96-VM VDI run. The workload (standard_profile) runs five iterations and can be observed by the power fluctuations. The

distinction between the power profiles is especially noticeable in-between iterations and during ramp-down toward the end of the run when the system load is low.

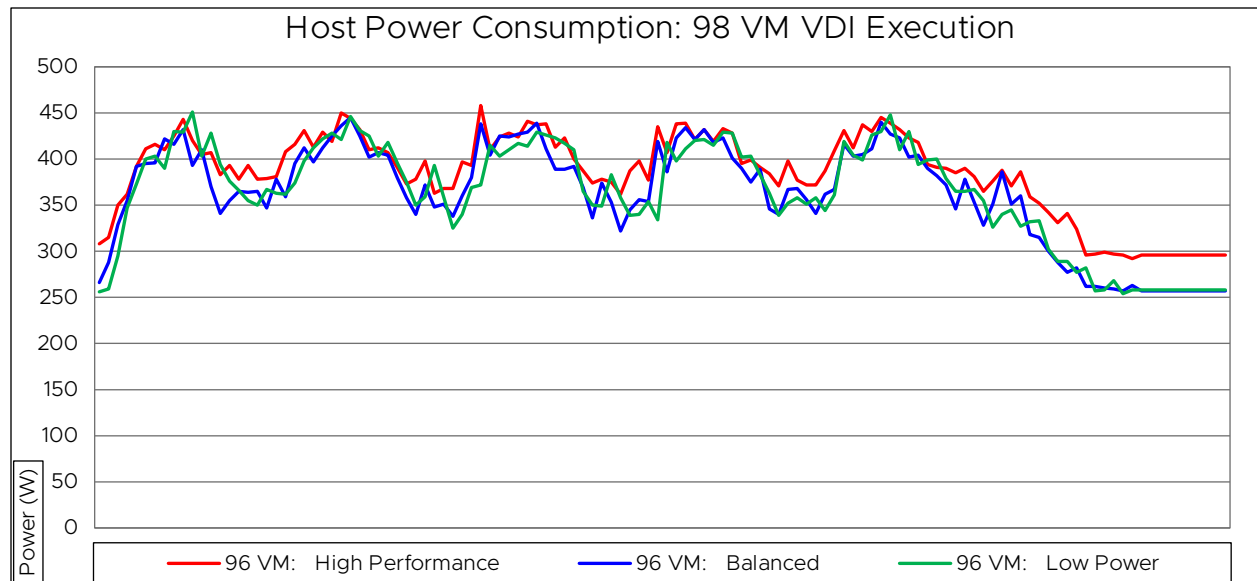


Figure 16: 96-VM VDI run of host power consumption for vSphere power policies

Figure 17 shows the host power consumption while executing the workload on 48, 96, 144, 192, and 240 VMs on the test system in the Balanced policy. The machine reaches saturation at 240 VMs, as the host power consumption peaks throughout the execution.

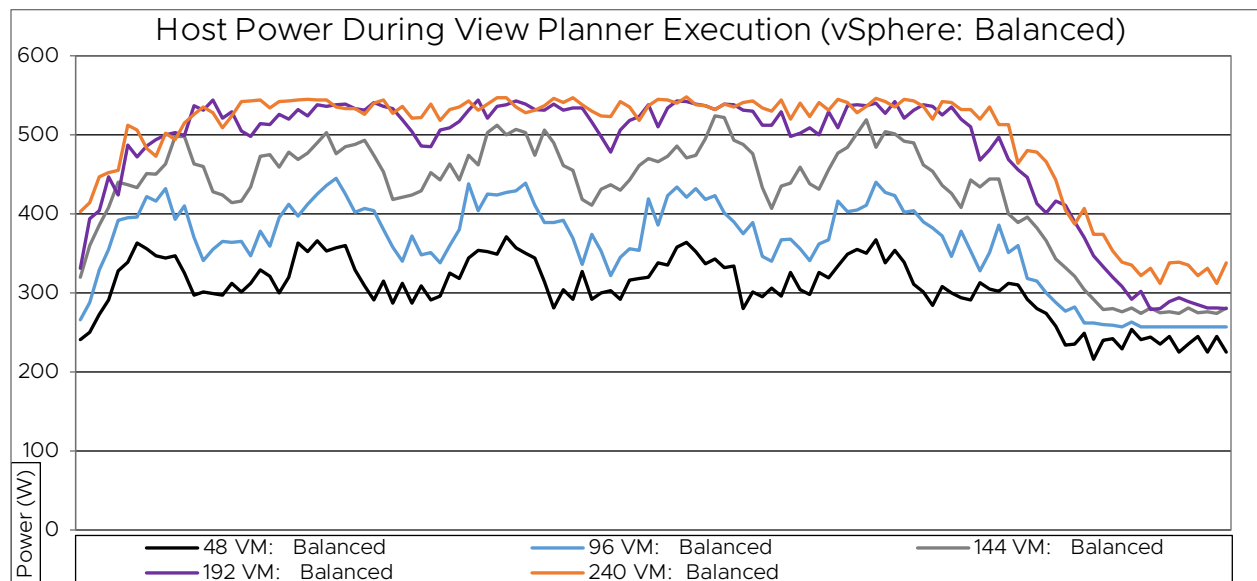


Figure 17: VDI run of host power consumption with the Balanced policy

Impact of vSphere Power Policies on Turbo Frequencies

This section illustrates the impact of different power policies on turbo frequencies. The ability of a processor to enter turbo mode and stay in turbo mode depends on factors such as CPU load, thermal budget, and power budget. The hardware carefully controls aspects of the turbo to operate the processor without sustaining damage. The processor has a specified number of turbo bins available and can be allocated to CPU cores when required. Note that the state of the idle cores directly affects the availability of the power budget to reach high turbo states in the active cores. For example, if the idle cores are in C1, they idle at higher power, reducing active cores' available thermal and power budget.

On the other hand, if the idle cores are in the C2 state, they consume much less power, leaving higher thermal and power budgets for the active cores. For instance, in the test system used for the study with the Cascade Lake processor, the max turbo frequency achieved is 3.10 GHz when all cores are running. But when only four cores are active, the max turbo frequency can go up to 3.70 GHz, given all idle cores are in C2. However, if the idle cores are in C1, the max turbo frequency will be limited to 3.10 GHz.

Since Balanced and Low Power policies use deeper C2-State, they can achieve higher turbo frequencies than the High Performance policy, especially when few cores are active. We ran SPECpower in a 4-vCPU VM on the 48-core system under High Performance and Balanced policies. Figure 18 shows performance on the primary Y-Axis and the average host power consumed on the secondary Y-Axis in the 4-vCPU VM. While only 4 CPU cores are active on the entire system, all other cores remain idle for most of the execution period.

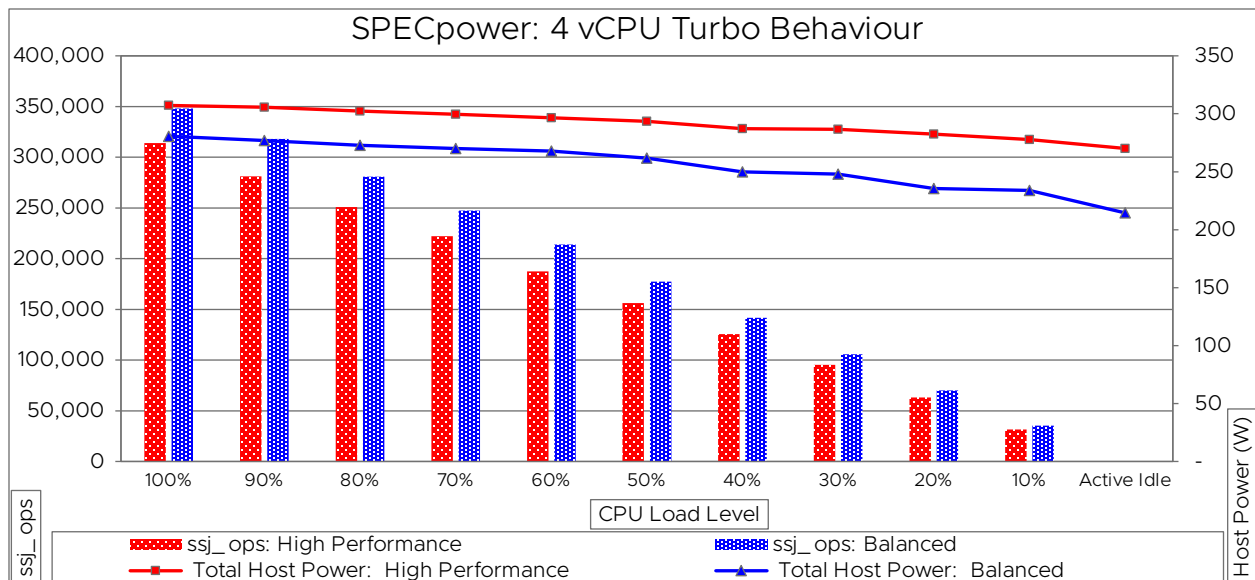


Figure 18: Impact of power policies on turbo mode

The idle cores on the Balanced policy use C2-State and in High Performance policy use C1-State, ready to jump back to the C0 running state if needed, although consuming more power. The graph shows that Balanced Policy achieves ~10% higher performance across all loads due to higher turbo frequencies. Note that the Balanced policy achieves better performance while consuming lower power by putting idle cores to deeper sleep state C2.

Figure 19 shows the %aperf/mperf from esxtop while executing SPECpower in High Performance and Balanced policies. We notice that on active cores, High Performance achieves a maximum %aperf/mperf of ~130% (~3.10 GHz), and Balanced achieves a maximum of ~151% (~3.70 GHz). This indicates that there can be cases when the Balanced policy can provide slightly better performance than the High Performance policy.

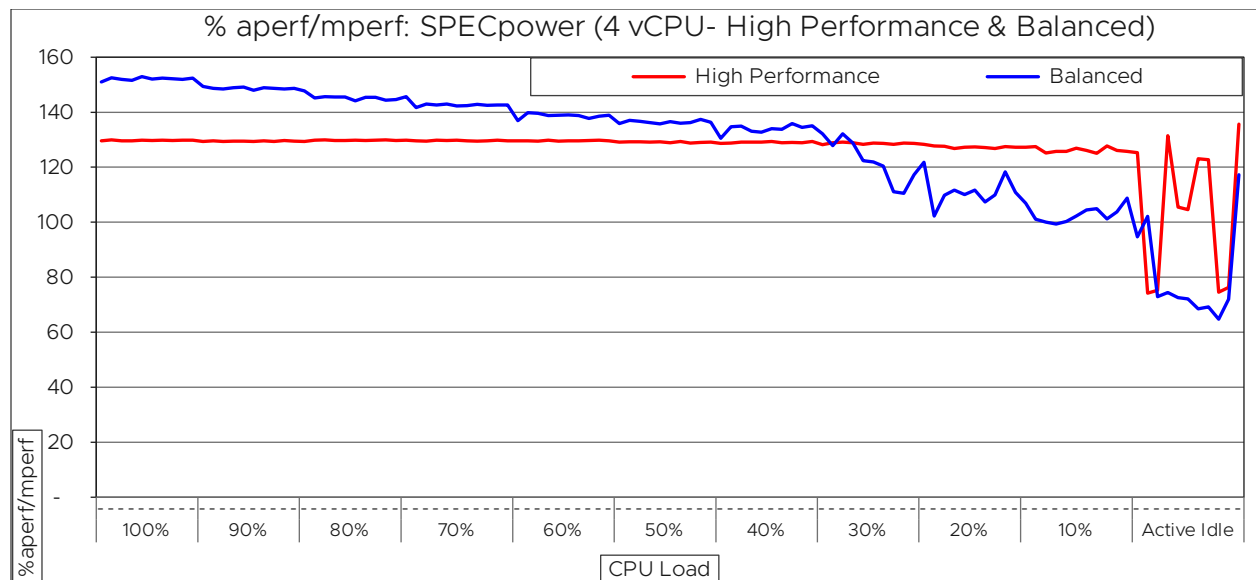


Figure 19: SPECpower %aperf/mperf on High Performance & Balanced

Best Practices

Based on our evaluation of the results presented in this paper, we recommend the following best practices to adopt when using vSphere Power Management as summarized in Table 6:

- Configure your BIOS settings to allow vSphere the most flexibility in using the power management features offered by your hardware. Specifically, select the OS Control mode (Performance Per Watt (OS)) under the BIOS power management options.
- To achieve the best performance-per-watt for various workloads, leave the power policy setting at the default, Balanced.
- Turn Turbo mode on with all C-States enabled to get the maximum performance and power benefits.
- To maximize performance for latency-sensitive applications that must execute within strict constraints on response time, switch the power policy to High Performance.
- For maximum power savings, switch the power policy to Low Power.
- If the host is idle or remains under-utilized for a prolonged amount of time, select Low Power profile to maximize power savings.

Power Policy	Application Focus	Cost vs. Benefits
High Performance	Latency-sensitive applications. (Higher CPU Load)	Reduces latency at the cost of higher idle power
Balanced	A mix of latency and throughput	The balance between latency and throughput
Low Power	Low CPU Load (Underutilizes systems)	Maximizes power savings at the cost of slightly increased latency
Custom	User configuration	Allows the user to tune P-States and C-States based on application requirements

Table 6: vSphere power policy summary

References

- [1] Q. Ali, “Host Power Management in VMware vSphere 5.5.” [Online]. Available: <https://www.vmware.com/techpapers/2013/host-power-management-in-vmware-vsphere-55-10205.html>
- [2] “Host Power Management Policies.” <https://docs.vmware.com/en/VMware-vSphere/7.0/com.vmware.vsphere.resmgmt.doc/GUID-4D1A6F4A-8C99-47C1-A8E6-EF3865603F5B.html>
- [3] A. Shehabi *et al.*, “United States Data Center Energy Usage Report,” LBNL--1005775, 1372902, Jun. 2016. doi: 10.2172/1372902.
- [4] “Advanced Configuration and Power Interface - an overview | ScienceDirect Topics.” <https://www.sciencedirect.com/topics/computer-science/advanced-configuration-and-power-interface>
- [5] “ACPI_6_2.pdf.” Accessed: Aug. 21, 2020. [Online]. Available: https://uefi.org/sites/default/files/resources/ACPI_6_2.pdf
- [6] “Power Management States: P-States, C-States, and Package C-States.” <https://software.intel.com/content/www/us/en/develop/articles/power-management-states-p-states-c-states-and-package-c-states.html> (accessed Aug. 21, 2020).
- [7] “Intel Processor Vendor-Specific ACPI,” *Intel*. <https://www.intel.com/content/www/us/en/standards/processor-vendor-specific-acpi-specification.html> (accessed Aug. 21, 2020).
- [8] “Intel® Xeon® Platinum 8260 Processor (35.75M Cache, 2.40 GHz) Product Specifications.” <https://ark.intel.com/content/www/us/en/ark/products/192474/intel-xeon-platinum-8260-processor-35-75m-cache-2-40-ghz.html> (accessed Aug. 23, 2020).
- [9] “SPECpower_ssj@2008.” https://www.spec.org/power_ssj2008/ (accessed Aug. 23, 2020).
- [10] L. D. Gray, A. Kumar, and H. H. Li, “Workload Characterization of the SPECpower_ssj2008 Benchmark,” in *Performance Evaluation: Metrics, Models and Benchmarks*, Berlin, Heidelberg, 2008, pp. 262–282. doi: 10.1007/978-3-540-69814-2_17.
- [11] “VMmark Virtualization Benchmark,” *VMware*. <https://www.vmware.com/products/vmmark.html> (accessed Aug. 23, 2020).
- [12] V. Makhija, B. Herndon, P. Smith, L. Roderick, E. Zamost, and J. Anderson, “VMmark: A Scalable Benchmark for Virtualized Systems,” 2006.
- [13] “Compare virtual desktop deployment platforms | View Planner,” *VMware*. <https://www.vmware.com/products/view-planner.html> (accessed Aug. 23, 2020).

About the Authors

Ranjan Hebbar is a Member of the Technical Staff in the vSphere Performance team. He has a Ph.D. in Computer Engineering from the University of Alabama in Huntsville (2021), with a focus on improving CPU efficiency by proposing and evaluating several PMU-events driven DVFS techniques. His current research interests span computer architecture, CPU power management, application characterization, throughput computing, and compiler evaluation.

Praveen Yedlapalli is a Staff Engineer I in the vSphere Performance team. He focuses on the CPU and memory management in the vSphere kernel. He is actively working on quantifying and improving power management in vSphere. Praveen has a Ph.D. from Pennsylvania State University (2014), focusing on enhancing CPU/memory bottlenecks. He is the author of many CPU/memory publications and has given talks at various academic and VMworld conferences

Acknowledgments

The authors thank all members of the VMware Performance Engineering team who contributed to and reviewed this paper. In addition, the authors would like to acknowledge Qasim Ali for providing constant technical background, valuable feedback, and support for the article. They also thank Tim Mann, Xunjia (Richard) Lu, and Tony Lin for helping with the experimental section and providing insights to interpret the results.

Appendix A

Table 1A: Custom policy option specifier

Parameter	Description
Power.ChargeMemoryPct	Percentage of idle power consumed by the memory.
Power.CStateMaxLatency	Do not use C-States whose latency is greater than this value.
Power.CStatePredictionCoef	A parameter in the ESXi algorithm for predicting how long a CPU that becomes idle will remain idle. Changing this value is not recommended.
Power.CStateResidencyCoef	When a CPU becomes idle, choose the deepest C-State whose latency multiplied by this value is less than the host's prediction of how long the CPU will remain idle. Larger values make ESXi more conservative about using deep C-States; smaller values are more aggressive.
Power.MaxCpuLoad	Use P-States to save power on a CPU only when the CPU is busy for less than the given percentage of real-time.
Power.MaxFreqPct	Do not use P-States faster than the given percentage of full CPU speed, rounded up to the next available P-State.
Power.MinFreqPct	Do not use any P-States slower than the given percentage of full CPU speed.
Power.PerfBias	Performance Energy Bias Hint (Intel only). Sets an MSR on Intel processors to an Intel-recommended value. For example, Intel recommends 0 for high performance, 6 for balanced, and 15 for low power. Other values are undefined.
Power.PerfBiasEnable	Performance Energy Bias Hint (Intel-only). Sets an MSR on Intel processors to an Intel-recommended value. For example, Intel recommends 0 for high performance, 6 for balanced, and 15 for low power. Other values are undefined.
Power.TimerHz	Controls how many times per second ESXi reevaluates which P-State each CPU should be operating on.
Power.UseCStates	Use deep ACPI C-States (C2 or below) when the processor is idle.
Power.UsePStates	Use ACPI P-States to save power when the processor is busy.