Rekindling Network Protocol Innovation with User-Level Stacks

Felipe Huici (NEC Europe)
Michio Honda (NetApp),
Joao Taveira Araujo (UCL),
Luigi Rizzo (Pisa University),
Costin Raiciu (Universitea Buchalest)

https://github.com/cnplab/multistack
Motivation

• Extending layer 4 functionality could address a lot of problems
  – Increased performance
    • MPTCP, WindowScale, FastOpen, TLP, PRR
  – Ubiquitous encryption
    • TcpCrypt
Motivation

• Extending layer 4 functionality could address a lot of problems
  – Increased performance
    • MPTCP, WindowScale, FastOpen, TLP, PRR
  – Ubiquitous encryption
    • TcpCrypt

Is it really possible to deploy layer 4 extensions?
Motivation

• Extending layer 4 functionality could address a lot of problems
  – Increased performance
    • MPTCP, WindowScale, FastOpen, TLP, PRR
  – Ubiquitous encryption
    • TcpCrypt

Is it really possible to deploy layer 4 extensions?

• Networks still accommodate TCP extensions
  – 86% of the paths are usable for well-designed TCP extensions despite middleboxes
Protocol Stacks in End Systems: The Theory

- OSes implement stacks
  - High performance
  - Isolation between applications
  - Socket APIs
- New OS versions adopt new protocols/extensions
Extending Protocol Stacks: The Reality

- OSes' release cycle is slow
- Support in the newest OS version does not imply deployment
  - Stakeholders are reluctant to upgrade their OS
  - Often new features, even if available, must be explicitly enabled
How Long does Deployment Take?

- **Windows**: SACK is default since Windows 2000. WS and TS implemented in Windows 2000 but enabled as default since Windows Vista (2009)

- **Linux**: SACK/TS on by default since 1999, WS since 2004

* Traffic trace from single transit link in Japan (MAWI)
• To ease upgrade, we need to **move protocol stacks up into user-space**
• To ease upgrade, we need to **move protocol stacks up into user-space**
• **Problem: no practical way to do this, we need:**
  – Isolation between applications
  – Support for legacy applications and the OS’s stack
  – High performance
MultiStack: Operating System Support for User-space Stacks

Netmap API

Multiplex / Demultiplex packets (3-tuple)

TCP port 80
UDP port 53
TCP port 22

User

Kernel

legacy apps
Socket API
OS's stack

Virtual ports

MultiStack

NIC

App 1
Stack 1

App N
Stack N

(TCP port 80)
(UDP port 53)
(TCP port 22)
MultiStack: Operating System Support for User-space Stacks

Apps/stacks register desired 3-tuple with the MultiStack kernel module
MultiStack: operating system support for user-space stacks

- Support for multiple stacks (including OS's stack)
- Namespace isolation based on traditional 3-tuple
- Very high performance
- Runs on FreeBSD and Linux (and it's open source!)
**MultiStack: operating system support for user-space stacks**

- Support for multiple stacks (including OS’s stack)
- Namespace isolation based on traditional 3-tuple
- Very high performance
- Runs on FreeBSD and Linux (and it’s open source!)
MultiStack: operating system support for user-space stacks

- Support for multiple stacks (including OS’s stack)
- Namespace isolation based on traditional 3-tuple
- Very high performance
- Runs on FreeBSD and Linux (and it's open source!)
MultiStack: operating system support for user-space stacks

- Support for multiple stacks (including OS’s stack)
- Namespace isolation based on traditional 3-tuple
- Very high performance
- Runs on FreeBSD and Linux (and it's open source!)
Multistack Base Performance (Tx)

pktgen
stack

core 0...n

3-tuple filter

NIC
Multistack Base Performance (Tx)

- App creates every packet from scratch, and sends it to the kernel
- Multistack validates the source 3-tuple of every packet, and copies the packet to the NIC’s TX buffer
Multistack Base Performance (Tx)

- App creates every packet from scratch, and sends it to the kernel
- Multistack validates the source 3-tuple of every packet, and copies the packet to the NIC's TX buffer
Multistack Base Performance (Rx)
Multistack Base Performance (Rx)

- Multistack receives a packet
- It identifies destination 3-tuple of the packet
- It delivers the packet to the corresponding app/stack
Multistack Base Performance (Rx)

- Multistack receives a packet
- It identifies destination 3-tuple of the packet
- It delivers the packet to the corresponding app/stack
Many Apps/Stacks (Tx)

pktgen
stack

pktgen
stack

pktgen
stack

...3-tuple filter

NIC

Socket API

(TCP port 80)

(UDP port 53)

(TCP port 22)
Many Apps/Stacks (Tx)

- App creates every packet from scratch, and sends it to the kernel
- Multistack validates the source 3-tuple of every packet, and copies the packet to the NIC’s TX buffer
• App creates every packet from scratch, and sends it to the kernel
• Multistack validates the source 3-tuple of every packet, and copies the packet to the NIC’s TX buffer
Many Apps/Stacks (Rx)
Many Apps/Stacks (Rx)

- Multistack receives a packet
- It identifies destination 3-tuple of the packet
- It delivers the packet to the corresponding app/stack
Many Apps/Stacks (Rx)

- Multistack receives a packet
- It identifies destination 3-tuple of the packet
- It delivers the packet to the corresponding app/stack
Many Apps/Stacks (Rx)

- Multistack receives a packet
- It identifies destination 3-tuple of the packet
- It delivers the packet to the corresponding app/stack

- A bit lower performance on many ports is due to the reduced number of packets taken in a single systemcall
Multistack Performance Summary

- 4 Gbps for 64 byte packets with a single CPU core
- 10 Gbps for 64 byte packets with two CPU cores
- 10 Gbps for 256 byte packets with a single CPU core
Performance with User-Level Stacks

- A simple HTTP server on top of our work-in-progress user-space TCP (UTCP)
- The same app running on top of OS’s TCP

Client establishes a TCP connection, and sends HTTP GET
Server replies with HTTP OK (1-32KB)
Single TCP connection is used for a single HTTP transaction
Conclusion

• Multi-stack: OS support for user-space stacks to rekindle widespread, timely deployment of new protocols/extensions

Try it out!  https://github.com/cnplab/multistack