**虚拟化中的SR-IOV**

**虚拟化环境中有很多的硬件加速技术，这些技术标准来源于行业内的领导者或各种组织机构，但是在实际项目落地时又有哪些会被启用呢？哪些启用的功能带来了性能上明显的提升呢？那么这些加速技术如果不痛不痒的话那么它们的存在究竟意义有多大呢？**

**无论哪家解决方案，若想启用一些加速功能，势必需要硬件的支持，这就导致在一些项目前期的调研或者POC环境里不太容易实现，毕竟有些要求是十分昂贵和苛刻的，比如RDMA。相对于一些需要资金投入的技术来说，SRIOV无疑是比较亲民且易于实现的，今天就选它来一探究竟。本篇将全部采用微软Hyper-V环境进行说明**

**SR-IOV，即单根虚拟化。Intel在早期为了支持虚拟化环境，在CPU和PCI总线上提供了三层虚拟化技术，它们分别是：**

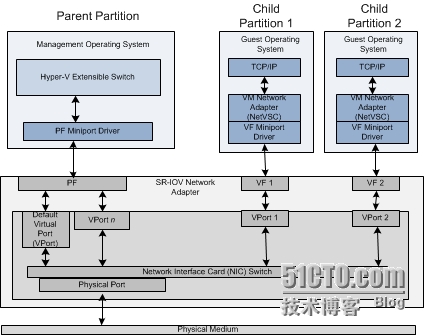
**（1）基于处理器的虚拟化技术VT-x**

**（2）基于PCI总线实现的IO虚拟化技术VT-d**

**（3）基于网络的虚拟化技术VT-c**

**从SRIOV的中文字面不难理解，它属于VT-d技术的一个分支，要实现SRIOV功能，前提条件就是你的网卡首先要支持SRIOV，你的主板要支持VT-d技术（支持VT-d自然也就支持SRIOV）**

**那么SRIOV究竟是干嘛用的呢？它能给虚拟化平台带来多么可观的性能提升呢？还是上一张架构图来看看吧：**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_1.jpg)**

**以上图为例逐个解释关键词：**

**1. PF就是物理网卡所支持的一项PCI功能，PF可以扩展出若干个VF**

**2. VF是支持SRIOV的物理网卡所虚拟出的一个“网卡”或者说虚出来的一个实例，它会以一个独立网卡的形式呈现出来，每一个VF有它自己独享的PCI配置区域，并且可能与其他VF共享着同一个物理资源（公用同一个物理网口）**

**3. PF miniport driver即PF驱动是工作于Hyper-V虚拟化平台父区域的，并在VF之前最先加载**

**4. VF miniport driver即VF驱动是工作于Hyper-V虚拟化平台子区域的，即GuestOS；需要注意的是，VF及PF之间是隔离的，任何经由VF驱动或所执行的结果都不会影响到其他的VF或PF**

**5. Network Interface Card即物理网卡，在启用SR-IOV之后会生成若干vport，物理NIC所要做的就是转发physical port与vport之间的流量**

**6. physical port顾名思义就是物理网口，在SR-IOV场景中physical port充当一个面向对外的网络媒介**

**7. VPort是个抽象出来的接口，类似于物理网口，它们被映射给每一个VF或者PF，供parentOS或guestOS来使用**

**通过以上架构的描述就可以看出，启用SR-IOV之后，物理NIC将通过VF与虚拟机（VF driver）进行数据交互，反之亦然。那么这样一来即可跳过中间的虚拟化堆栈（即VMM层），以达到近乎于纯物理环境的性能；这一点也是SRIOV最大的价值所在，它有别于以往虚拟机通过仿真设备和虚拟化层进行流量传递的情况，那么究竟SR-IOV与传统环境相比能提升多少，来做个实验：**

**宿主机OS：windows server 2012R2**

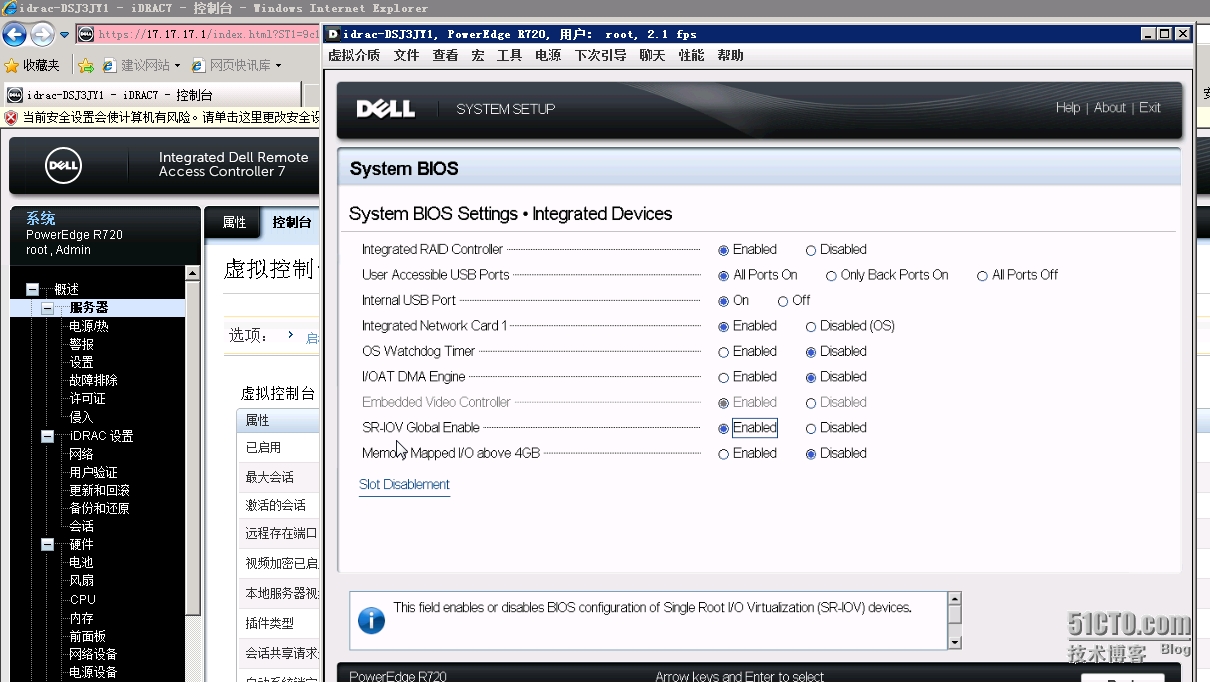
**虚拟机OS：windows server 2012R2**

**服务器型号：DELL R720**

**网卡：intel x520 series**

**######################################################################################**

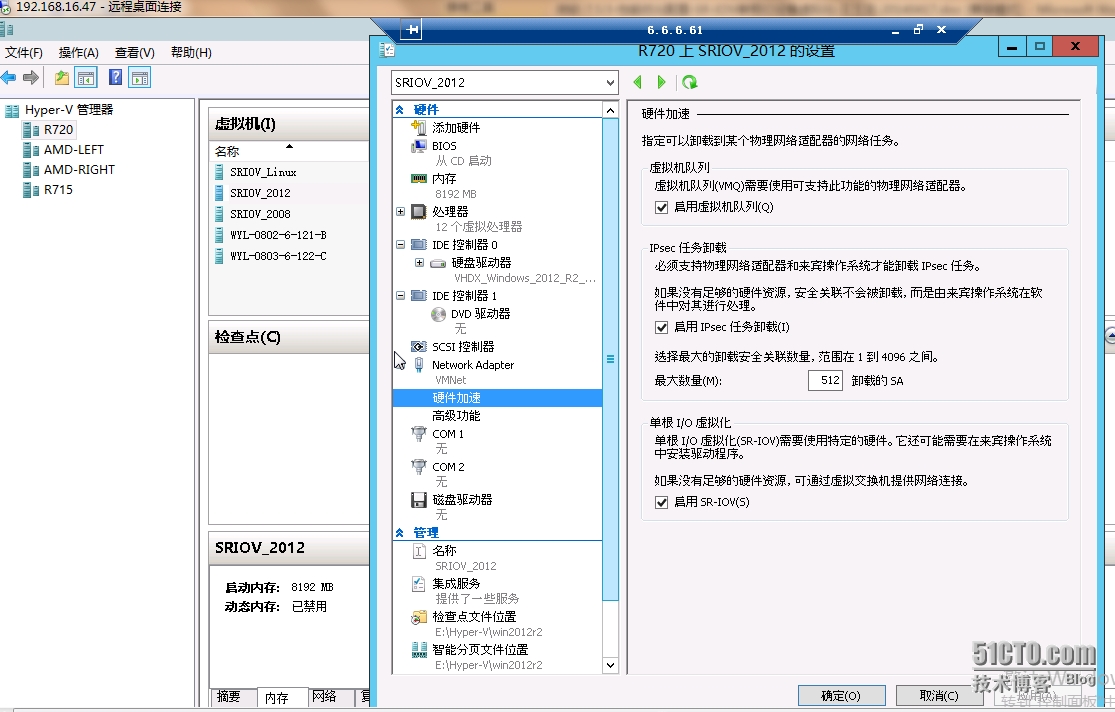
**首先在服务器BIOS设置中将SRIOV功能开启**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_2.jpg)**

**物理机确认开启了SRIOV功能之后，接下来在操作系统层面操作，首先Hyper-V若要使用SRIOV，有两处需要修改，一个是虚拟交换机，如下图确认在创建虚拟交换机时开启了SRIOV（单根虚拟化），需要注意的是虚拟交换机一旦创建后，SRIOV功能无法在修改，也就是说你要是忘了开启那对不起，麻烦您删了重来**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_3.jpg)**

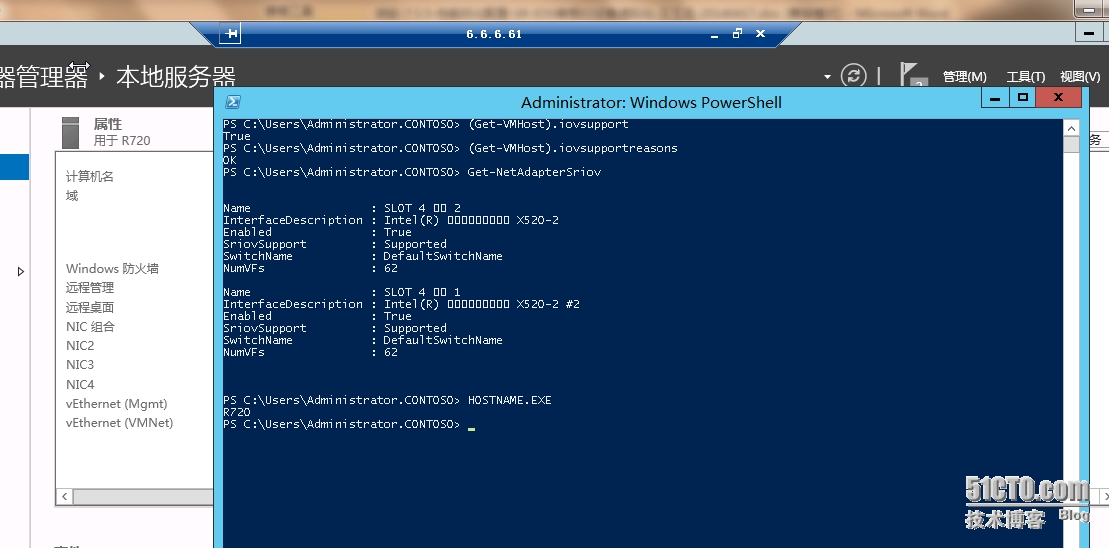
**虚拟交换机启用SRIOV之后，就要在我测试的虚拟机上操作了，在虚拟机的vNIC（虚拟网卡）上开启SRIOV，如下图所示，这里是可以随时开关的**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_4.jpg)**

**确认了上面的操作之后，通过powershell可以进一步确认系统是否识别了我的设置，在当前宿主机执行（get-vmhost）.iovsupport或iovsupportreasons来查看返回结果，有关powershell中对象的属性可以通过管道符“|gm”来查看**

**另外如下图所示，通过get-netadaptersriov来查看当前主机上支持sriov的物理网卡有哪些，并且从返回结果来看，我的x520-2网卡最多支持62的vf。**

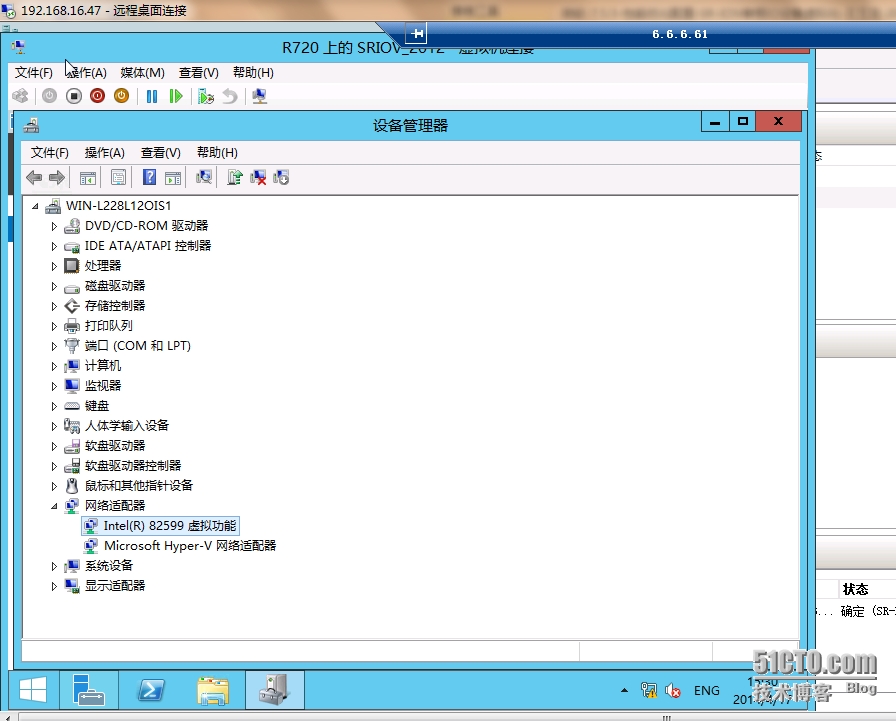
**顺带提一句Peripheral Component Interconnect Special Interest Group(外围部件互连专业组)，简称PCISIG，这个组织定义了每个设备最多可支持的vf数量为256个**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_5.jpg)**

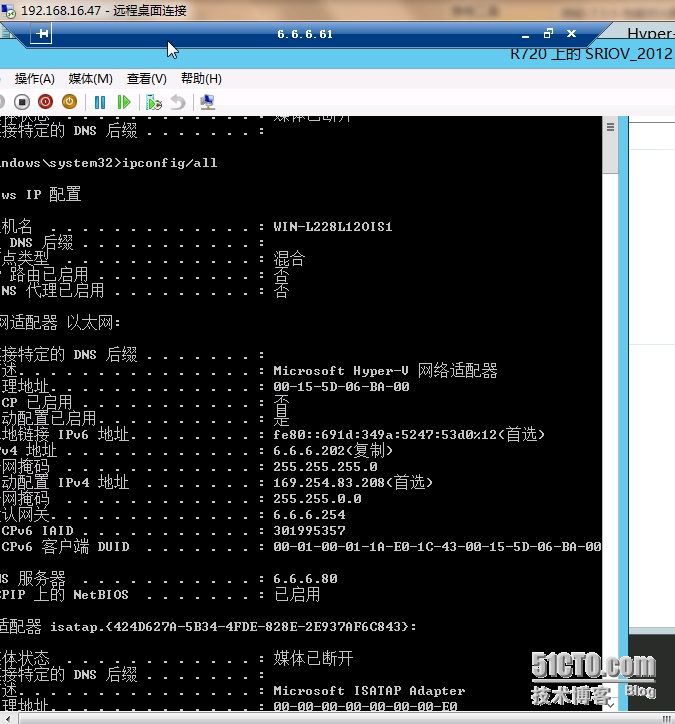
**在正确且成功开启了sriov功能之后，我启动这台测试虚拟SRIOV\_2012，可以看到hyper-v下方显示当前SRIOV是活动状态，但奇怪的是我发现有两个IP。。怎么回事呢？**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_6.jpg)**

**进入到虚拟机系统里来看设备管理器，发现多了一块网卡，叫做“Intel(R）82599虚拟功能”，其实我这块Intel x520 series网卡是基于Intel 82599控制芯片的，后面的虚拟功能翻译过来就是virtual function，也就是虚出来的一个VF，它以一块虚拟网卡的形式呈现在虚机操作系统里了，因此我刚才看到了两个IP地址**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_7.jpg)**

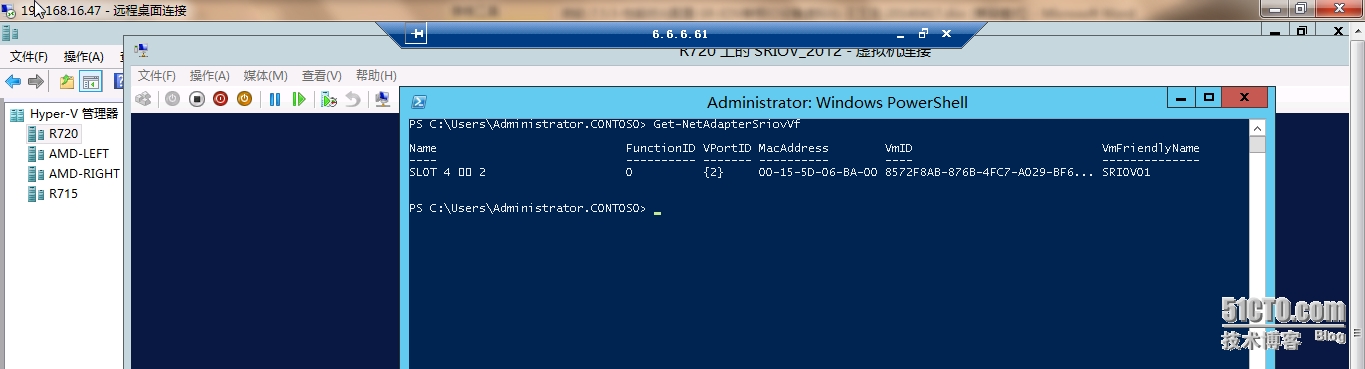
**这里可能有个小bug，就是我需要重新配一次IP，这样这台虚拟机才不会出现两个IP地址，如下图目前这个正常的测试IP显示的是（复制）**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_8.jpg)**

**重新输一遍之后就恢复正常了，也就是说原先的IP地址没有直接映射到我的VF上面，下图显示当前IP已经恢复正常了，只有一个6.6.6.0的IP**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_9.jpg)**

**同样通过powershell命令“get-netadaptersriovvf”可以看到当前生成的VF信息**

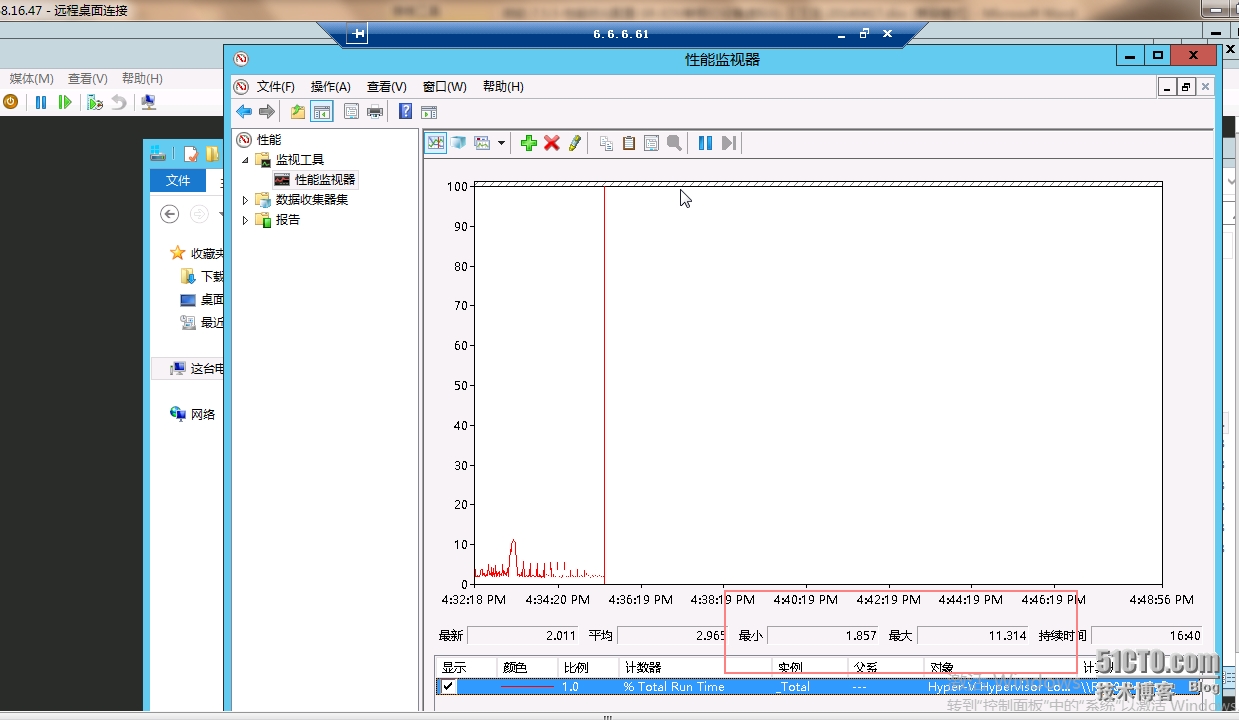
**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_10.jpg)**

**######################################################################################**

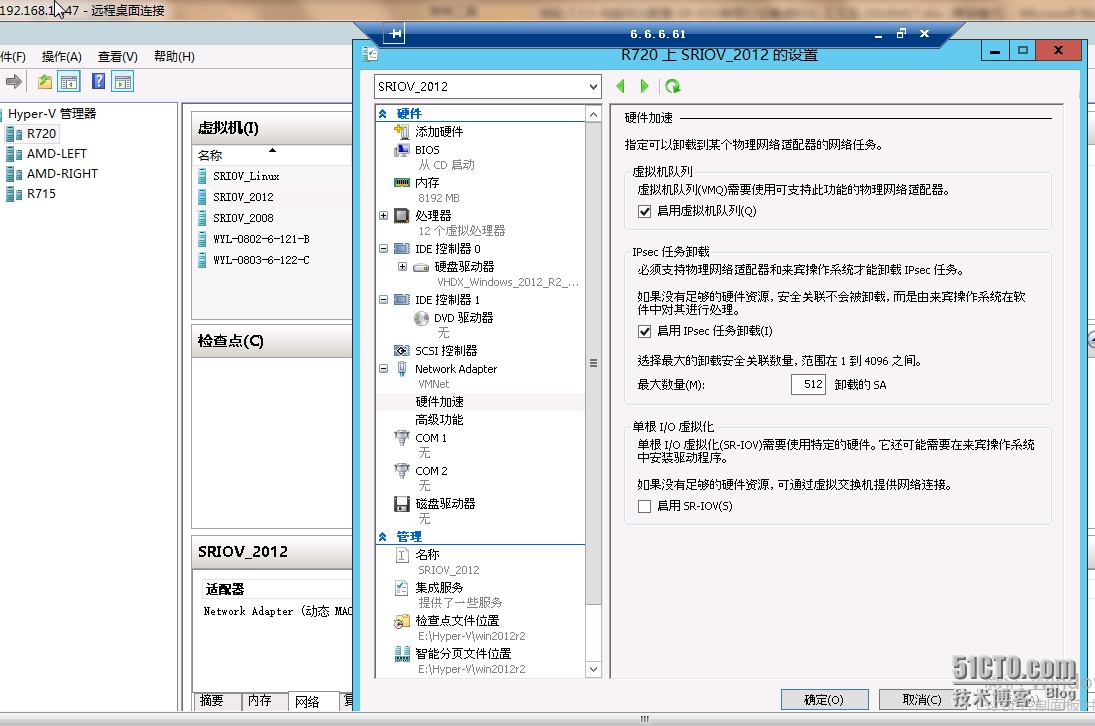
**下面开始一个拷贝测试，通过网络传输一个iso文件，在启用SRIOV的情况下，传输速度大约460MB/S**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_11.jpg)**

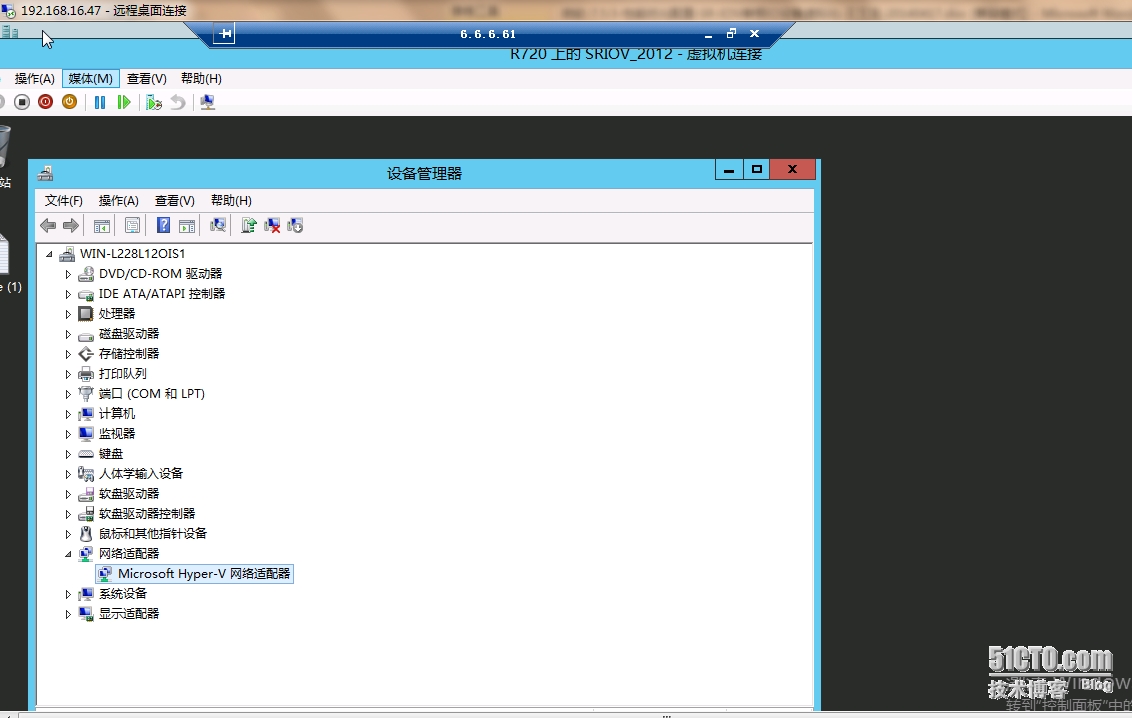
**在传输文件同时，我使用工具（burnintest）对虚拟机CPU进行加压，以尽量模拟实际情况，观察结果如下图，通过性能监视器看到CPU使用率最小值不到2%，最大值11%多，相差约9.5%**

****

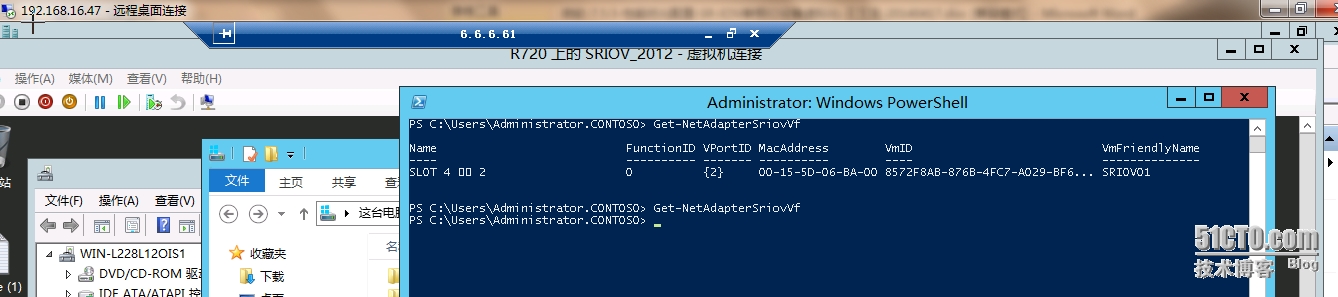
**接着关掉这个虚拟机的SRIOV功能**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_13.jpg)**

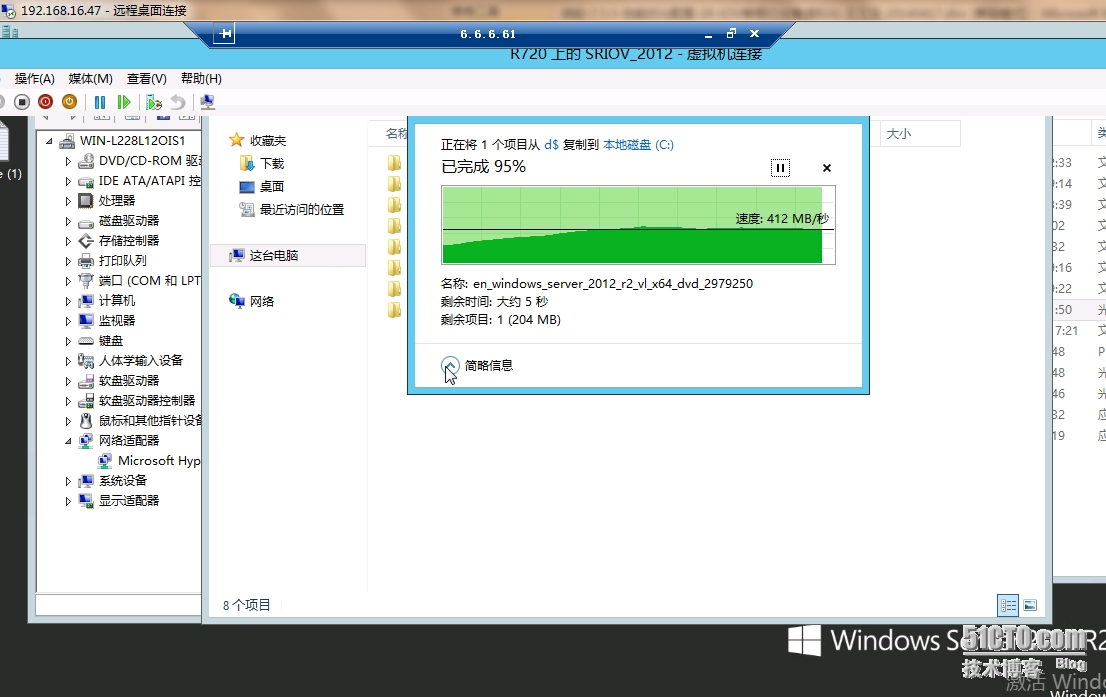
**可以看到VF没有了，如下图**

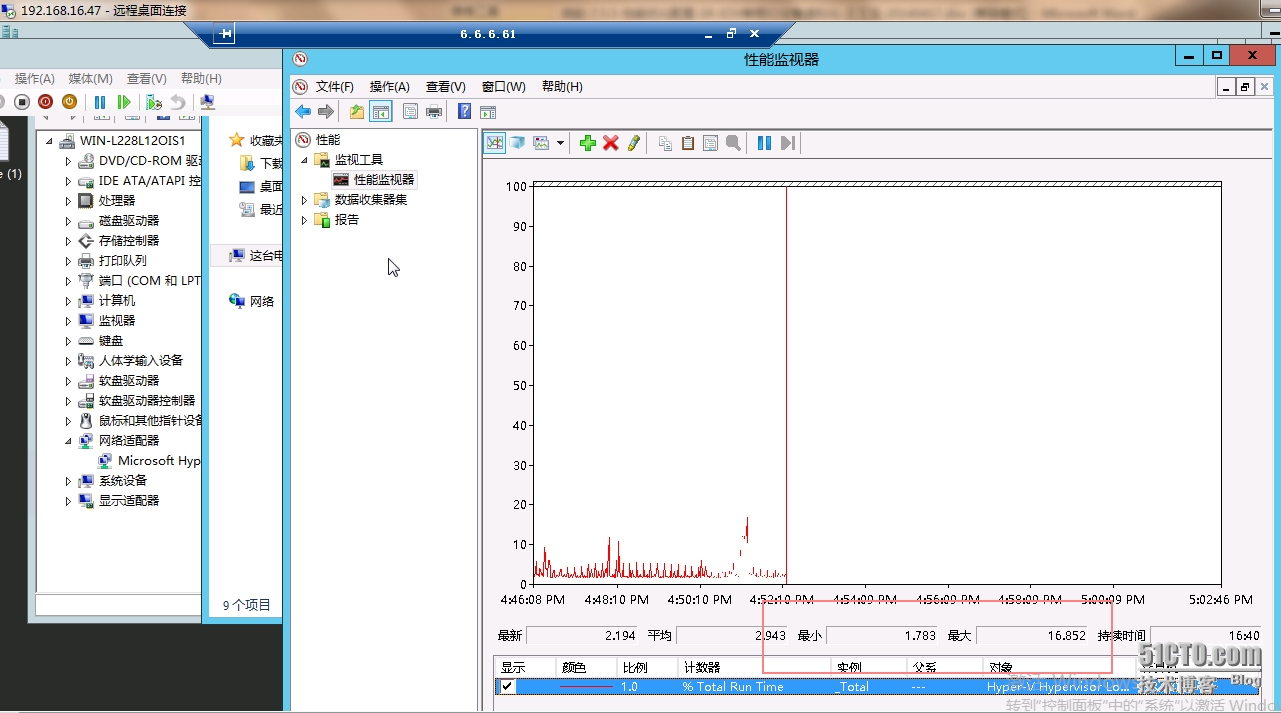
**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_14.jpg)**

**通过powershell确认VF的确离我们远去了~**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_15.jpg)**

**同样再通过网络拷贝一次文件，依旧是iso文件（这里不用考虑缓存因素，我在每次拷贝之前都会进行一些复制操作以便尽量充满缓存），传输速率大概在410MB/S左右**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_16.jpg)  
 同样传输期间对虚拟机CPU进行加压，观察性能监视器结果，CPU负载最小值不到2%，最大值接近17%，相差约15%**

**[](http://image76.360doc.com/DownloadImg/2014/07/2219/43637752_17.jpg)**

**综合上述情况来看，对比SRIOV功能开启与关闭，拷贝同样iso文件以及相同的CPU加压方式，结果如下：**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **开启SRIOV** | **关闭SRIOV** | **差值** |
| **CPU使用率** | **9.5%** | **15%** | **5.5%** |
| **传输速率** | **460MB/S** | **410MB/S** | **50MB/S** |

**########################################################################################**

**通过上面的测试可以看出在SRIOV开启或关闭的不同情况下，对比还是有一定效果的，我的测试环境还是不够严谨的，因为在实际生产环境中还要考虑诸多因素，例如磁盘的IO，虚拟机的CPU配置等等情况，但是即便比较粗陋，这个数据还是具有一定参考价值的，将近6%左右的CPU负载以及相差50MB/S的速率我想对于任何一个有大批量并发请求的虚拟化平台用户来讲都是相当可观的，所以说SRIOV对于当前私有云用户来讲还是很有价值的。**

**说起SRIOV，其实还有一个“兄弟”不得不提，它就是VMQ（virtual machine queue），我的个人理解是SRIOV更适合于出流量，它优化了数据在虚拟机与物理机之间交互的路径，简化了这个过程，使得性能近乎于纯物理环境，但是对于入流量来讲，大部分处理工作还是交由CPU来做，外部流量进来之后，系统需要确认哪些数据发送给哪些虚拟机，这些路由和分发工作在大访问量的场景下是十分费力的，VMQ功能恰恰解决了这个问题，同样是需要物理网卡支持，将入流量提前进行路由并列队，系统直接分配给需要接收的虚拟机，提高了性能，但是VMQ基于“先到先得”的机制，建议仅对那些有大批量访问请求的虚拟机使用，以免那些只有很少入流量的机器影响了关键业务。**

**关于虚拟化的硬件加速功能还有很多，如果以后有条件了会一并奉上与大家分享。**