

vMotion Compatibility ...



fr0gger03 16 posts since

Apr 1, 2005

Vmotion compatibility for:

Intel CPU -

http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1991

AMD CPU -

http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1992

Overriding Masks -

http://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1993



fr0gger03 16 posts since

Apr 1, 2005 1. Re: vMotion Compatibility Apr 1, 2008 7:05 PM

Deeper dive - for those of you who care

Put on your pajamas, 'cuz this could put you to sleep. If you happen to be a fella like me with a software background, the following might be helpful for the below discussion of vMotion and CPU compatibility. **All Hail Wikipedia!**

Scalar Processors (From http://en.wikipedia.org/wiki/Scalar_processor)

Scalar processors are the monogamy class of computer processors.[1] A scalar processor processes one data item at a time (typical data items being integers or floating point numbers). In a vector processor, by contrast, a single instruction operates simultaneously on multiple data items. The difference is analogous to the difference between scalar and vector arithmetic.

Vector Processors (From http://en.wikipedia.org/wiki/Vector_processor)

A vector processor, or array processor, is a CPU design that is able to run mathematical operations on multiple data elements simultaneously. This is in contrast to a scalar processor which handles one element at a time. The vast majority of CPUs are scalar (or close to it). Vector processors were common in the scientific computing area, where they formed the basis of most supercomputers through the 1980s and into the 1990s, but general increases in performance and processor design saw the near disappearance of the vector processor as a general-purpose CPU.

Today most commodity CPU designs include some vector processing instructions, typically known as SIMD (Single Instruction, Multiple Data), common examples include SSE and AltiVec. Modern video game consoles and consumer computer-graphics hardware rely heavily on vector processing in their architecture. In 2000, IBM, Toshiba and Sony collaborated to create a Cell processor, consisting of one scalar processor and eight vector processors, for the Sony PlayStation 3.

Processor Register (From http://en.wikipedia.org/wiki/Processor_register)

In computer architecture, a **processor register** is a small amount of storage available on the CPU whose contents can be accessed more quickly than storage available elsewhere. Most, but not all, modern computer architectures operate on the principle of moving data from main memory into registers, operating on them, then moving the result back into main memory-a so-called load-store architecture. A common property of computer programs is locality of reference, that is the same values are often accessed repeatedly; and holding these frequently used values in registers improves program execution performance.

Processor registers are at the top of the memory hierarchy, and provide the fastest way for a CPU to access data. The term is often used to refer only to the group of registers that are directly encoded as part of an instruction, as defined by the instruction set. More properly, these are called the "architectural registers". For instance, the x86 instruction set defines a set of eight 32-bit registers, but a CPU that implements the x86 instruction set will contain many more registers than just these eight.

Instruction Set (from http://en.wikipedia.org/wiki/Instruction_set) - READ THIS ONE AT LEAST

An instruction set, or instruction set architecture (ISA), is the part of the computer architecture related to programming, including the native data types, instructions, registers, addressing modes, memory architecture, interrupt and exception handling, and external I/O. An ISA includes a specification of the set of opcodes (machine language), the native commands implemented by a particular CPU design.

Instruction set architecture is distinguished from the microarchitecture, which is the set of processor design techniques used to implement the instruction set. Computers with different microarchitectures can share a common instruction set. For example, the Intel Pentium and the AMD Athlon implement nearly identical versions of the x86 instruction set, but have radically different internal designs.

SIMD (From <http://en.wikipedia.org/wiki/SIMD>)

In computing, **SIMD** (Single Instruction, Multiple Data) is a technique employed to achieve data level parallelism, like with vector or array processor. First made popular in large-scale supercomputers (contrary to MIMD parallelization), smaller-scale SIMD operations have now become widespread in personal computer hardware. Today the term is associated almost entirely with these smaller units.

SSE3 (From <http://en.wikipedia.org/wiki/SSE3>)

SSE3, also known by its Intel code name Prescott New Instructions (PNI), is the third iteration of the SSE instruction set for the IA-32 architecture. Intel introduced SSE3 in early 2004 with the Prescott revision of their Pentium 4 CPU. In April 2005, AMD introduced a subset of SSE3 in revision E (Venice and San Diego) of their Athlon 64 CPUs. The earlier SIMD instruction sets on the x86 platform, from oldest to newest, are MMX, 3DNow! (developed by AMD), SSE and SSE2.

Requirements for vMotion:

In order to successfully vMotion a guest from one host to another, the Source and Destination hosts must be:

- Part of the same datacenter in Virtual Center
- Connected to the same GbE network
- A dedicated GbE connection is recommended, though not required
- Connected to the same storage
- Destination host must have sufficient resources to host the new vm
- VM must not have any physical devices connected (CD, floppy, USB, etc)
- CPU models and feature sets must be compatible

CPU Compatibility

vMotion CPU compatibility requirements are covered in the following VMware KB articles:

- KB 1991 - Cpu Compatibility for Intel CPU
- KB 1992 - CPU Compatibility for AMD CPU
- KB 1993 - How to specify CPUID feature bit masks
- Note -KB 1993 articles still describe how to do masking / relaxation of CPU features, but this is unsupported UNSUPPORTED

Why might I not be able to vMotion between 2 servers?

On an ESX server, the CPU is a largely 'pass-through' device; that is to say, there is little to no virtualization of the CPU itself - the vast majority of CPU instructions from the guest are passed straight-through to the CPU.

Furthermore, if the CPU features supported on each host are too divergent, there is no way to successfully hand off the currently executing instructions to the new host, and a crash will result. This is obviously unacceptable, we therefore disable the ability to do so.

Furthermore, it should be noted that ALL hypervisors have the same issues with hot migration of a workload across CPUs.

What are some examples of why I might not be able to vMotion a guest?

- You can't run a vm with a 64-bit O/S on a host without a 64-bit CPU
- Applications requiring / leveraging SSE3 CPU instructions will crash a guest if migrated to a host without SSE3
- AMD and Intel have different instruction sets - examples:
 - 3DNow! and PREFETCH instruction sets only on AMD
 - MONITOR and MWAIT hyper threading instructions only on INTEL
- Different CPU models may have internal registers unique to that model

Which CPU features matter?

- CPU Vendor
- CPU Family (Pentium 4, Intel Core, AMD K8, AMD K10)
- Support for 64-bit guests
 - Intel: EM64T, VT
 - AMD: AMD64 on K8 revD, and newer CPUs
- SSE3 - Instructions for optimized signal processing, 3D graphics, hyperthreading
 - Intel Pentium 4, Prescott core (early 2004)
 - AMDK8 rev. E (April 2005)
- SSSE3 (AKA SSE4, Merom New Instructions) - Supplemental SSE3 instructions
 - Intel XEON 5100 (Core 2, Woodcrest) and newer (mid-2006)
- SSE4.1
 - Intel Core 2, Peryn core and newer
- FFXSR - optimization for FXSAVE / FXRSTOR instructions
 - AMD K8 rev D and newer
- RDTSCP - read serialized TSC pair
 - AMD K8 rev F and newer
- CMPXCHG16B
 - AMD K8 rev F and newer
 - Affects 64 bit guests only
 - All VMware supported Intel EM64T CPUs have this feature

Is there an easy way to tell which servers I can use vMotion with?

- Dell - http://www.dell.com/downloads/global/solutions/vmotion_compatibility_matix.pdf
- HP - <ftp://ftp.compaq.com/pub/products/servers/vmware/vmmotion-compatibility-matrix.pdf>

Attachments:

- [dell-vmotion_compatibility_matix.pdf](#) (39.6 K)
- [hp-vmotion-compatibility-matrix.pdf](#) (20.4 K)