Internet-scale Virtual Networking
Using Identifier-Locator Addressing

Petr Lapukhov
Network Engineer
Facebook
Virtual networking is confusing!
What problem FB is trying to solve?
Linux application containers

Simpler and more lightweight than
Container networking: challenges

• Many containers per host: address sharing
• Containers can move: address would change
Container networking: two goals...

• IPv6 address per *process*
• Address *mobility* <>
Identifier Locator Addressing (ILA)
Identifier / Locator split
Predecessors: ILNP/GSE/8+8...
IPv6 Address

- 128 bit
- Used for routing
- Immutable name
Mobility with Locator/ID split

• Every host gets /64 prefix - locator (!)
• Processes migrate between machines
• Identifier remains the same, locator changes
• Mutable locator require transport stack modification <>
ILA specifics

- **Hides** locator changes from transport layer
- **Transport** always sees one **fixed** locator (/64 prefix)
- Stateless rewrites (NAT) **below** transport layer <>
ILA Host

• Every host needs a routable locator: IPv6 /64 prefix
• Hosts need to maintain ILA mapping cache
• Non-ILA hosts talk to ILA hosts via ILA routers <>
Host 1

Process 1

ILA Address
face:b00c::1234

Locator
tec0:cafe::/64

Host 2

Process 2

ILA Address
face:b00c::5678

Locator
tec0:beef::/64

SIR Prefix
face:b00c::/64

Before NAT
On Wire
(after 1st NAT)

On Wire
(after 2nd NAT)
SIR Prefix

• SIR = “Standard Identifier Representation”
• SIR Prefix = 64 bit “fixed-locator” seen by transport
• Injected into network by all ILA Routers (anycast) <>
ILA Addresses: one “virtual” /64 subnet

Process 1
face:b00c::1234

Process 2
face:b00c::abc

Process 3
face:b00c::5678

ILA network
face:b00c::/64

ILA Router

Non-ILA networks
ILA Router

• Knows of all active mappings
• Injects /64 SIR prefix into IPv6 network
• “Mediates” between ILA and non-ILA hosts
• May also mediate between ILA-hosts
• Acts like an IPv6 router on “virtual” /64 segment <>
ILA Hosts

Injects SIR prefix

face:b00c::

ILA Host responds directly to non-ILA

Translates

ILA Router

Injects SIR prefix

IPv6

Non-ILA Host

Talks to
Using ILA Router to b/w ILA hosts

ILA Hosts

ID: 1234

Route using

Tells to face:b00c::1234
send to ILA router

ID: 5678

ILA Router

Translates and routes to

Sends redirect

Injects SIR prefix

No locator for face:b00c::1234 send to ILA router
What about control plane?
Goal: disseminate ILA mappings
Good news: there is no standard!
ILA specifics

- ILA *routers* know of all mappings
- ILA *hosts* always *publish* into mapping system <>
ILA: Data-plane assistance

• ILA routers may send **redirect** messages
• Hosts may send **stale mapping** messages
• Similar to ICMPv6 messages
Now the fun: identifier mobility
Container moves b/w hosts

ILA Hosts

A

B

C

ILA Routers

Scheduler requests 1234

Flow

ID: 1234

ID: 5678

New Locator Forward for 1234

Invalid

Redirect

Fallback to

Flow
Mobility recap

• Data-plane driven cache invalidation
• ILA routers provide fallback on cache invalidation <>
Network Setup

• Every server gets /64 route
• Summarized to /54 on rack switch
• Summarized to /46 on pod switch
• Sums up to /32
• Can fit 32 data-centers per /32

DC Hierarchy
Spine
Pod = /46
Rack = /54
Host Configuration

• New /64 per host - every machine @FB
• Part of host bootstrap info
• Applied by Chef recipe

$ ip -6 a 1s
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536
   inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qlen 1000
   inet6 2803:6082:18e0:e825::1/64 scope global deprecated
       valid_lft forever preferred_lft forever
   inet6 2401:db00:11:d03a:face:0:25:0/64 scope global
       valid_lft forever preferred_lft forever
   inet6 fe80::f652:14ff:febe:fe54/64 scope link
       valid_lft forever preferred_lft forever
Unique IPv6 per process!

• Random 64bit ID allocated on container start
• UUID64 - timestamp + host name + some magic <>
How can process use IPv6?

• Passed explicitly as environment variable
• …Could be enforced via LD_PRELOAD
• Namespaces/ipvlan currently experimental <>
DNS Support

- DNS name per container
- E.g. ‘tsp-prn.netsystems.test-task.0.tw.local’
- Both AAAA and PTR created simultaneously
- ZippyDB as backing store <>
Host support: Kernel 4.x+

• ILA rewrites: Light-weight tunnels (LWT)
• Linux route lookup + rewrite action
• Programmable via netlink API <>
Host support: *ip route* primer

```bash
modprobe ila

# Set local SIR address
ip -6 addr add face:b00c:0:0:2555:0:1:0/128 dev lo

# Add peer with ILA translation
ip -6 route add face:b00c:0:0:2555:0:2:0/128 encap ila 2803:6080:8960:4473 via 2401:db00:20:4001::a

# Add local prefix translation
ip -6 route add table local local 2803:6082:1950:401:2555:0:1:0/128 encap ila face:b00c:0:0 dev lo
```
ILA Routers @ FB

• Linux machine with IPv6 forwarding enabled
• Regular routing with LWT “ila” rules
• Currently: all hosts are ILA routers <>
Control plane hack

ZippyDB

ILA Hosts

Container

Async replication

Download mappings every 5s

Publish Mapping +

ILA caches synchronized

Container starts & address get
Control plane recap

- ZippyDB to **push** & **pull** mappings
- Runs on ~ 10k+ hosts
- Low number of mobile tasks (100s)
- Very easy to experiment with <>
Operational implications

• ICMP: TTL expired, unreachable (traceroute, PMTUD)
• Contain “translated” SRC/DST addresses
• Need fix in kernel to translate back <>
What’s next?
eBPF
eBPF

• BPF (Berkeley Packet Filter) - stuff you use in tcpdump
• eBPF - extended BPF
• JIT-compiled BPF with richer instruction set
• Virtual machine in Linux kernel! <>
Why it’s a big deal?

• eBPF allows extending kernel functions
• …From user-space. On the fly.
• Multiple points of code injection in kernel
• We built the ILA router code in eBPF <>
**eXpress Data Path**

- **XDP** == Linux kernel bypass inside kernel!
- *Fast* in-kernel networking
- Packet processing pre-network-stack via eBPF
- E.g. lookup and address rewrite
- Punt to network stack if needed <>
The finale
ILA is...
IPv6 Address per process
Location independence
Builds on XDP + eBPF
Thank you