Identifier Locator Addressing

Tom Herbert

netconf – 2/13/2015

Data center virtualization

Capability that every task in the data center can be seamlessly live migrated per discretion of a job scheduler

- Requirements (networking)
  - **The capability of task migration does not adversely affect networking control, security, performance, or behavior**
  - Transparent to applications and users
  - **Zero** performance difference between before and after introducing capability
  - Tasks run in containers, probably **not** VMs. Most tasks will probably never be migrated
- Solution
  - Give every task its own address
  - Address stays with task over migrations, open connections also
  - IPv6 with Identifier locator addressing (ILA) in data center

Identifier Locator Addressing

- Split IPv6 address into locator and identifier like in ILNP
  - Only data plane concepts from ILNP, not control plane
  - Address formats for virtualization with VNID also defined
- Identifier indicates unique identity of task (who)
- Locator indicates physical host where task runs (where)
- Identifier+locator is fully qualified IPv6 for sending on wire
- If task migrates between hosts, its locator changes but its identifier does not

Address split

- Locator
  - 64 bits identifier of physical hosts
  - Routable
  - Not used as connection endpoint
- Identifier
  - 64 bit logical endpoint address of virtual node
  - Not routable
  - Used as connection endpoint

Benefits

- Virtualization without encapsulation
- 64 bit address space can address a lot of nodes, allow autonomous task identifier assignment
- ILA addressed packets look just like regular application packets on the wire, all the normal offloads should just work (RSS, ECMP, TSO, etc.)
- Potentially reduce storage requirements for addresses in some instance (possibly routers)
- This is **not** LISP!

Isolation non-requirement

- Unlike “normal” network virtualization (VM+encapsulation protocol) no virtual network isolation needed (or offered)
- ILA applies where tasks and users are trusted
  - Internal users of current DCs
  - Not obviously hostile
  - Implement compliant congestion controls
- If users are not trusted, alternate mechanisms that allow strong controls or security like GUE should be used
  -

Identifier To locator mapping

- Need to map identifiers to locators
  - Resolve locator for identifier before transmit time (essentially glorified ARP)
  - Equivalent to mapping virtual address to physical address in canonical network virtualization
  - Mappings are dynamic as locator for identifier can change
- Hosts maintain a cache of mappings
  - Mapping propagation done by a network “control plane”
  - Push and pull models
  - Work with control planes for NV as new address type

User visible addresses

- Identifiers are encoded in “Standard identifier Representation” (SIR) IPv6 address
  - Upper 64 bits is SIR prefix, not routable. Lower 64 bits is identifier
  - To send, SIR overwritten with locator
  - At peer, locator overwritten with SIR prefix before giving to application
- Use of SIR addresses is transparent to application/user. Can be returned in DNS, used to connect TCP, etc.
- Local addresses also provided in SIR notation
- Note implied use of stateless NAT. SIR->ILA, ILA->SIR

Implementation questions 1

- Given performance and transparency requirements, it *seems* like this should be an integrated network virtualization solution (no vSwitch)
- Routing, netfilter, IP tables, etc. need to operate on identifiers. External interface could use SIR, but is there rationale/benefits to making these identifier aware?
- Need restrict local address binding of tasks. Bind to INADDR6_ANY means bind to identifier(s) for task.
  - Network namespaces do this, but do we need all the complexity and capabilities that come with that?
  - How far do we need to go, should application only see identifier addresses for its task? Should see they all addresses (nothing like this in current use of task/addresses by the way)

Implementation questions 2

- Should locator mappings be part of routing table (similar to ARP and cached in connection socket)
  - As a route that can be cached with connection sockets
  - Fast table lookup on every send?
  - Would be nice if we could set locatosr in UDP/TCP before checksum (avoid NAT operation)
- New 64 bit identifier address type? Potential saving of 128 bytes per PCB?
- Should task addresses should be configured as real interface addresses (ping on full 128 bit address should work)?
- Is TCP_REPAIR sufficient for connection migration?