Performance of vSphere 6.7 Scheduling Options

Performance Study - April 11, 2019

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

Table of Contents

Executive summary

VMware vSphere® 6.7 U2 includes new scheduler options that secure it from the L1TF vulnerability, while also retaining as much performance as possible. This paper provides an overview of the security issues, description of this new scheduler option, and the results of performance testing with different scenarios. Depending on many factors, the overall impact to performance varies, but, in general, the new scheduler option allows for most scenarios to recapture a significant amount of the performance that was lost due to the L1TF security mitigations and patches.

Introduction

In August of 2018, a security vulnerability known as L1TF [1], affecting systems using Intel processors, was revealed, and patches and remediations were also made available. Intel provided micro-code updates for its processors, operating system patches were made available, and VMware provided an update for vSphere. The full details of the vCenter and ESXi patches are in a VMware security advisory [2] that links to individual KB articles.

The ESXi-provided patches included a Side-Channel Aware Scheduler (SCAv1) that mitigates the concurrent-context attack vector for L1TF. Once this mode is enabled, the scheduler will only schedule processes on one thread for each core. This mode impacts performance mostly from a capacity standpoint because the system is no longer able to use both hyper-threads on a core. A server that was already fully utilized and running at maximum capacity would see a decrease in capacity of up to approximately 30%. A server that was running at 75% of capacity would see a much smaller impact to performance, but CPU use would rise.

In vSphere 6.7 U2, the side-channel aware scheduler has been enhanced (SCAv2) with a new policy to allow hyper-threads to be utilized concurrently if both threads are running vCPU contexts from the same VM. In this way, L1TF side channels are constrained to not expose information across VM/VM or VM/hypervisor boundaries

To further illustrate how SCAv1 and SCAv2 work, the diagrams below show generally how different schedulers might schedule vCPUs from VMs on the same core. When there is only one VM in the picture (figure 1), the default placement, SCAv1 and SCAv2 will assign one vCPU on each core. When there are two VMs in the picture (figure 2), the default scheduler might schedule vCPUs from two different VMs on the same core, but SCAv1 and SCAv2 will ensure that only the vCPUs from the same VM will be scheduled on the same core. In the case of SCAv1, only one thread is used. In the case of SCAv2, both threads are used, but this always occurs with the vCPUs from the same VM running on a single core.

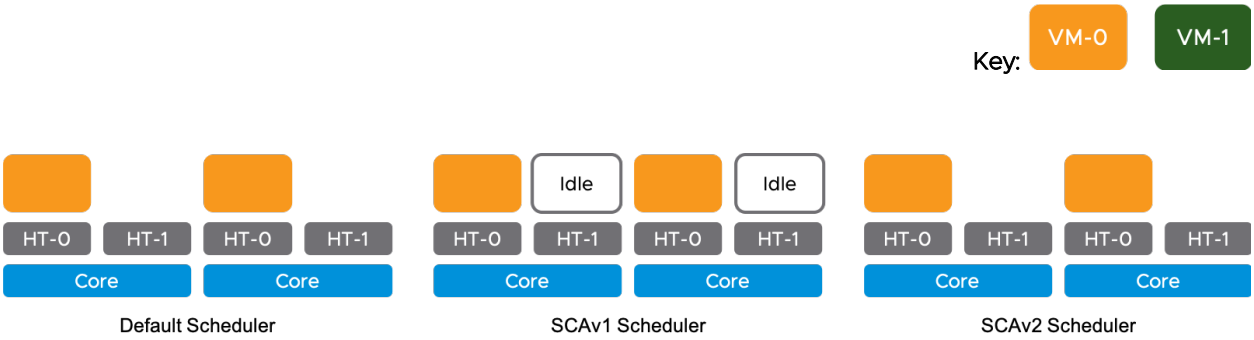

Figure 1. One 2-vCPU VM on two hyper-threaded cores (undercommitted)

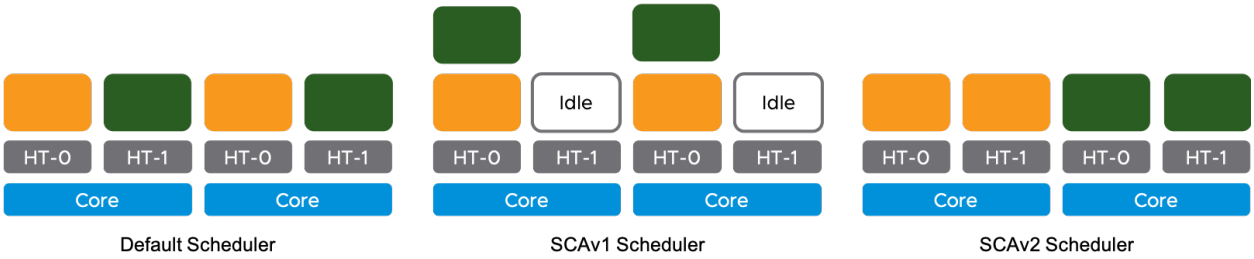

Figure 2. Two 2-vCPU VMs on two hyper-threaded cores (overcommitted)

This new policy from SCAv2 has the following characteristics.

Hardened address-space isolation: SCAv2 implements a hardened address space separation across VM/VM and VM/hypervisor to ensure that no secret data can leak across VM boundaries.

Coordinated VM execution: SCAv2 carefully coordinates the execution of VMs so that no VM/hypervisor or contexts from different VMs can run concurrently on hyper-threads of the same physical core. Only guest execution from the same VM is allowed to share hyper-threads.

Dynamic placement optimization: SCAv2 dynamically adapts the workload placement algorithm at runtime for optimal performance based on server utilization. When the server is under-committed, SCAv2 will spread work evenly across multiple physical cores to achieve maximum performance. When the server is over-committed, SCAv2 will consolidate work securely onto hyper-threads to make use of the additional capacity provided by hyper-threading.

Security

The choice of ESXi scheduling policy depends on the security boundaries that need to be enforced.

A CPU that is vulnerable to L1TF can expose information via the concurrent-context attack vector. A VM running on one thread of a core can observe information used by the other thread on that core. To control what information is exposed, you must choose the proper scheduling policy: Host Security Boundary, VM Security Boundary, or Process Security Boundary.

When running on a CPU that is not vulnerable to L1TF, ESXi always uses the scheduling model with the highest performance.

Host security boundary

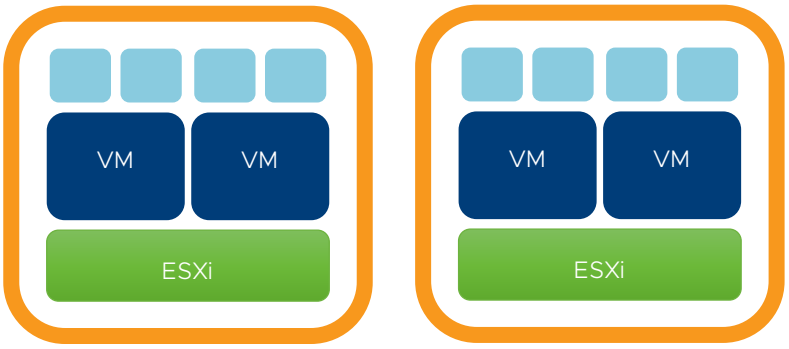

Figure 3. Model of host security boundary

The host security boundary allows information leakage between all VMs and the hypervisor running on a given host. VMs on the host must all be considered part of the same information security boundary as each other and the hypervisor.

This default scheduling policy retains full performance and is used when **hyperthreadingMitigation=FALSE**.

Relying on the host security boundary is not recommended. Using L1TF concurrent-context attack vector, a VM on the host can observe any information on that host. All credentials, crypto keys, and secrets used by the hypervisor or other VMs on the host could be obtained by a malicious guest. This could include domain credentials, crypto keys, or other secrets that have long term value to attackers.

VM security boundary

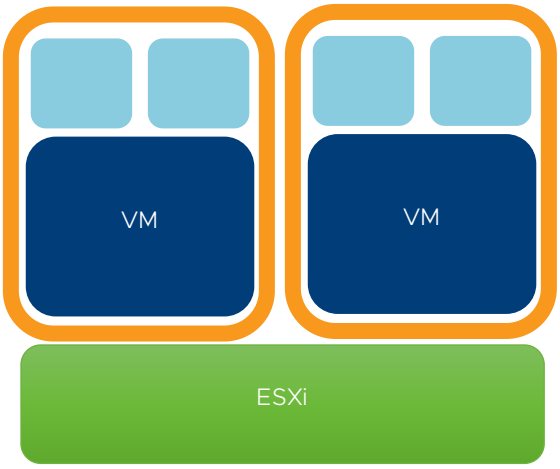

Figure 4. Model of VM security boundary

The VM security boundary prevents information leaking between two different VMs on a host or between a VM and the hypervisor. Concurrent-context speculative side-channel attacks can be used to reveal information across different security domains of a single ESXi VM.

This is SCAv2 and is enabled by setting **hyperthreadingMitigation= TRUE** and **hyperthreadingMitigationIntraVM= FALSE**.

This option provides a balance of performance and security for environments where the VM is considered the information security boundary.

Process security boundary

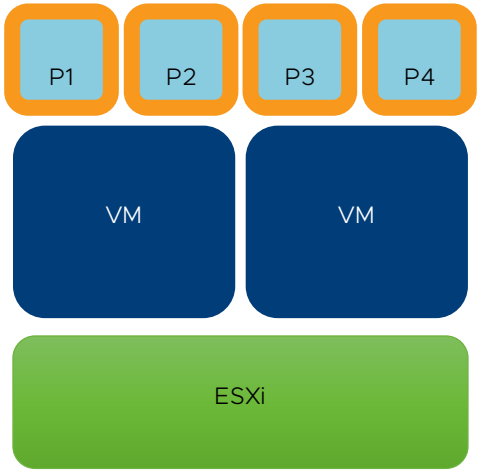

Figure 5. Model of process security boundary

The process security boundary ensures that concurrent-context attacks using speculative side channels do not expose information across different processes or security contexts within the guest.

This is SCAv1 and is enabled by setting **hyperthreadingMitigation= TRUE** and **hyperthreadingMitigationIntraVM= TRUE**.

Because VMware guests do not know about the underlying hyper-threading topology of the host, this is the only option that can prevent concurrent-context speculative side-channel attacks within a VM. This option has the highest security and lowest performance.

For more details on L1TF mitigations, please see VMware KB 55806 [3].

Configuration of scheduler policies in vSphere 6.7 U2

vSphere 6.7 U2 will use the default unmitigated scheduler unless configured to use one of the other options. In order to use SCAv1 or SCAv2, a couple of VMkernel options must be set and the host rebooted for it to take effect. If upgrading from a previous version where the VMkernel settings were previously changed, they will be preserved after the upgrade. Options for scheduler policy configurations:

Default – Unmitigated and most performant – Not concerned about L1TF security vulnerability.

HyperthreadingMitigation = FALSE

HyperthreadingMitigationIntraVM = N/A

SCAv1 - Auto Hyperthreading off – What was made available at L1TF announcement in August 2018.

HyperthreadingMitigation = TRUE

HyperthreadingMitigationIntraVM = TRUE

SCAv2 - New policy in vSphere 6.7 U2

HyperthreadingMitigation = TRUE

HyperthreadingMitigationIntraVM = FALSE

There are several ways that these values can be set. VMware KB 55806 [3] covers them in detail. A brief summary is provided below, as well.

In the vSphere Client, you can use the Advanced System Settings for the host to filter for and set **VMkernel.Boot.hyperthreadingMitigation** and **VMkernel.Boot.hyperthreadingMitigationIntraVM** options, as shown in the screenshot below.

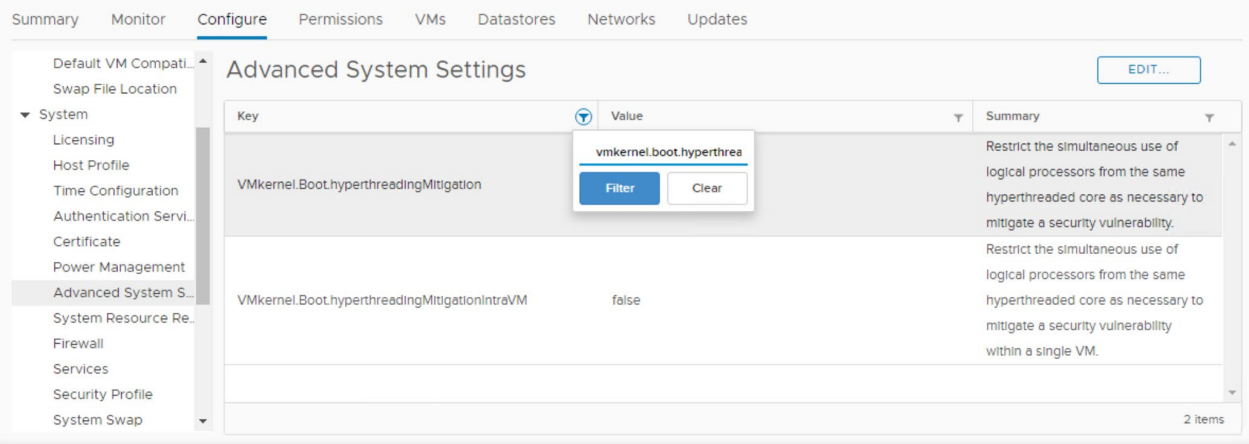

Figure 6. Advanced System Settings, where you can set VMkernel options to enable the new policy in vSphere 6.7 U2

You can instead use esxcli to set these options as shown here:

```
esxcli system settings kernel set -s hyperthreadingMitigation -v TRUE
esxcli system settings kernel set -s hyperthreadingMitigationIntraVM -v FALSE
```

As another option, you can run the **esxcfg-advcfg** command from the ESXi shell directly if needed. For example, the following commands would be used in the ESXi shell to enable the SCAv2 scheduler:

```
esxcfg-advcfg -k TRUE hyperthreadingMitigation
esxcfg-advcfg -k FALSE hyperthreadingMitigationIntraVM
```

Performance testing

A variety of workloads with a range of load levels were tested with the new scheduling options on vSphere 6.7 U2. In summary, the results showed that, in most cases, SCAv2 returned some of the performance lost due to the mitigations made necessary by the L1TF security vulnerability. SCAv2 accomplishes this by allowing the system to be able to use the hyper-threads while still maintaining the mitigation of L1TF security vulnerabilities with careful scheduling of only "sibling" vCPUs of a virtual machine on the same core. Some workloads and configurations benefit more or less from SCAv2 depending on a variety of factors. This section looks at some specific examples in detail.

Performance capacity

The performance effect of the L1TF mitigations is mostly an impact to overall system capacity. Intel Hyper-Thread technology has the ability to run two threads on a single core, but due to the issues found with the L1TF vulnerability, it is no longer possible to freely schedule processes on these two threads at the same time from a security point of view. This additional thread on the same physical core typically increases the amount of work that a server can do by 15% to 30%. Without the ability to use hyper-threads, the total capacity of the server drops by a similar amount.

If a server is using all of its cores at very high utilization levels, then it is at or near full capacity. The loss of the ability to use hyper-threads will result in a noticeable loss in performance that will still depend on the workload, but it could be approximately 30% in the worst case.

A server that is running with only some of its cores at very high utilization levels with some available capacity is common in many environments. The impact of the L1TF mitigations in this environment will mostly show up as increased CPU utilization levels on the host and a small impact to performance as seen by the applications or end users.

For these reasons, we tested with each of the workloads—as much as the specific workload would allow—at full utilization and at a targeted 75% load level. This allowed us to find the maximum and the reduced impact levels. The maximum impact measures a system with an unusually high utilization and load level, while the reduced impact measures a system with a utilization and load level that are more common in production.

For each test workload, we report a range of performance impact. The range of impact is reported for both SCAv1 that doesn't use hyper-threads, and SCAv2 which does use hyper-threads but with the methodology described earlier in this paper. We expect that most environments will be approximately in the middle of these ranges, but it will depend on the actual usage level of the host.

Results

Summary of results

This summary of results shows a range of expected impact in performance for both SCAv1 and SCAv2 as compared to the default scheduler as the baseline of performance. If SCAv1 or SCAv2 were able to achieve the same performance, it would be 1.0, and if it achieved 75% of the performance, it would be .75. The range in the chart is the maximum performance impact and the impact at the reduced load.

Due to the way the View Planner benchmark works, we weren't able to get the same type of reduced load numbers as the other benchmarks. Please see the View Planner section below for full details.

	Performance shown as percent of default	
Workload	SCAv1 Scheduler	SCAv2 Scheduler
SQL Server on Windows with HammerDB	.64 to .65	.85 to .99
TCP-V on Linux	.70 to .81	.84 to .94
View Planner VDI benchmark with Windows desktops	.69	.83
VMmark 3.1 multi-VM and infrastructure operations benchmark	.80 to .94	.91 to .94
Monster VM with Oracle and DVD Store 3	.81 to .94	.75 to .89

Table 1. Summary of results for various benchmarks

The graphs below split out the maximum and reduced load numbers for each test and SCAv1 and v2. It is the same data as presented in the above chart, but organized slightly differently.

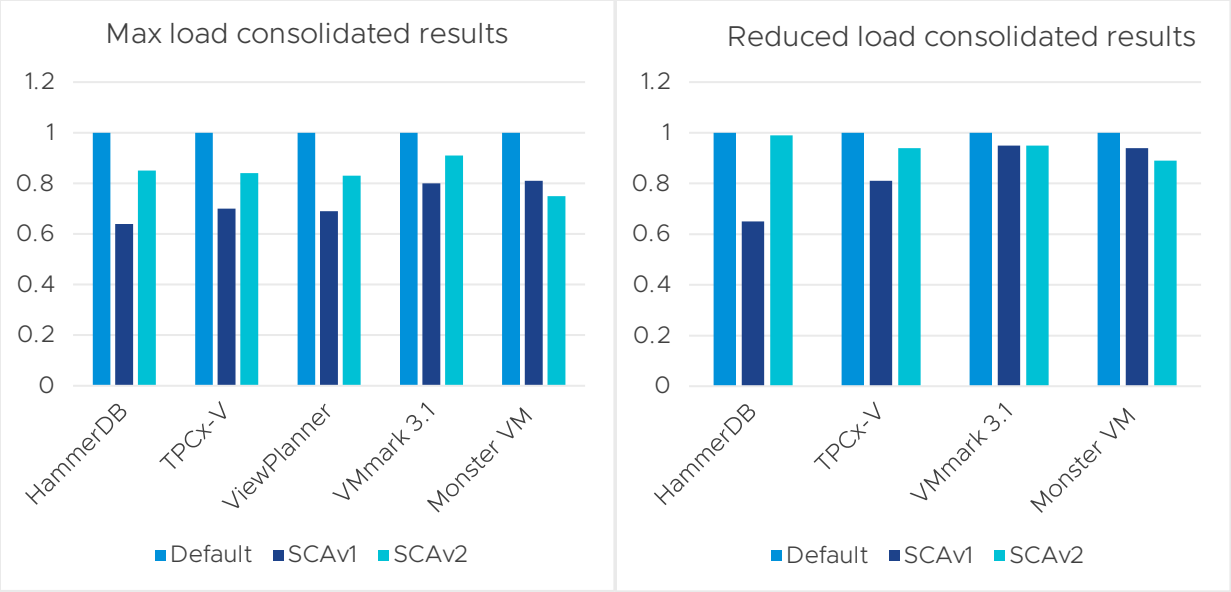

Figure 7. Summary of results for various benchmarks

The charts show that the SCAv2 scheduler, represented by the third bar in each group, recovers a significant percentage of performance in all cases except for the Monster VM test case. The reduced load numbers show that at server usage levels of approximately 75%, the overall impact to performance is much lower. With SCAv2 and the reduced load, tests show that the largest performance impact measured in these tests was 11%.

The details of each of these tests is in the following sections.

SQL Server on Windows with HammerDB

HammerDB is an open source database benchmark used to test various workload types. It supports essentially all of the major databases, but for this test, we used SQL Server 2016.

The server used for the SQL Server testing was a dual-socket system with Intel Xeon Gold 6148 @ 2.4 GHz processors. Each processor has 20 cores, but for the purposes of these tests, we limited it to 4 cores in the BIOS, resulting in a total of 8 cores and 16 threads with hyper-threading enabled. This allowed for virtual CPU to physical CPU overcommitment tests to be easily performed.

In the initial tests, two VMs with 4 vCPUs and one VM with 8 vCPUs were created with 32 GB of vRAM assigned to each. This resulted in 16 vCPUs total across the three VMs, which is an overcommitment ratio of 2x (2 vCPUs for every 1 physical CPU). In the default and SCAv2 test cases, this meant that the number of vCPUs was exactly twice the number of cores and the same as the logical threads on the server. In the case of the SCAv1 scheduler, where hyper-threading is essentially off, there are twice the number of vCPUs as logical threads.

An additional set of tests were done with the same VMs, but the vCPUs of one of the 4 vCPU VMs was increased to 8. This increased the total number of vCPUs to 20, which increases the overcommitment ratio from 2x to 2.5x. The 2.5x overcommitment tests were the maximum load tests for these HammerDB tests.

A final set of tests were run at a reduced load level to simulate a more typical customer situation where the system isn't being pushed all the way to the limits of its capabilities. In this test, an 8 vCPU VM and a 4 vCPU VM were run for a total of 12 vCPUs and an overcommitment ratio of 1.5x.

The tests were run on SQL Server 2016 on Windows Server 2016 using HammerDB's TPC-C workload profile. Users were configured with 25 per 4 vCPU VM and 80 per 8 vCPU VM. The HammerDB load client was loaded on each of the SQL Server VMs locally.

Total vCPUs / pCPUs	Ratio of vCPUs to pCPUs	SCA v1 OPM (% of Baseline)	SCA v2 OPM (% of Baseline)
16 / 8	2.0	.65	1.01
20 / 8 (Max load)	2.5	.64	.85
12 / 8 (Reduced load)	1.5	.65	.99

Table 2. Test results for the HammerDB TPC-C workload with a SQL Server 2016 database

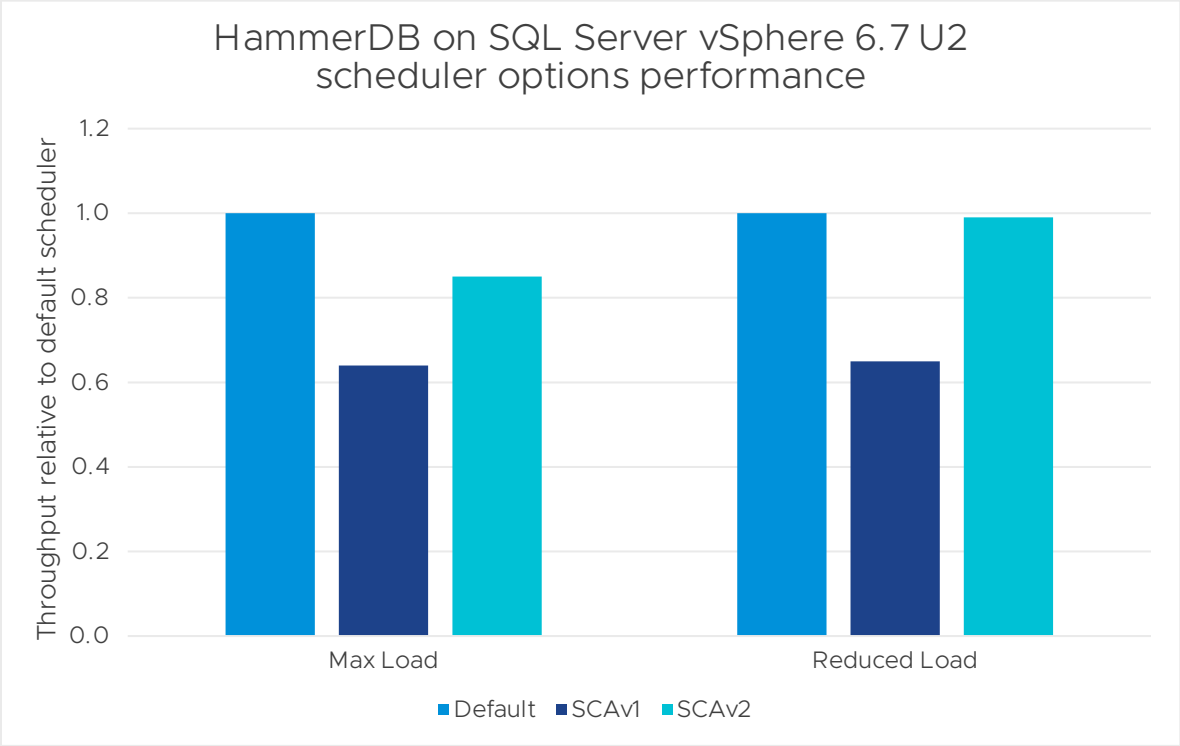

Figure 8. SQL Server 2016 performance with different scheduler options in vSphere 6.7 U2

The results show that the performance impact of the SCAv1 scheduler was 35%, but the SCAv2 scheduler is able to improve performance significantly. In the case of the maximum load test scenario, the performance impact of mitigation was lessened by a little more than half to only 15%.

In the reduced load scenario, the SCAv2 scheduler returned almost all of the performance. With two VMs running the SCAv1 scheduler that didn't use hyper-threading, this configuration was able to achieve .65 of the default scheduler performance, while the v2 scheduler was able to achieve .99.

TPCx-V derived workload on PostgreSQL Linux

TPCx-V is a virtualization database workload that was used as the basis for the tests in this section. The results here are non-compliant and cannot be compared to published results. For more details on the TPCx-V workload, please see the TPCx-V User's Guide [4].

TPCx-V measures the performance of a virtualized server platform under a demanding database workload. It stresses CPU and memory hardware, storage, networking, hypervisor, and the guest operating system. The TPCx-V workload is database-centric and models many properties of cloud services, such as multiple VMs running at different load demand levels, and large fluctuations in the load level of each VM.

For these tests, we configured a single tile of VMs on the host. A tile has a total of 12 VMs, divided into 4 groups. Each group has 3 VMs. VM1 receives the transactions and invokes PostgreSQL stored procedures to execute them. The decision support transactions (high IOPS, low CPU demand) are submitted to VM2, the OLTP transactions (low IOPS, high CPU demand) are submitted to VM3. The 4 groups differ in the amount of load they receive. Groups 1-4 receive 10%, 20%, 30%, and 40% of the overall load, respectively. So, there is a wide variety of workload types and levels among the 12 VMs.

A two-socket host with Intel Xeon 6148 Gold @ 2.4 GHz processors and 768 GB of RAM was used. Each processor had 20 cores for a total of 40 cores and 80 threads with hyper-threading enabled for the system.

VMs were configured to have from 1 to 16 vCPUs and from 2 to 80 GB of RAM. The listing in the table below shows the specifics for all 12 VMs that were used. CentOS 7.3 and PostgreSQL 9.3 were installed on all VMs.

VM	vCPUs	RAM		VM	vCPUs	RAM
1	1	2 GB		7	4	2
2	2	26.4		8	4	51.8
3	4	14.6		9	12	24.4
4	2	2		10	4	2
5	4	43		11	6	80.1
6	8	19.5		12	16	48.8

Table 3. Results of a TPCx-V derived workload run against a PostgreSQL database

Results are reported in terms of transactions per second (TPS) and average response time (ms).

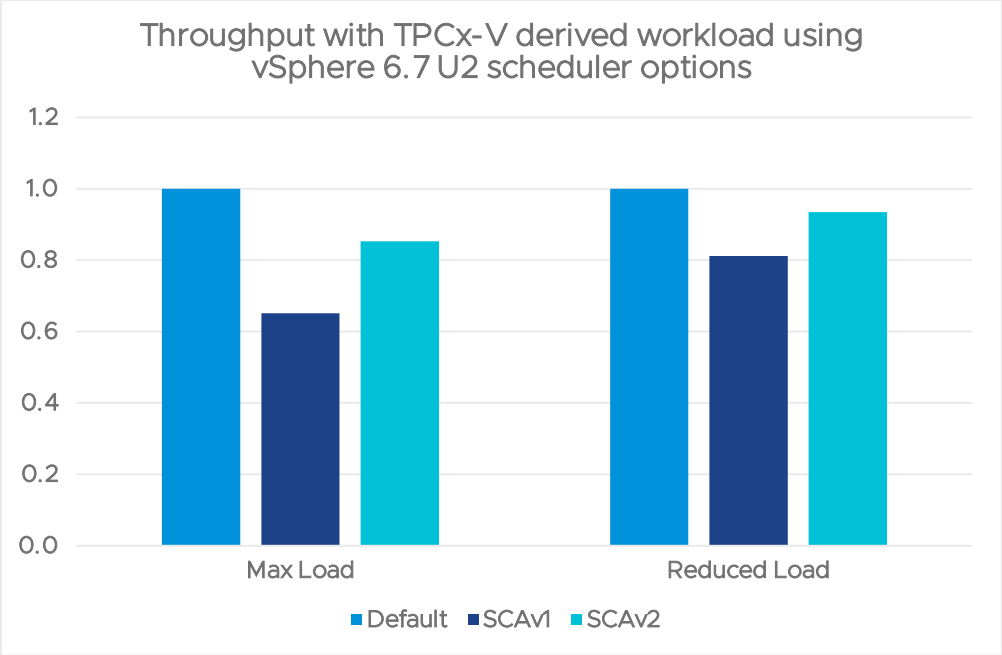

Figure 9. Results of a TPCx-V derived workload run against a PostgreSQL database

	Default Throughput / RT	SCAv1 Throughput / RT	SCAv2 Throughput / RT
Max Load	1271 tps / 20 ms	828 tps / 20 ms	1084 tps / 20 ms
Reduced Load	966 tps / 13 ms	784 tps / 15 ms	903 tps / 13 ms

Table 4. Test results for throughput for transactions per second (tps) and average response time (ms)

The peak for SCAv2 is 85% of the peak for the default scheduler. But if we pull back the load to 76% of peak, throughput with SCAv2 is at 94% of the default with response times matching. This shows that at lower than maximum capacity levels, the performance impact is greatly decreased with the SCAv2 scheduler returning a large percentage of performance that was lost with SCAv1.

Oracle Database on Linux with a large monster VM

A single, large VM consuming all of the host was created on an Intel four-socket server with Intel Broadwell-based processors. The server had 24 cores per socket and 96 cores total with 192 threads when hyper-threading was enabled. The VM was installed with CentOS 7.5 and Oracle Database 12c. The benchmark used for testing was the open source DVD Store 3 [5], which simulates an online store. The VM was assigned 256 GB of RAM and a test database of approximately 240 GB was created. A Fibre Channel–based array with flash disks was used for storage.

In all test configurations, the load was driven up higher by steadily increasing the number of test users that were running on a separate load-driver VM. The maximum throughput was found for each configuration and is reported below.

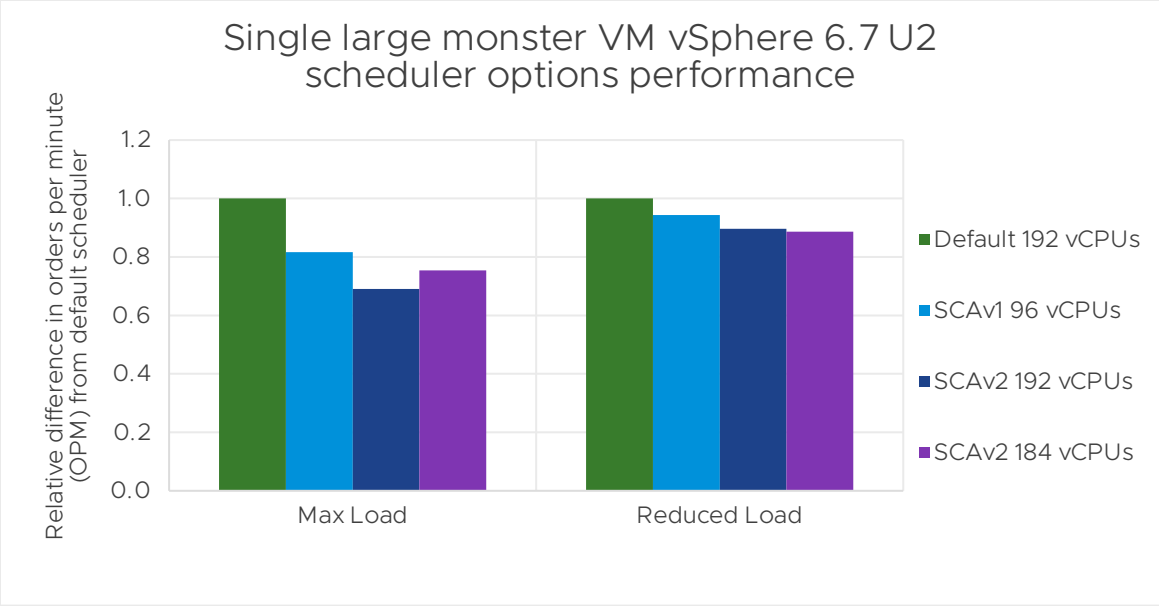

Figure 10. Performance scheduler options with a single, large monster VM running on vSphere 6.7 U2

The performance of the default scheduler test case with a 192 vCPU VM was the best, achieving just over 180,000 operations per minute at its peak. The next best performing was the SCAv1 scheduler. This test case used a VM that was 96 vCPUs because that was the maximum size VM that could be created with this scheduling mode where it is not possible to use the hyper-threads.

Two tests with the SCAv2 scheduler are reported here, both below the performance of the SCAv1 scheduler. The lowest performing test was a 192 vCPU VM with SCAv2. In order to improve the performance of the VM, we reduced the size by one core per socket (2 vCPUs per socket) to 184 vCPUs. This allows the system to have some available cores to run networking worlds without having to deschedule the VM.

In this test case with a single large VM, we did uncover a scenario where SCAv2 did not improve on the performance of the SCAv1 scheduler. This performance difference is mitigated some by reducing the size of the VM slightly, but it is still a bit lower. In the case where there is a single, large VM consuming the entire host, SCAv1 may provide a slight performance advantage over SCAv2.

The reduced load test point was shown to be only a 5% and 10% difference from the default for SCAv1 and SCAv2 respectively. This compares with 19% and 25% differences at the higher max load datapoint. Systems that are not at maximum capacity will not see as big of an impact to performance.

Virtualization infrastructure and mixed load – VMmark3

VMmark3 [6] is VMware's virtualization benchmark that combines several workloads to simulate the mix of applications in customer datacenters. It is based around the notion of a defined set of VMs called a tile. Each tile contains all of the VMs needed for the workloads that make up VMmark. Load is incremented by adding more tiles of VMs. Each workload in all tiles must meet the defined quality-of-service metrics in order for a run to be valid. At the end, a score is generated based on the number of tiles and how well the workloads performed within each tile.

The host used for these tests was a dual-socket server using Intel Xeon Platinum 8180 @ 2.5Ghz 56 core processors and 768 GB of RAM.

For these tests, VMmark was tested in single host mode, meaning that all infrastructure operations were disabled and as many tiles as possible were run on a single host. The maximum number of tiles that could be supported for each scheduler option was determined, as well as the VMmark score for these runs. The graph below shows the results.

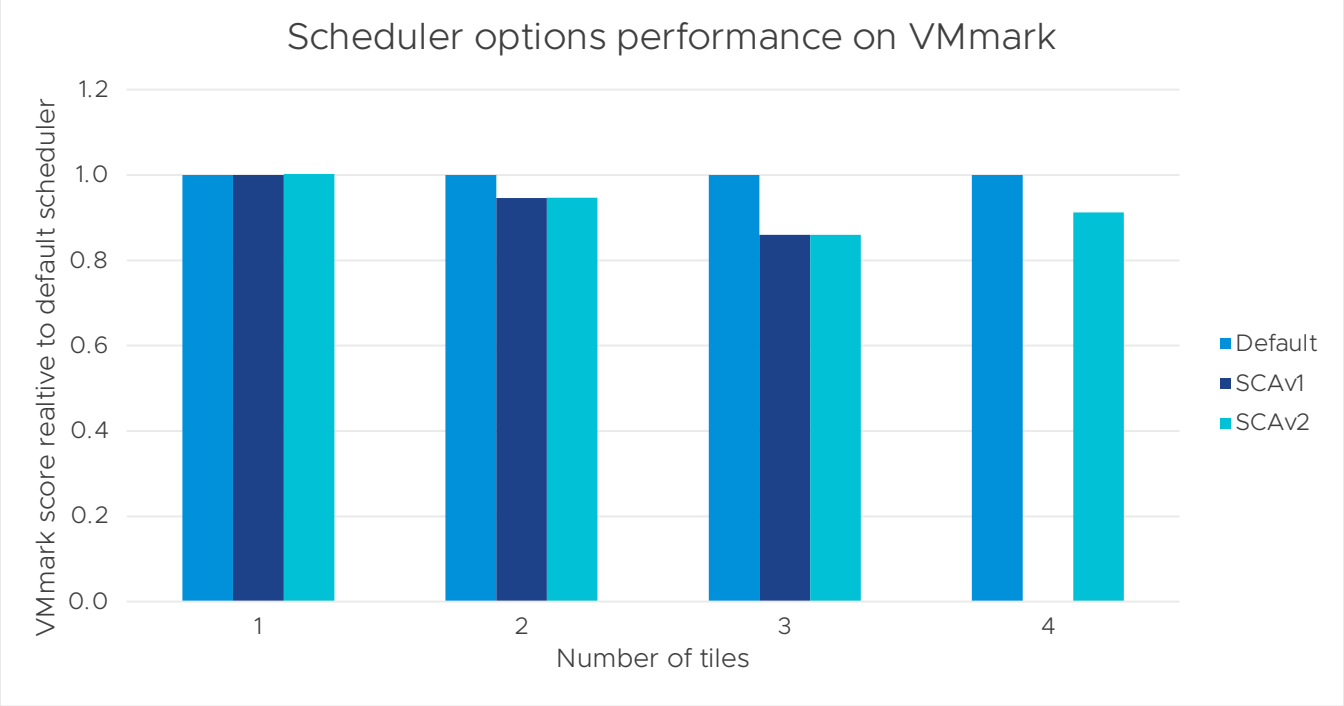

Figure 11. Performance of scheduler options using the VMmark benchmark

With one and two tiles, all three options are at similar performance levels, but when we reach three tiles, things begin to separate. The default scheduler achieves a higher score because it maintains better overall response time and throughput in the component VMmark workloads. With four tiles, the SCAv1 scheduler was not able to successfully run with acceptable application performance levels. The SCAv2 scheduler was able to keep up with the default scheduler and still run four tiles, but at a reduced score due to some lower performance in the tile applications.

Virtual desktop infrastructure – View Planner

View Planner [7] is VMware's benchmark for virtual desktops. It simulates users running many popular applications doing common user tasks. The tests measure how many virtual desktops can be supported while still maintaining response times better than the defined maximum allowed. This ensures that the user's virtual desktops are still responsive at peak load. Results are measured in terms of number of users.

A two-socket server with Intel Xeon 6140 Gold @ 2.3 GHz processors and 768 GB of RAM was used for these tests. The server had 18 cores per socket and 36 cores total with 72 threads when hyper-threading was enabled. The virtual desktop VMs were configured to have 2 vCPUs, 4 GB of memory, 50 GB of disk space, and Windows 10.

An initial baseline test was done with the default scheduler, followed by tests with the SCAv1 and SCAv2 schedulers. The results below show the number of virtual desktop sessions that could be supported for each configuration.

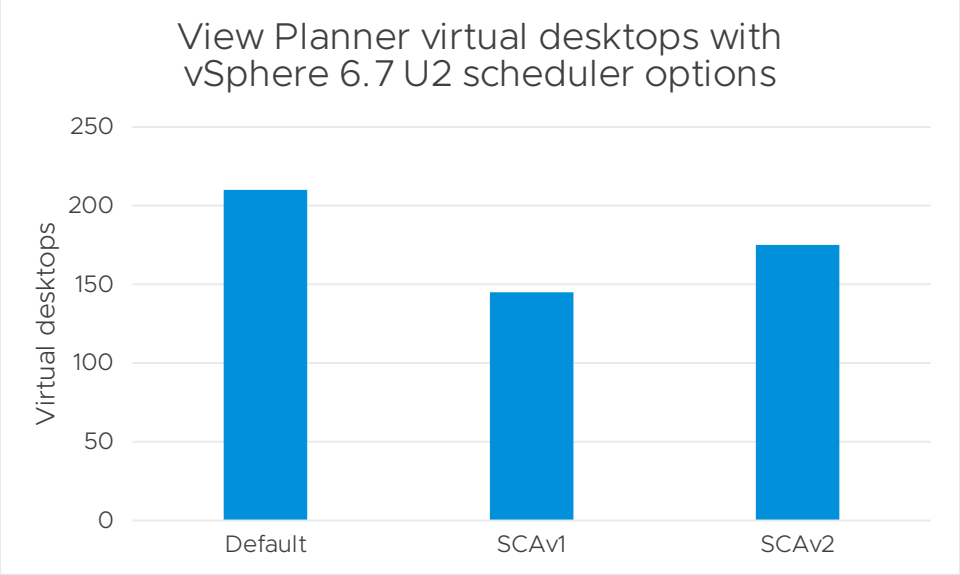

Figure 12. Comparison of the number of virtual desktops able to run in the View Planner workload

The default achieved a maximum of 210 desktops, while SCAv1 was able to achieve 145, and SCAv2 reached 175. This shows that, once again, SCAv2 was able to regain some of the performance that was lost with the SCAv1, while still remaining secure from L1TF vulnerabilities.

Conclusion

The security vulnerabilities introduced with L1TF can now be mitigated safely with the new SCAv2 option available in vSphere 6.7 U2. This new scheduler option allows the hyper-threads to be used only by scheduling vCPUs from the same VM on the same core, and this regains some of the performance lost due to mitigations made necessary initially. VMware KB 55806 [3] is the main article for detailed information about L1TF vulnerabilities and patches, and it should be used for additional information, including current patch and update availability.

References

[1] Intel Corporation. (2018) Resources and Response to Side Channel L1TF. https://www.intel.com/content/www/us/en/architecture-and-technology/l1tf.html

[2] VMware, Inc. (2018, August) VMSA-2018-0020: VMware vSphere, Workstation, and Fusion updates enable Hypervisor-Specific Mitigations for L1 Terminal Fault - VMM vulnerability. https://www.vmware.com/security/advisories/VMSA-2018-0020.html

[3] VMware, Inc. (2019, February) VMware response to "L1 Terminal Fault - VMM" (L1TF - VMM) Speculative-Execution vulnerability in Intel processors for vSphere: CVE-2018-3646 (55806). https://kb.vmware.com/s/article/55806

[4] TPC. (2015, August) TPCx-V User's Guide. http://www.tpc.org/tpcx-v/resources/tpcxv_users_guide/tpcxv_users_guide.html

[5] Todd Muirhead. DVD Store 3. https://www.github.com/dvdstore/ds3

[6] VMware, Inc. (2019, April) VMmark. https://www.vmware.com/products/vmmark.html

[7] VMware, Inc. (2019, January) View Planner. https://www.vmware.com/products/view-planner.html

About the authors

This paper was a collaboration between several people; they include: Qasim Ali, David Dunn, Rahul Garg, Xunjia Lu, Todd Muirhead, Reza Taheri, and James Zubb.