Linux Network Virtualization

Bogdan Purcăreață
Software Engineer, Virtualization Team

June 2013

Table of Contents

Motivation

- Networking resources are limited
- Virtual machines need access to the exterior
  - KVM
  - LXC
  - whatever the technology
- Flexibility
- Manageability

The Network Namespace

- Virtualize network resources
  - Devices
  - IP addresses
  - Routes
  - Sockets
- Different networking stacks
- Easy to create and configure
- Low overhead

Network Namespace Usage

- Virtualization – own view of system resources
  - Multiple eth0 and lo devices
  - Several Apache servers listening on *:80 on the same host
- Isolation – no access to outside resources
  - No traffic sniffing
  - No outside interface shutdown

Interesting Features

- Security
  - Compromising a server in a network ns isolates the damage
- Resource Management
  - Network resources can easily be assigned to a set of processes
- Traffic control
  - Improved flexibility
- Consolidation
  - Aggregate several servers, no impact on their configuration
- Mobility
  - Easy to checkpoint resources
  - Move IP across network, avoid conflicts at restart

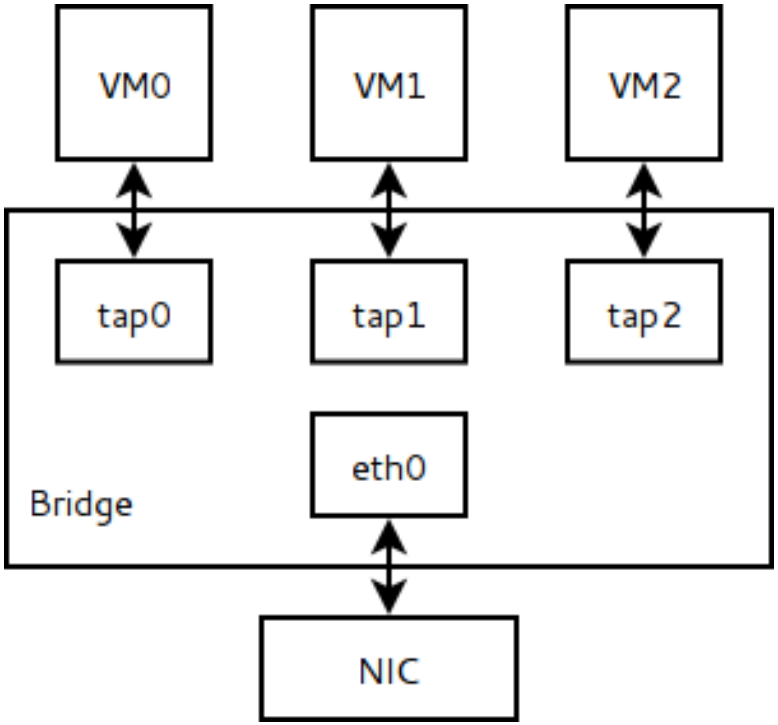
Usage

- CLONE_NEWNET flag
  - clone()
  - unshare()
- CAP_NET_ADMIN required
- 1 loopback interface per network namespace
- Network device “moving”
  - *“only the network namespace owner can move a network device”*
- etun device
  - Communication between namespaces
  - “Virtual ETHernet device (tunnel)”

Virtual Ethernet Bridging

- IEEE 802.1d
- Physical NICs (Network Interface Controller)
  - They lose identity once part of bridge
  - Only bridge TCP/IP info becomes relevant
- Virtual interfaces (TAP)
- LAN extension

Virtual Ethernet Bridging - Diagram



picture from http://hzqtc.github.io/image/bridge.png

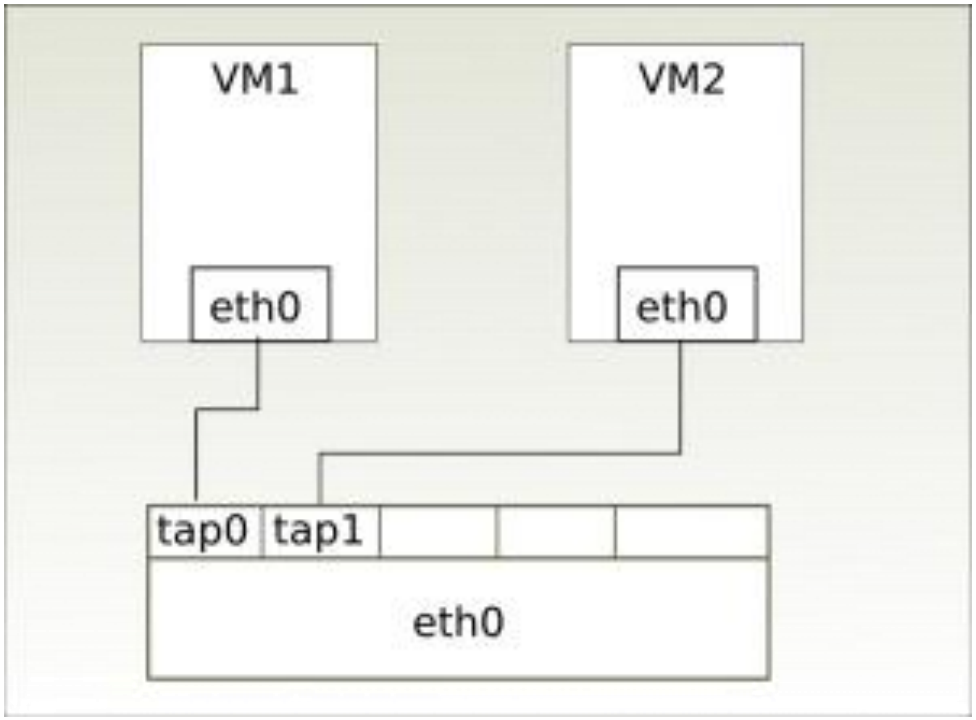
TUN / TAP Devices

- Virtual network kernel devices
- Entirely software
- No hardware network adapters
- TAP – L2, Ethernet frames
- TUN – L3, routing

MACVLAN

- MACVLAN driver
  - Virtual network interfaces
  - “Cling on” physical network interface
  - Each virtual interface has its own MAC
  - Physical interface = lower interface
  - Mac-address based virtual LAN tagging
- Tap interface
- MacVTap = MACVLAN + Tap
- Isolation between virtual interfaces and the lower device
- Lower overhead than VETH

MacVTap Diagram

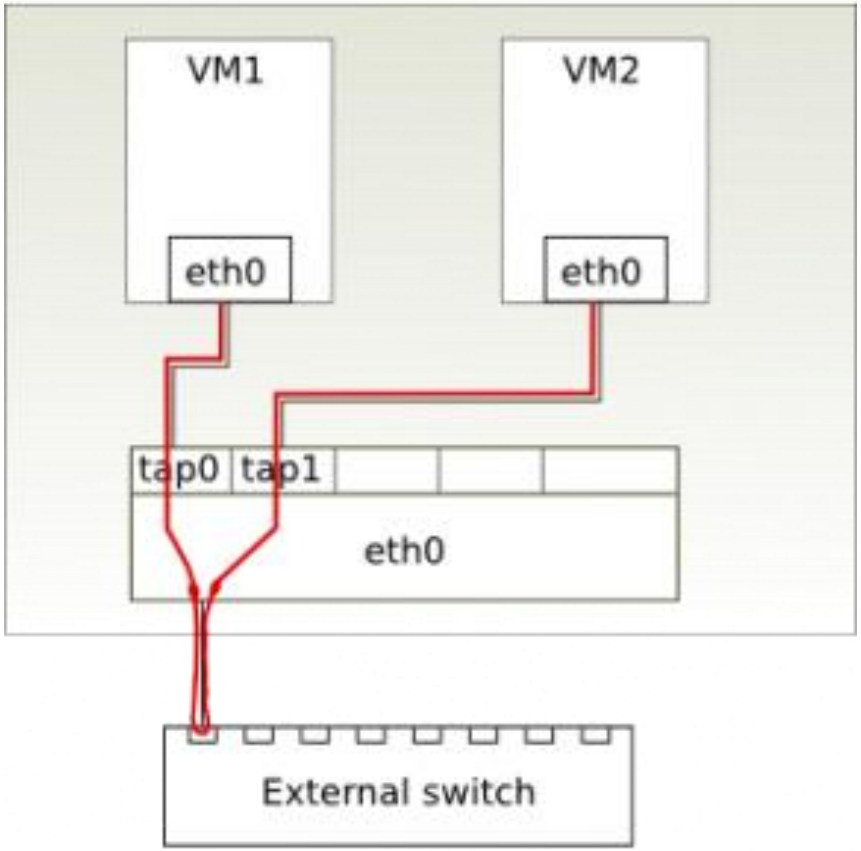


picture from http://seravo.fi/wp-content/uploads/seravo/2012/10/tap-300x221.png

MACVLAN Modes

- VEPA (Virtual Ethernet Port Aggregator)
  - Data between endpoints on the same lower device are sent via the lower device
  - Offload to external switch
  - Switch must support “Reflective Relay”
- Bridge
  - Endpoints on the same lower device can communicate directly
- Private
  - Isolation between endpoints on the same lower device
  - Connectivity only with external network

MacVTap VEPA



picture from http://seravo.fi/wp-content/uploads/seravo/2012/10/hairpin-290x300.png

References

- http://seravo.fi/2012/virtualized-bridged-networking-with-macvtap
- http://openvpn.net/index.php/open-source/documentation/miscellaneous/76-ethernet-bridging.html
- https://www.kernel.org/doc/Documentation/networking/tuntap.txt
- http://lwn.net/Articles/219794/
- http://www.ibm.com/developerworks/library/l-virtual-networking/
- http://www.pocketnix.org/posts/Linux%20Networking:%20MAC%20VLANs%20and%20Virtual%20Ethernets


freescale™