XenLoop
A Transparent High Performance Inter-VM Network Loopback

Jian Wang*, Kwame-Lante Wright*, Kartik Gopalan*
*Binghamton University (State University of New York)
+The Cooper Union
Talk Outline

- Motivation
  - What worry about Inter-VM (Virtual Machine) communication?
- Xen and Xen networking subsystem background
- XenLoop – What does it do?
- Why develop XenLoop?
- How does it work?
- How well does it work?
- Summary and future work
Motivation: Why Inter-VM Communication?

- Virtualization technology is mainly focused on building the *isolation barrier* between co-located VMs.
- However, applications often wish to *talk across this isolation barrier*.
- E.g. High performance grid apps, web services, virtual network appliances, transaction processing, graphics rendering.
Why not just use TCP or UDP?

- Transparent to applications BUT
- High communication overhead between co-located VMs

<table>
<thead>
<tr>
<th></th>
<th>Flood Ping RTT (Microsecs)</th>
<th>TCP Bandwidth (Mbps)</th>
<th>UDP Bandwidth (Mbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Native Loopback</td>
<td>6</td>
<td>4666</td>
<td>4928</td>
</tr>
<tr>
<td>Xen Inter-VM</td>
<td>140</td>
<td>2656</td>
<td>707</td>
</tr>
<tr>
<td>Inter-Machine</td>
<td>101</td>
<td>941</td>
<td>710</td>
</tr>
</tbody>
</table>
Xen and Xen Networking Subsystem

Communication data path between co-located VMs
- Packet transmission via page-sharing
- Packet reception via page-flipping or copying
XenLoop Overview

- Enables direct traffic exchange between co-located VMs
  - Via inter-domain shared memory FIFO channels
  - No Dom0 intervention in data path

- Transparency for Applications and Libraries
  - No code modifications, recompilation, or re-linking
  - Operates transparently beneath existing socket interfaces and libraries.

- Kernel Transparency
  - No changes to either the guest OS code or hypervisor
  - Self-contained kernel module
XenLoop Overview (contd.)

- Automatic discovery of co-located VMs
  - No administrator involvement

- On-the-fly setup/teardown XenLoop channels
  - Only when traffic is exchanged

- Migration transparency
  - Migrate without disrupting ongoing network communications
  - Switch between the standard network path and the XenLoop channel.
### Why Develop XenLoop?

<table>
<thead>
<tr>
<th>Feature</th>
<th>XenSocket</th>
<th>XWay</th>
<th>IVC</th>
<th>MMNet</th>
<th>XenLoop</th>
</tr>
</thead>
<tbody>
<tr>
<td>User Transparent</td>
<td>X</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Kernel Transparent</td>
<td>✓</td>
<td>X</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Transparent VM Migration</td>
<td>X</td>
<td>X</td>
<td>Not fully transparent</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Standard protocol support</td>
<td>X</td>
<td>Only TCP</td>
<td>Only MPI or app protocols</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Auto VM Discovery &amp; Conn. Setup</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Complete memory isolation</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>X</td>
<td>✓</td>
</tr>
<tr>
<td>Location in Software stack</td>
<td>Below socket layer</td>
<td>Below socket layer</td>
<td>User Library + syscalls</td>
<td>Below IP layer</td>
<td>Below IP layer</td>
</tr>
<tr>
<td>Copying Overhead</td>
<td>2 copies</td>
<td>2 copies</td>
<td>2 copies</td>
<td>2 copies</td>
<td>4 copies at present</td>
</tr>
</tbody>
</table>
Netfilter hook to capture and examine outgoing packets.

One-bit bidirectional channel to notify the other endpoint that data is available in FIFO.
XenLoop Channel Setup/Teardown

- Happens on-demand
  - Upon first packet exchange b/w co-located VMs
  - No packet exchange ➔ no FIFO is set up.

- Endpoint identifiers passed “out-of-band” via netfront.

- Channel torn down automatically when one of the guest migrates away, shuts down, or suspends.
  - Traffic switches back to Netfront-Netback
Sender intercepts packets destined to co-located guests below the IP layer
- Using netfilter hook (post-routing)

If packet is headed to co-located guest, then the sender copies intercepted packets into FIFO
- Packets headed out of the physical box sent via netfront-netback

Receiver copies packets out from FIFO and hands them to IP layer.

Why two copies?
- Because cross-domain page-txfer/sharing is more expensive.
- Also maintains a simple design
Discovering Co-located Guests

- Active guest creates a “XenLoop” entry in \textit{XenStore} advertising it’s identity.
  - XenStore = Key-Value Database to store guest properties.
  - Entry deleted when guest shuts down, dies, or migrates away.

- A discovery module in Dom0 periodically collects active guest information from XenStore
  - Advertises presence to other co-located guests via out-of-band network packets.
XenLoop Performance

Netperf UDP_STREAM

Throughput (Mbps)

Message size (log2(bytes))

- netback/netfront
- native loopback
- native inter-machine
- xenloop
XenLoop Performance (contd.)

Netpipe-mpich Bandwidth

Throughput (Mbps)

Data size (bytes)
XenLoop Performance (contd.)

Netpipe-mpich Latency

![Graph showing latency vs. data size for different communication methods: netback/netfront, native loopback, native inter-machine, xenloop. The x-axis represents data size in bytes, and the y-axis represents latency in seconds. The graph illustrates the performance differences between these methods at various data sizes.]
XenLoop Performance (contd.)

Migration Transparency

- Colocated VMs
- Separated VMs
- Separated again
Future Work

- Compatibility with routed-mode Xen setup
  - Implemented. Under testing.

- Packet interception b/w socket and transport layers
  - Do this without changing the kernel.
  - Will reduce 4 copies to 2 (as others), significantly improving bandwidth performance.

- XenLoop for Windows guest?
  - Windows ↔ Linux XenLoop Channel
  - XenLoop architecture mostly OS agnostic.
XenLoop publicly available

http://osnet.cs.binghamton.edu/projects/xenloop.html