

# Linux Timer Rate

---

## Introduction

A hardware timer is used by modern systems for a variety of fine-grained operations at the operating system level. VMware's virtualization platforms virtualize this timer in the kernel. Because the virtual timer provided to the VM is actually software, it is subject to the same resource restrictions as other processes. The busier the system the more the timer execution must contend with other hypervisor activities. There are two implications of this:

1. When the system is very busy, the software timer may not execute as regularly and virtual time may fall behind.
2. Depending on how frequently the OS wishes to be interrupted by the timer, the hypervisor must do different amount of work.

From [VMware KB article #1420](#).

Linux guest operating systems keep time by counting timer interrupts. Unpatched 2.4 and earlier kernels program the virtual system timer to request clock interrupts at 100Hz (100 interrupts per second). 2.6 kernels, on the other hand, request interrupts at 1000Hz - ten times as often. Some 2.4 kernels modified by distribution vendors to contain 2.6 features also request 1000Hz interrupts, or in some cases, interrupts at other rates, such as 512Hz.

Furthermore, an SMP-capable Linux kernel requests additional timer interrupts from the virtual local APIC timer. An SMP-capable kernel running on a one-CPU system generates twice as many total timer interrupts as the corresponding UP kernel, while such a kernel running on a two-CPU system requests three times as many. In general, an SMP-capable kernel running on  $<n>$  CPUs requests  $<n+1>$  times as many interrupts per second as a UP kernel. For example, an unmodified 2.6 Linux kernel running on a two-CPU virtual machine requests a total of 3000 clock interrupts per second.

When a guest asks for more than 1000 clock interrupts per second, it can be difficult for the virtual machine to keep up, especially if other applications are running on the host at the same time. This can cause the clock in the guest operating system to fall so far behind real time that it is unable to catch up. The overhead of delivering so many virtual clock interrupts can also hurt guest performance and increase host CPU consumption.

The amount of work required to manage the virtual timer is greatest with [Red Hat Enterprise Linux 5 \(RHEL5\)](#) SMP systems, which use a clock frequency of 1000 Hz and suffer from a multiplicative amount of work due to SMP support. For instance, the following table of timer interrupts was created on a 1000 Hz RHEL VM:

vCPU Count	Interrupts/sec
2	6,000
4	20,000
8	72,000

So, the amount of work that needs to be done by the hypervisor increases dramatically with the addition of vCPUs. In addition, decreasing the timer interrupt rate greatly decreases the work that needs to be done by the VMkernel to virtualize the timer. In RHEL 5.1, a Linux kernel that enables reducing the timer rate was included. By adding the

## Linux Timer Rate

parameter "divider=10" to the boot parameters, the amount of work required of the VMkernel to virtualize the timer goes down by an order of magnitude.