Network stack personality in Android phone

Cristina Opriceana, **Hajime Tazaki** (IIJ Research Lab.)

Linux netdev 2.2, Seoul, Korea

08 Nov. 2017

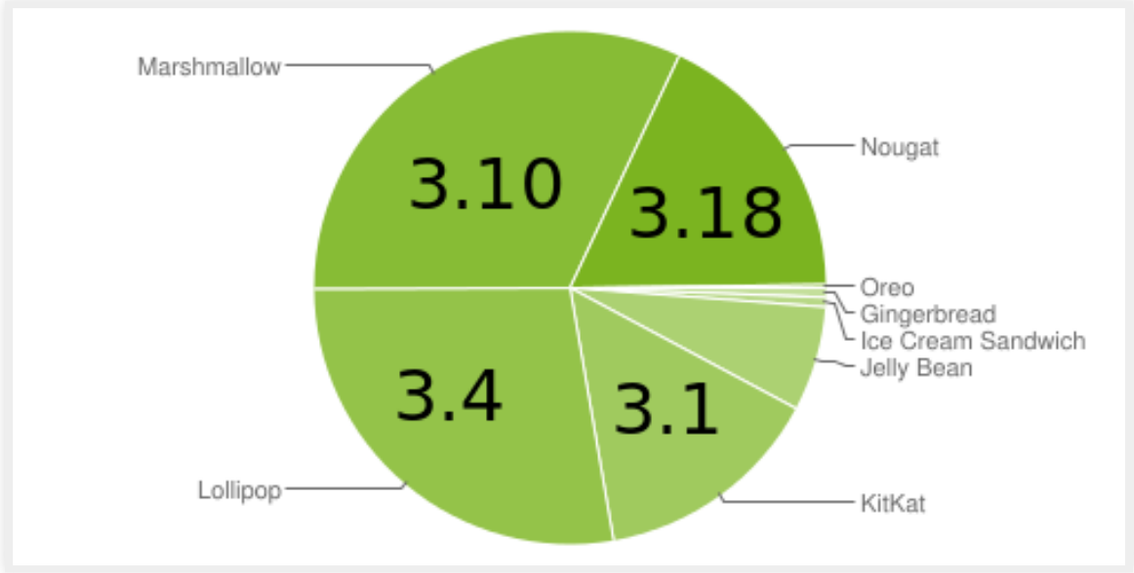
Librarified Linux taLks (LLL)

- Userspace network stack (NUSE) in general (netdev0.1)
- kernel CI with libos and ns-3 (netdev1.1)
- Network performance improvement of LKL (netdev1.2, by Jerry Chu)
- How bad/good with LKL and hrtimer (BBR) (netdev2.1)
- Updating Android network stack (netdev2.2)

Android

a platform of billions devices

- billions installed Linux kernel
- Questions
  - When our upstreamed code available ?
  - What if I come up with a great protocol ?



https://developer.android.com/about/dashboards/index.html

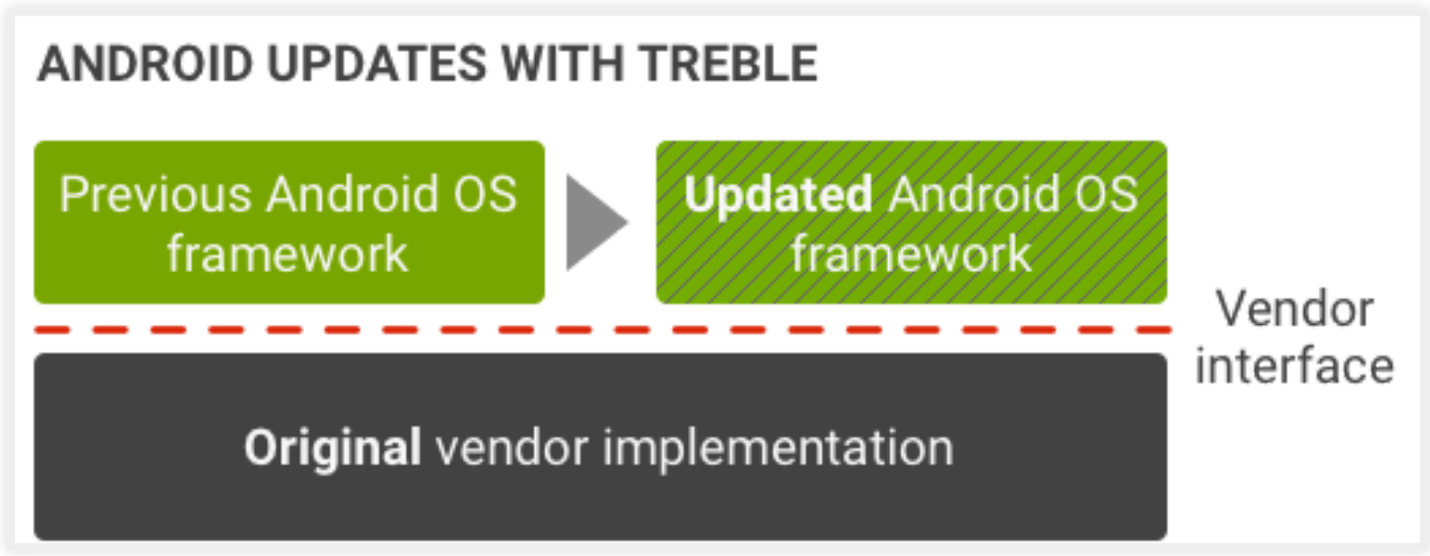
Android (cont'd)

- When our upstreamed code available ?
  - wait until base kernel is upgraded
  - backport specific function
- What if I come up with a great protocol ?
  - craft your own kernel and put into your image

Long delivery to all billions devices

Approaches to alleviate the issue

- Virtualization (KVM on Android)
  - Overhead isn't negligible to embedded devices
- Project Treble (since Android O)
  - More modular platform implementation
- Fushia
  - Rewrite OS from scratch
- QUIC (transport over UDP)
  - Rewrite transport protocols on UDP



https://source.android.com/devices/architecture/treble

https://source.android.com/devices/architecture/treble

An alternate approach

- network stack personality
  - use own network stack implemented in userspace
  - no need to replace host kernels
  - but (try to) preserve the application compatibility
- NUSE (network stack in userspace)
  - No delay of network stack update
  - Application can choose a network stack if needed

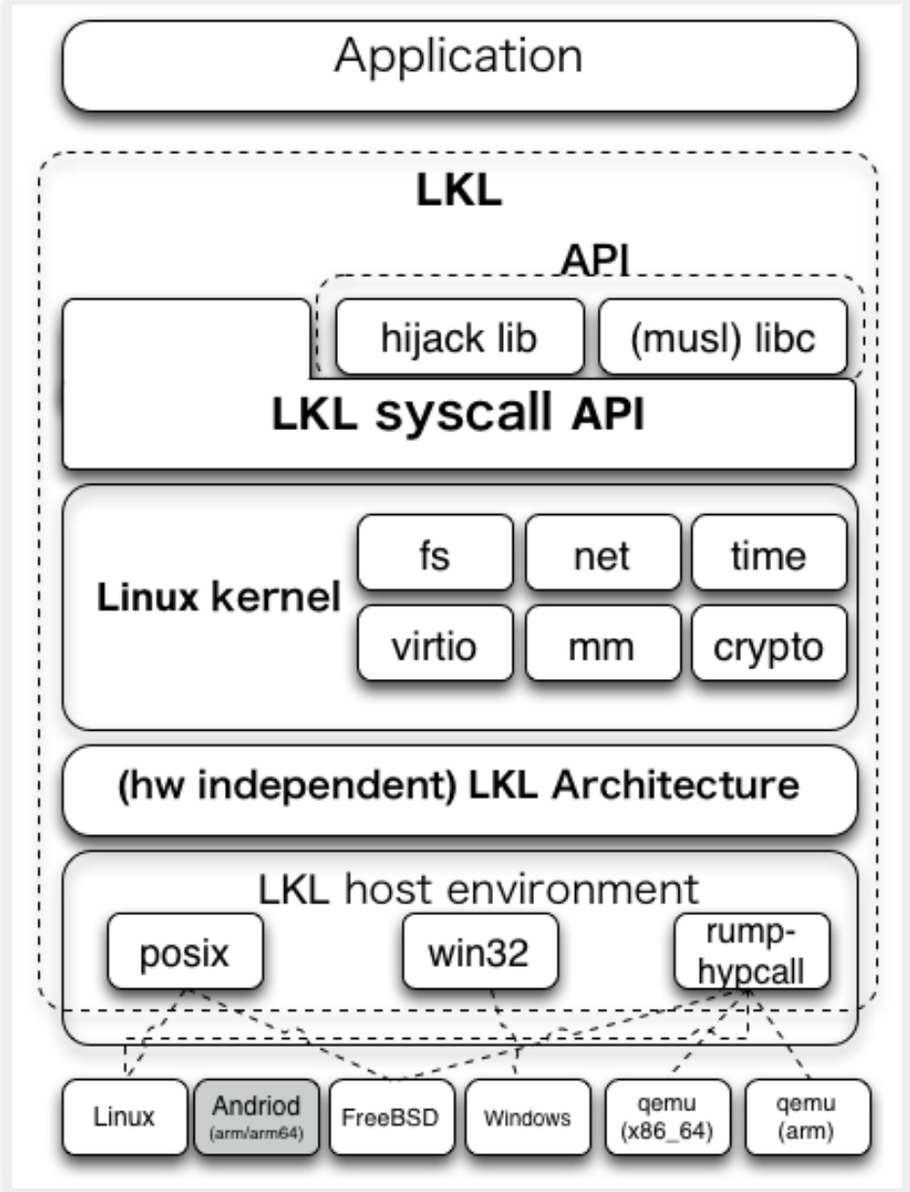
Userspace implementations

- Toys, Misguided People
- Selfish

- Motivation
  - Trying to present that a Toy is practically useful

Linux Kernel Library intro (again)

- Out-of-tree architecture (h/w-independent)
- Run Linux code on various ways
  - with a reusable library
- h/w dependent layer
  - on Linux/Windows /FreeBSD uspace, unikernel, on UEFI,
  - network simulator (ns-3)
  - **Android**



LKL: current status

- Sent RFC (Nov. 2015)
  - no update on LKML since then
- have evolved a lot
  - fast syscall path
  - offload (csum, TSO/LRO)
  - CONFIG_SMP (WIP)
  - json config
  - qemu baremetal (unikernel)
  - on UEFI

https://github.com/lkl/linux

Extensions to LKL

- Android (arm/arm64) support (lkl/linux#372)
- raw socket extension (only handle ETH_P_IP) (not upstreamed yet)
- hijack library enhance (not upstreamed yet)

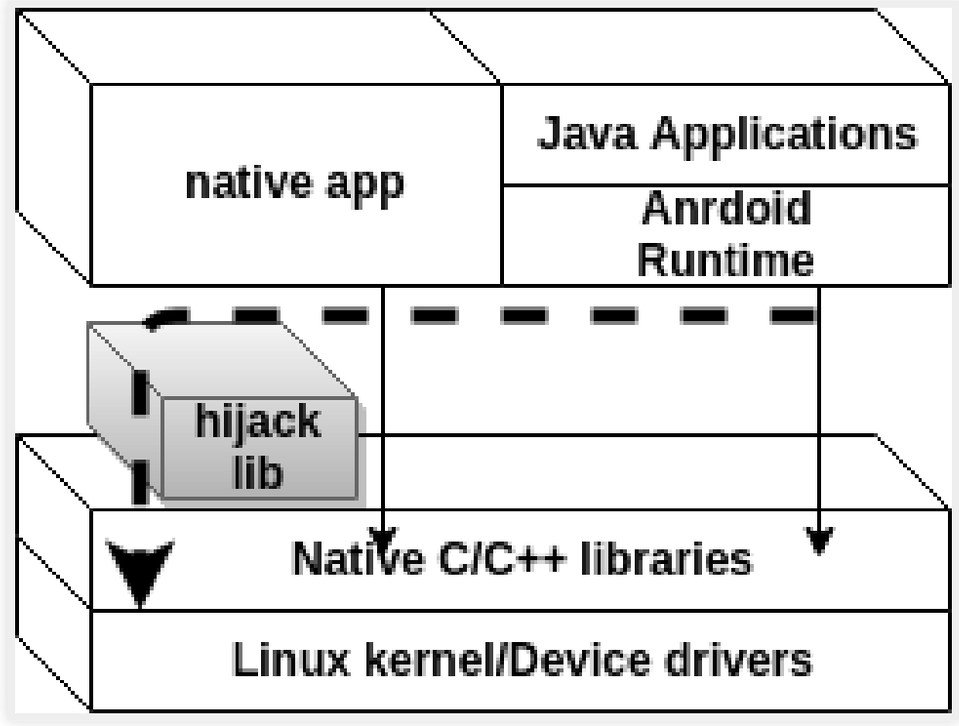
HOWTO

```
% LD_PRELOAD=liblkl-hijack.so netperf XXX # console app
% setprop wrap.app LD_PRELOAD=liblkl-hijack.so # Java app
```

```
{
"gateway": "10.206.211.1",
"interfaces": [
 {
 "ifgateway": "202.214.86.129",
 "ip": "202.214.86.168",
 "mac": "02:87:f8:27:22:02",
 "masklen": "26",
 "param": "/dev/tap23",
 "type": "macvtap"
 }
],
"debug": "0",
"singlecpu": "1",
"delay_main": "500000",
"sysctl": "net.ipv4.tcp_wmem=4096 87380 2147483647;net.mptcp.mptcp_debu
}
```

hijack library

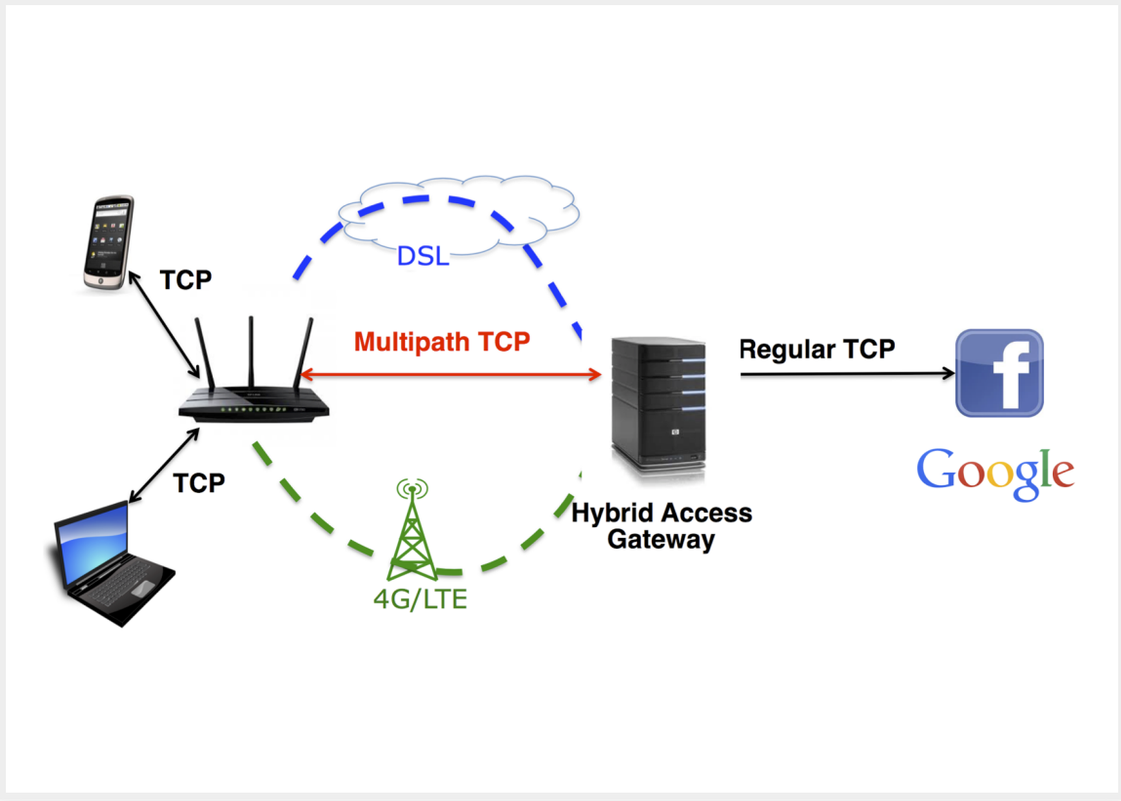
- For smooth replacement (i.e., hijack) for Android UI app syscalls (java-based)
  - bionic is more familiar than glibc
  - only socket-related calls are redirected
  - handling a mixture of host and lkl descriptors



New feature introduction

- Example
  - Multipath TCP (http://multipath-tcp.org/)
  - out-of-tree for long time

Multipath TCP



- An extension to TCP subsystem
- application compatibility (unlike SCTP)
- Use multiple paths
  - better throughput (aggregation)
  - smooth recovery from failure (handover)

http://blog.multipath-tcp.org/blog/html/2015/12/25/commercial_usage_of_multipath_tcp.html

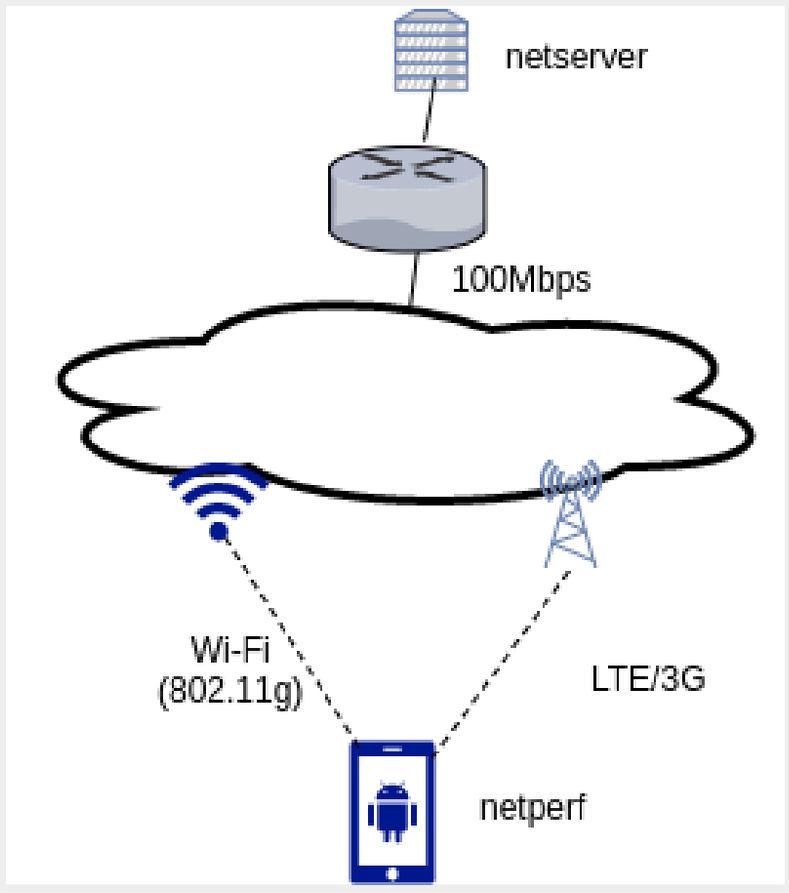
Demo

- verify site (cat /proc/net/mptcp base detection)
  http://amiusingmptcp.de/

No penalty with userspace network stack ?

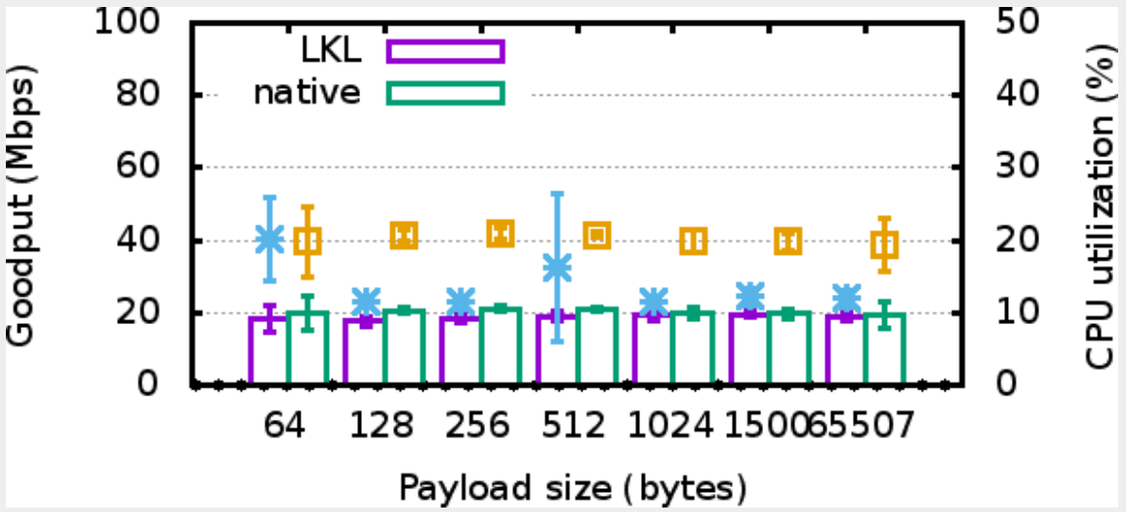
- Condition
  - To use Linux mptcp w/o replacing kernel
- Questions
  - Is NUSE working fine (Will users wanna use it) ?
  - How different from native Linux kernel ?
  - With tolerable amount of overhead ?

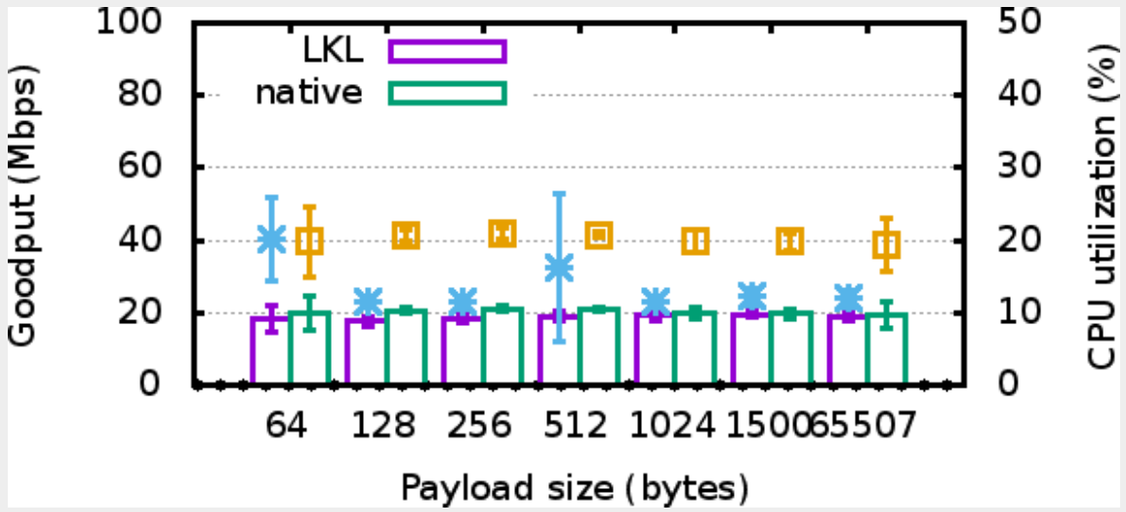
netperf measurement



- Client
  - Nexus5 anrdoid 6.01 (rooted)
  - LTE, wifi
  - LKL arm/android patched
  - or native kernel
- Server
  - Ubuntu 16.04 (amd64) on KVM
  - virtio/Etherlink (uplink: 100 Mbps)
  - mptcp-4.4.70 (v0.92)
- Software
  - netperf 2.7.x
  - 10 seconds TCP_STREAM, TCP_MAERTS
  - 5 trials, over 64-64K byte packet

Single path (Wi-Fi only)





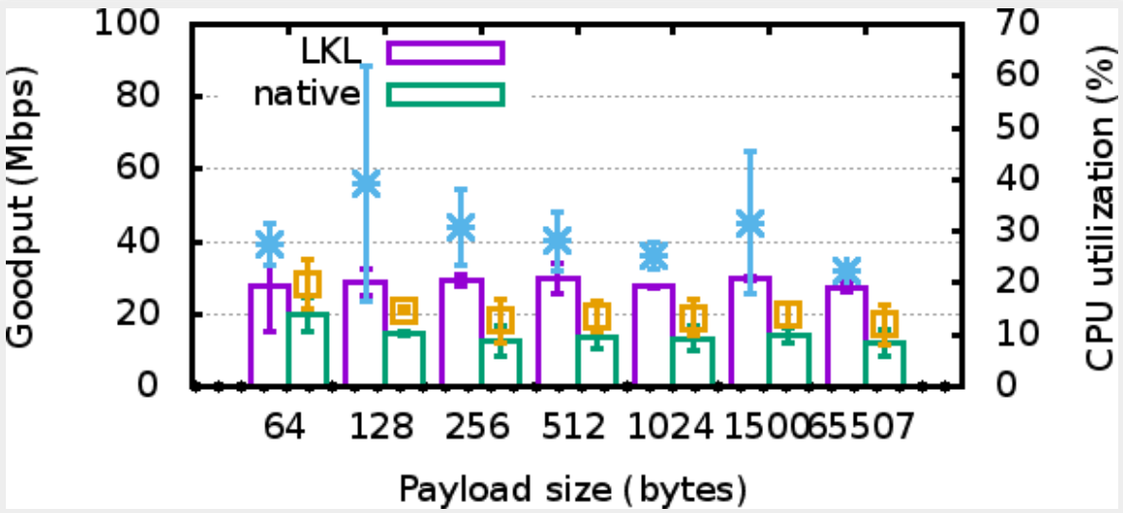
Tx (TCP_STREAM)

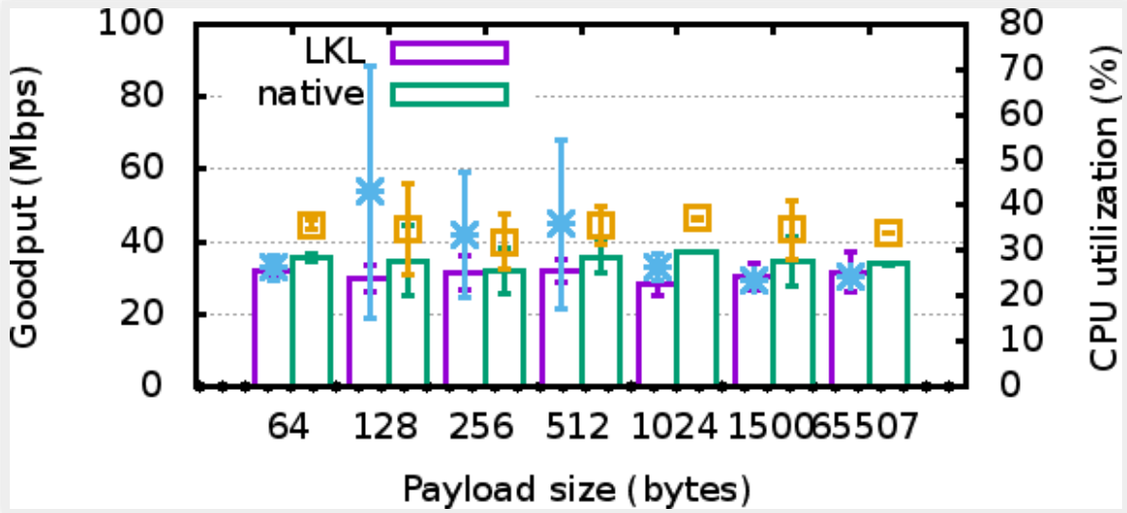
- Condition
  - phone: LKL v.s. (stock) kernel

Rx (TCP_MAERTS)

- Comparable goodput
- CPU utilization: LKL < native

Multipath TCP





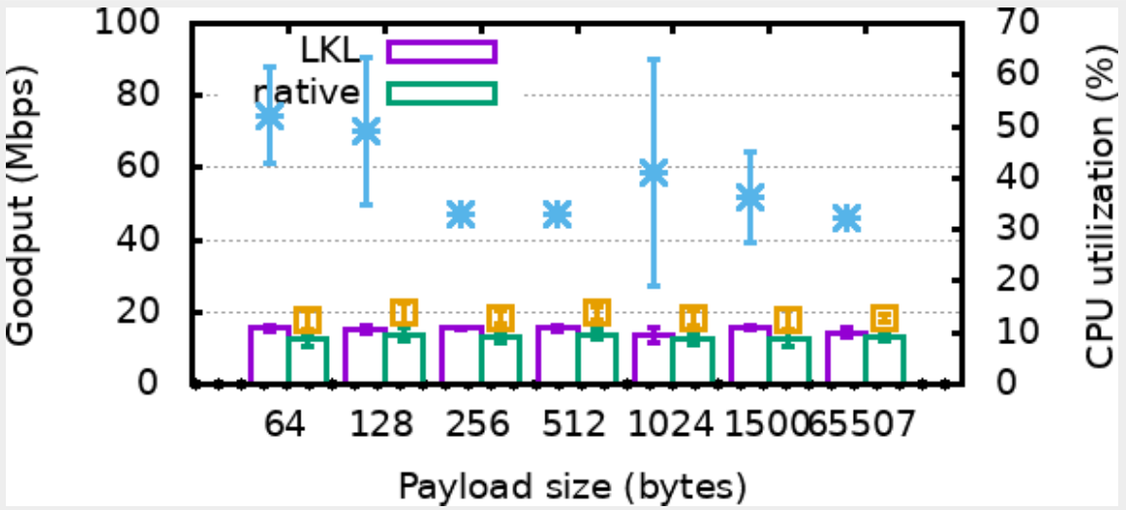
Tx (TCP_STREAM)

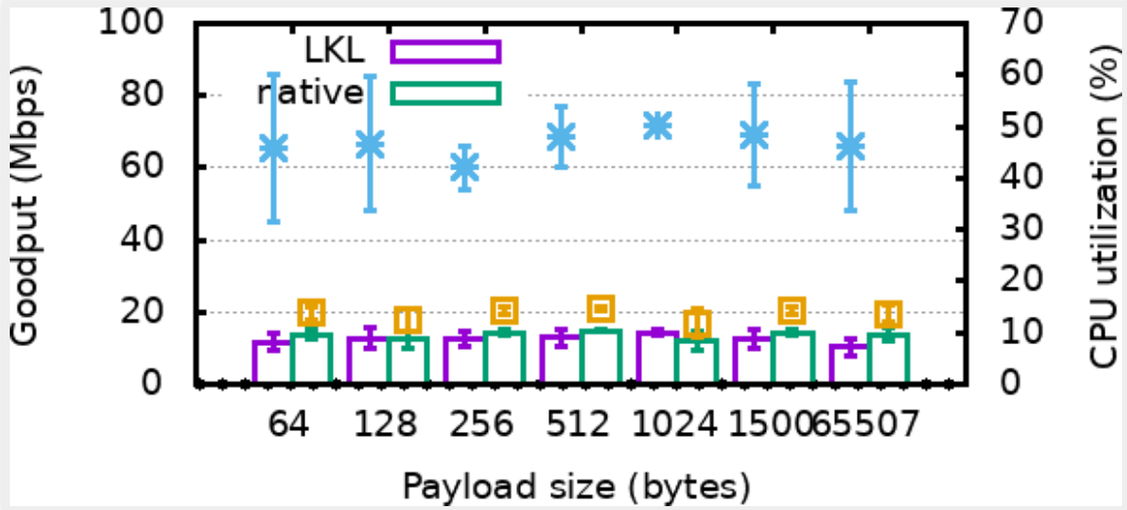
- Condition
  - phone: LKL v.s. mptcp kernel

Rx (TCP_MAERTS)

- Goodput (Tx) LKL > native
  - even it's using multipath
- CPU: unstable (LKL)
  - LKL > native

Multipath TCP (Korea/KT)





Tx (TCP_STREAM)

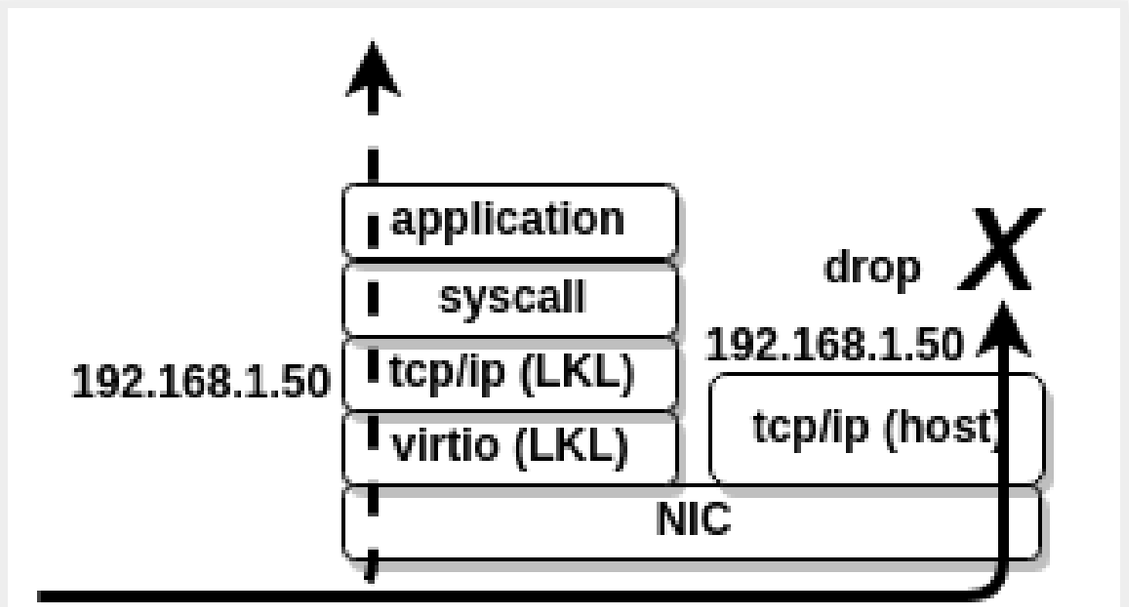
- Condition
  - phone: LKL v.s. (stock) kernel
  - native uses single-path/
    LKL uses multi-path
  - at Ibis hotel

Rx (TCP_MAERTS)

- Goodput: No much gain with LKL
  - even it's using multipath
- CPU: unstable (LKL)
  - LKL > native

Observations

- IP conflicts may heavier
  - processed **twice** (host/lkl) per packet
- Results are often unstable
  - difficult measurement under wireless media



Limitations

- Implementations
  - DHCP only boot time (handover will fail)
  - IPv4 only on cellular interface (rmnet0)
- Fundamental limitations of hijack library
  - asynchronous signal unsafe
  - MT unsafe
- Required tweaks
  - grant NET_RAW permission (packet socket)
  - need filter out RST packet from host

```
iptables -A OUTPUT -p tcp --tcp-flags RST RST  -j DROP
```

Further investigations

- other platform
  - iOS11 now shipped userspace implementation
- profiling

Summary

- Use out-of-tree kernel as a library on Android
  - *make your code easier to distribute*
  - with privileged installation/operation
- Comparable goodput over WiFi/LTE
- Unstable CPU utilization with LKL
- You can prepare your library file for your own purpose

Backups

Alternate network stacks

- lwip (2002~)
- mTCP [NSDI '14]
- SandStorm [SIGCOMM '14]
- rumpkernel [ATC '09]
- SolarFlare (2007~?)
- libuinet (2013~)
- SeaStar (2014~)

None of them are feature-rich, or one-shot porting