A Remote I/O Solution for the Cloud

Cynthia Taylor, Joseph Pasquale
University of California, San Diego

CLOUD 2012
June 28, 2012
Motivation
Why Remote I/O?
Transparency
Transformation
No Single Solution

• Different devices

• Different applications

• Different network conditions

• Different optimal solutions
Architecture
Diverse Beneficiaries Require Easy Customization and Extensibility

• Device designers

• Application designers

• Users
Networked Device Driver Abstraction for Transparency
Modular Architecture
Need to Connect Device and Application
Device Module

client

device
module

device
driver

device

server

application
Network Modules
Application Module
Need to Add Data Processing for Network

• Averaging
• Bundling
• Buffering
• Compressing
• Discarding
• Encrypting
• Multiplexing
• Synchronizing
Transformation Module Pairs
Example Module Pairs

• Compression/Decompression

• Bundling/Unbundling

• Encryption/Decryption
Compression
Composability
Summary

• Device driver abstraction supports transparency

• Modular design supports customization, extension

• Transformation module pairs allow processing of data
Implementation
Implementation Goals

• Efficiency

• Ease of implementation

• Leveraging existing mechanisms
Kernel vs user space

• Insecure/buggy code is dangerous to run in kernel

• Allows developers to use any existing tools/libraries

• Copies between process boundaries must go through kernel
Run Predominately in Userspace to Support Extensibility
Modules as Processes Support Customization

- Can compose at run-time
- Scheduled by the kernel
- Automatically block on I/O
- Separate address spaces
Pipes Copy Between Processes
Implementation Summary

- Implemented at user-level whenever possible to support **extensibility**

- Modules are implemented as processes to support **customization**

- Pipes implementation for **ease of implementation**
Performance
Test bed

- Dell Optiplex 320
- Intel Celeron
- 133 Mhz FSB Clock
- Ping time of .12 ms between machines
- 11.3 MB/s throughput
Space Navigator
End-to-End Time of the Space Navigator
Overhead of Space Navigator Driver

36 usec  380 usec  17 usec
Comparing to VNC
Networked Device Driver vs VNC: Frames per Second

![Graph showing comparison between Networked Device Driver (NDD) and VNC for frames per second vs pixels per frame. The graph illustrates the performance of both methods across different pixel densities, with VNC showing a consistent advantage in frames per second.]
Adding Compression
Networked Device Driver vs VNC: Adding Compression
Summary

• Overhead is order of magnitude less than speed of network

• Performance similar to that of VNC
Conclusion
Summary

- System for I/O over network
- Application sees as driver
- Supports Transformation Modules
- Reasonable performance overhead