Optimizing Network Throughput for Virtual Machines

S. McClellan and B. Guilfoyle
VM Host Architecture

**ZNYX ZX2000:**
- Standards-based
- 1U, 2U, 5U blade server systems
- 10G embedded switch/routers
- All redundant subsystems

**Virtualization:**
- Software-enabled network compute platform for SDN
- Designed and optimized for virtualization
- Integrated environment for out-of-the-box VM deployment
Two Topics:

- Network throughput to VMs
  - Effect of NIC-based offload/acceleration
- Interesting anomalies in VM performance
  - Effect of stack-resident timers
Topic 1: VM congestion .. Solved?

- VM’s create bottlenecks inside the system
  - Single Ethernet NIC, shared
  - Software-based “switch” via Hypervisor

- Issue noted previously (Intel pre-2007)
  - Result: Virtual Machine Device Queues (VMDq)
  - Rudimentary L2 “switch/classifier” inside the Ethernet NIC
    - Managed/configured by the Hypervisor
    - Hardware-accelerated frame processing

- Current generation uses “SR-IOV”
  - PCI-SIG standardized I/O virtualization

Results using Windows Server 2003, VMWare ESX pre-release.

Questions:
• How does this stack up today, using Linux KVM and SR-IOV?
• How do results fare with various combinations of VMs, Memory, Threads, CPU affinities?
• Are there any statistical inferences?
CPUs per VM

1,2,4 CPUs per VM:
- High CPU utilization
- Erratic performance

8,16,32 CPUs per VM:
- Low CPU utilization
- Max-performance for 2,4,8 threads
- Half-performance otherwise

Santa Clara, CA  USA
April 2013
Affinity (#1): Use Cores/Threads

Conditions:
- 1 VM
- 8 cores per CPU
- 1 CPU per VM
- 1 Thread of data

Result:
- Poor link utilization (~50%)
- No real “better” core affinity
Affinity (#2): Choose Cores Wisely

Conditions:
- 1 VM
- 1, 2, 4 Threads
- 2 CPUs per VM
- Various cores

Result:
- VMs should use at least 2 cores on the same CPU
- VMs should not use cores on different CPUs
- VMs should not use core & hyperthread on the same CPU
Conditions:
(optimum per previous data)
- 2 CPUs per VM
- Separate cores on same die
- 4 threads per test

Result:
- Complete link utilization (aggregate, all VMs)
- Roughly linear fairness across VMs (log scale)
- 2,4 VMs with 2 CPUs per VM is “sweet spot” (2-socket, 8-core)
Benefit of HW Acceleration

Result:
• 1 VM with 8,16,32 CPUs (2-socket, 8-core system)
• Complete link utilization for 2,4,8 threads (99% confidence intervals shown)
• Note differences in confidence intervals between cases
Topic 2: VM network anomalies

- IP stacks have timers that can affect throughput
- These timers are set up for “last century” networks
  - Slow or high-latency links
  - Long time-of-flight configurations
  - IETF specifications are “tuned” this way
- Virtual Machines exist in a completely different world
  - Short, fat virtual links
  - Extremely fast time-of-flight
IP Retransmission Timers 101(a)

- Send a packet, expect an ACK
  - No ACK = Retransmission (bad)
- How long to wait?
  - Round-trip Time Out (RTO)
  - Worst-case wait-time for this link
- How to figure RTO value?
  - Estimate via “Jacobson Algorithm”
  - Based on Round-Trip Time (RTT) from prior ACKs
  - Uses a “theoretically good” formulation
    - Chebyshev bound and parameter estimation
IP Retransmission Timers 101(b)

- Jacobson Algorithm has “issues”
  - Extreme response to sudden changes in RTT
  - Slow convergence after bad responses
  - Improper bias for stable links

- Jacobson Algorithm isn’t the only “issue”
  - Secondary processing often trumps Jacobson
  - Algorithm structure and parameters are essentially tuned for old networks (long, slow, high latency)
    - Modern networks are fast, with relatively low RTT
    - Virtualized networks are short and fast, with extremely low, stable RTT … except in certain cases …
Example: Jacobson

This data is replayed from probes on a live virtual network

**Problems:**
- Fast links (extremely low RTT)
- Stable links (almost no variation in RTT)
- Sporadic timeouts … with stable gaps
Example: VM performance

Note:
- RTT is almost zero
- Areas of “high” variance are still very low RTT compared to static boundaries (1000)
- Many induced timeouts if static minimum is disabled
Conclusions

Observations:
• Data throughput to VMs can be significantly enhanced by hardware-assist (SR-IOV, etc.)
  – There are “sweet spots” depending on HW architecture
  – VM configuration can leverage sweet spots
• IP stacks are not well-prepared for virtualization

Solutions:
• VM deployment requires a lot of expertise in HW, SW, and system configuration
• ZNYX Networks develops integrated, optimized solutions based on industry standards