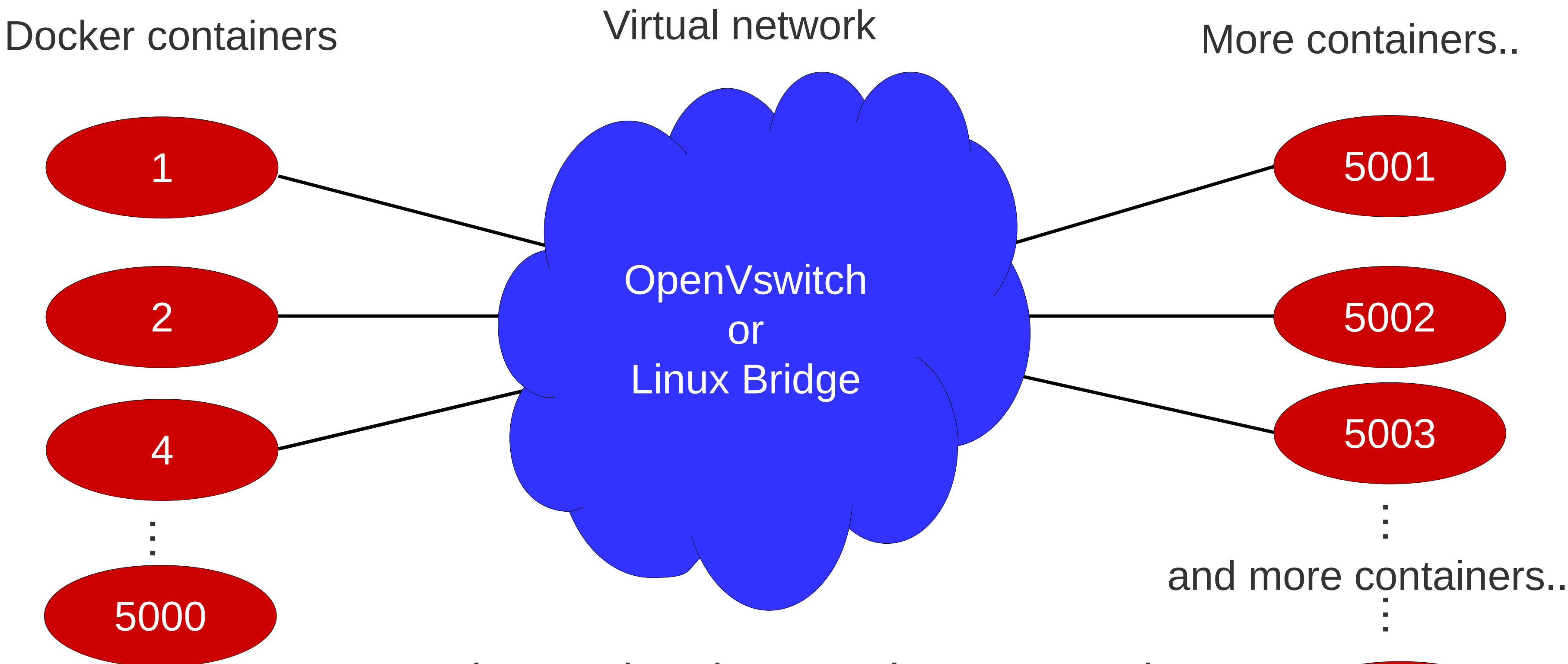
IBM Linux Technology Center

Exploration of Large Scale Virtual Networks

Open Network Summit 2016

David Wilder
wilder@us.ibm.com

A Network of Containers



Each container has a unique “network name-space” consisting of:

- Virtual network interfaces
- Routing and arp tables
- Network statistics

Project Objectives

- Build and evaluate large virtual networks on Linux using:
  - Docker,
  - Linux Bridge,
  - OpenVswitch.

- Learn how virtual networks:
  - scale.
  - perform.
  - break.

What I Measured:

- Aggregate bandwidth
- TCP connection rate
- Latency – broadcast and unicast

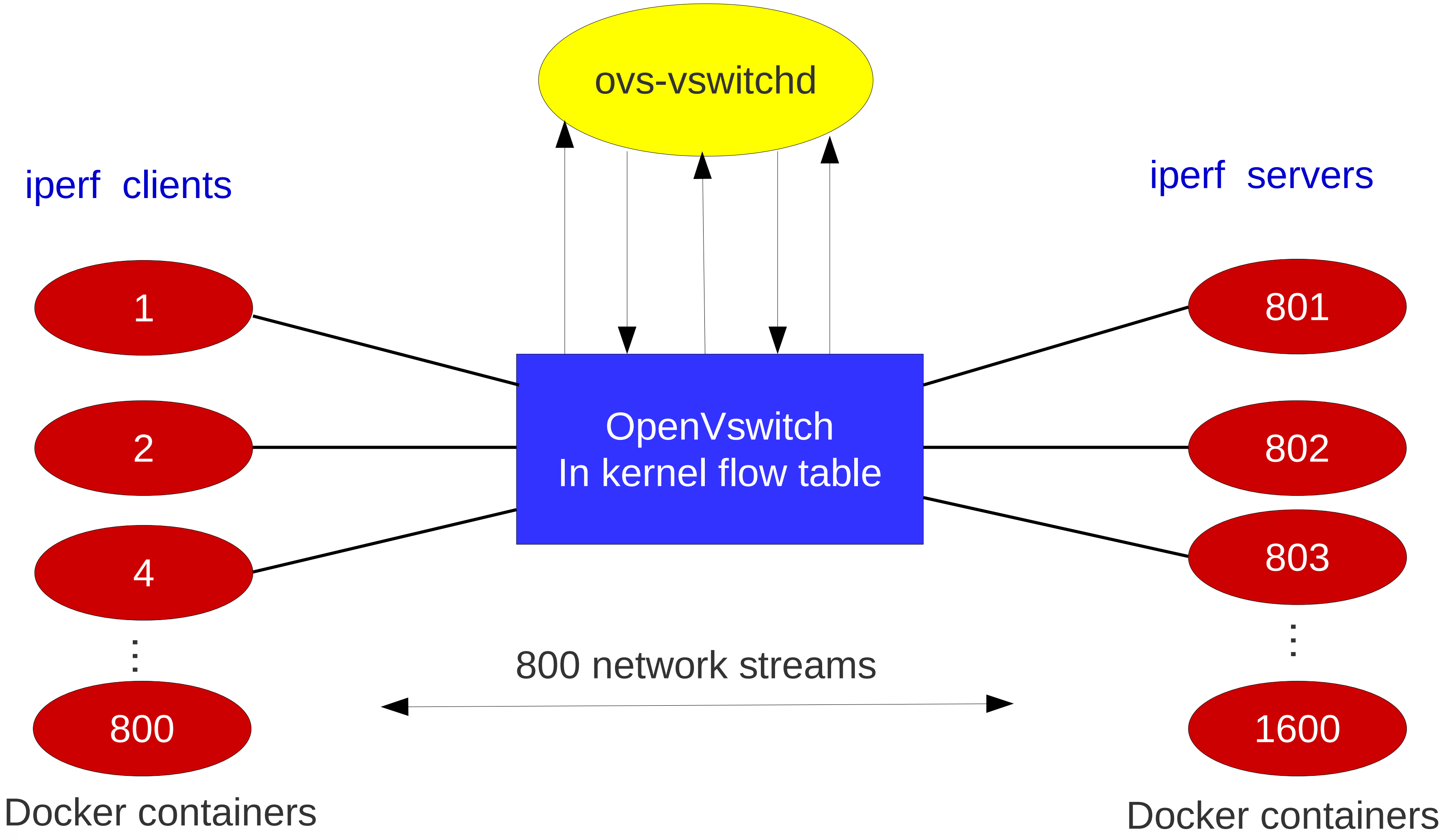
- Up to 1600 docker nodes
- IPV4

Some details..

- 80 processor power8 (SMT=8)
- 132 GB

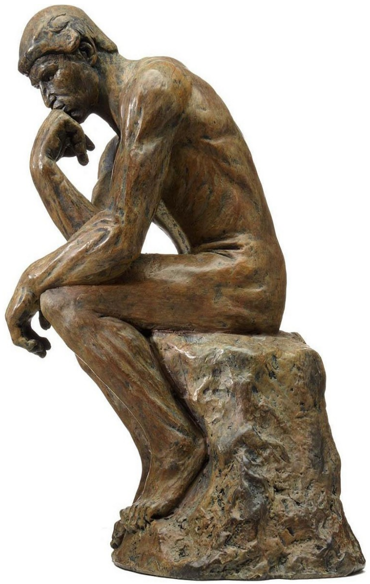
- Linux kernel version: 4.4.0 rc5+ (ppc64LE)
- OpenVswitch V2.5.90
- Docker 1.5.0

Throughput Measurements

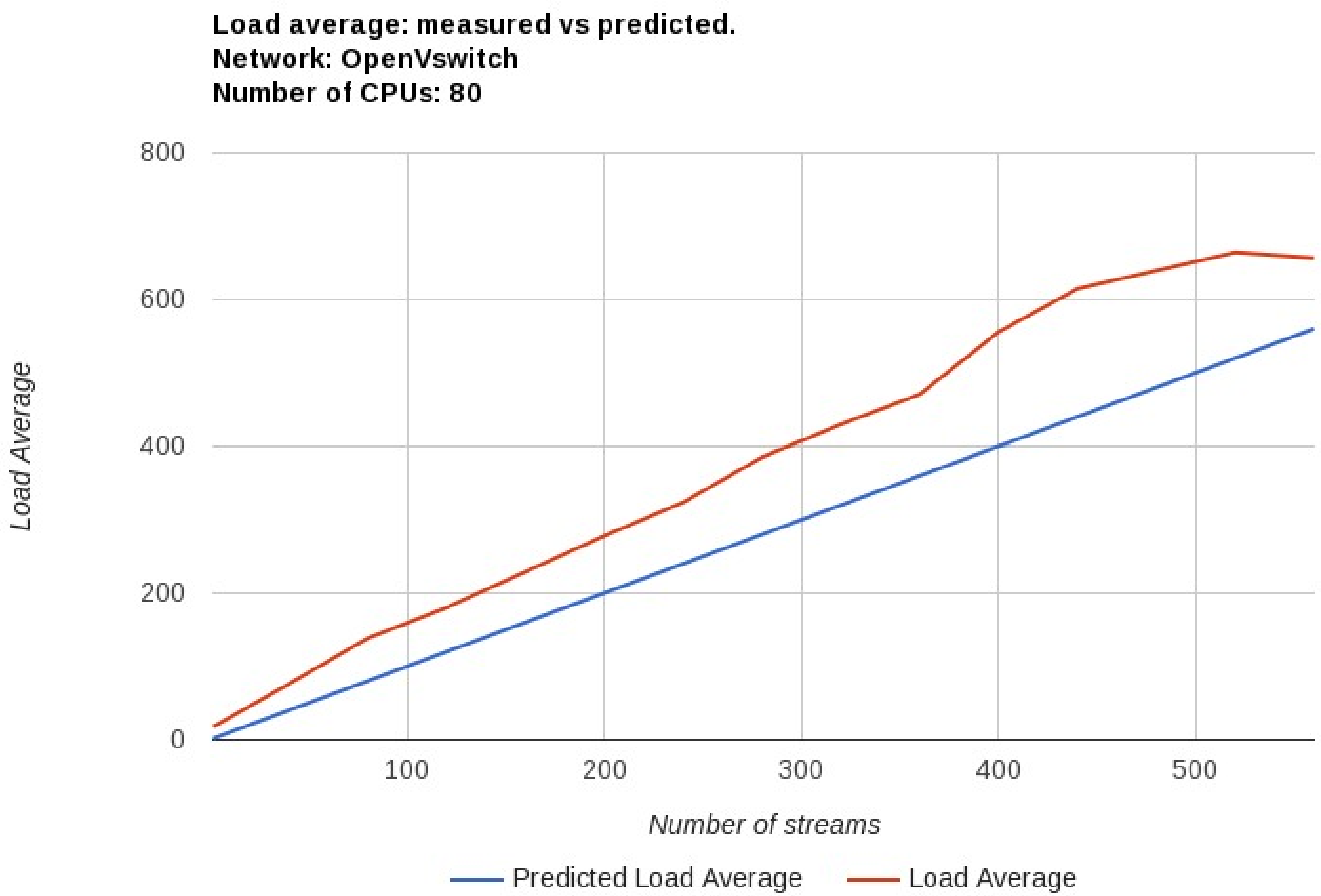


Predictions

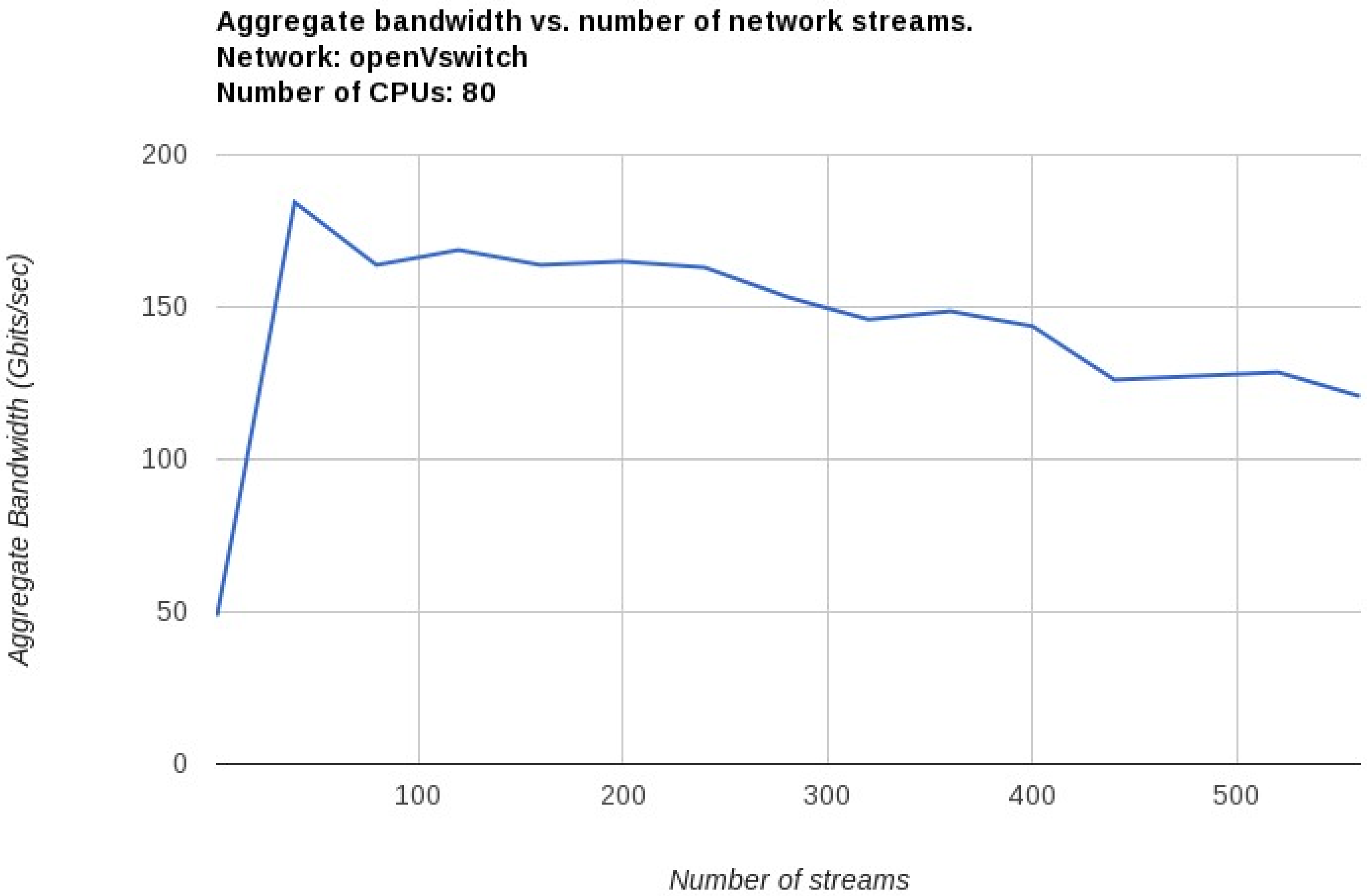
- Iperf is not performing any real I/O and will become cpu bound.
  - CPU load average ~= Number of iperf streams.
  - Throughput will level out when the number of stream equals the number of processors.



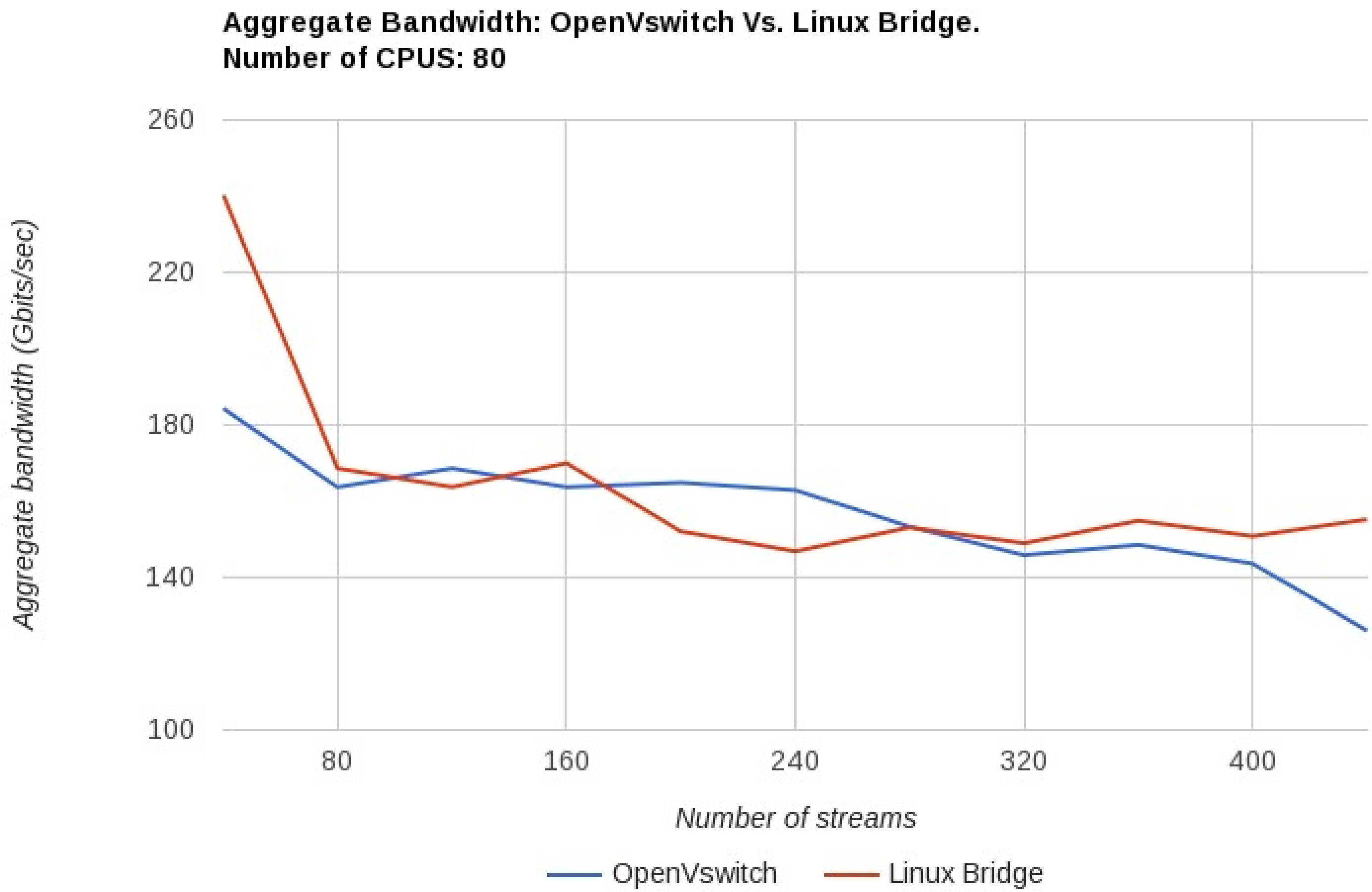
CPU Utilization:



Aggregate Bandwidth



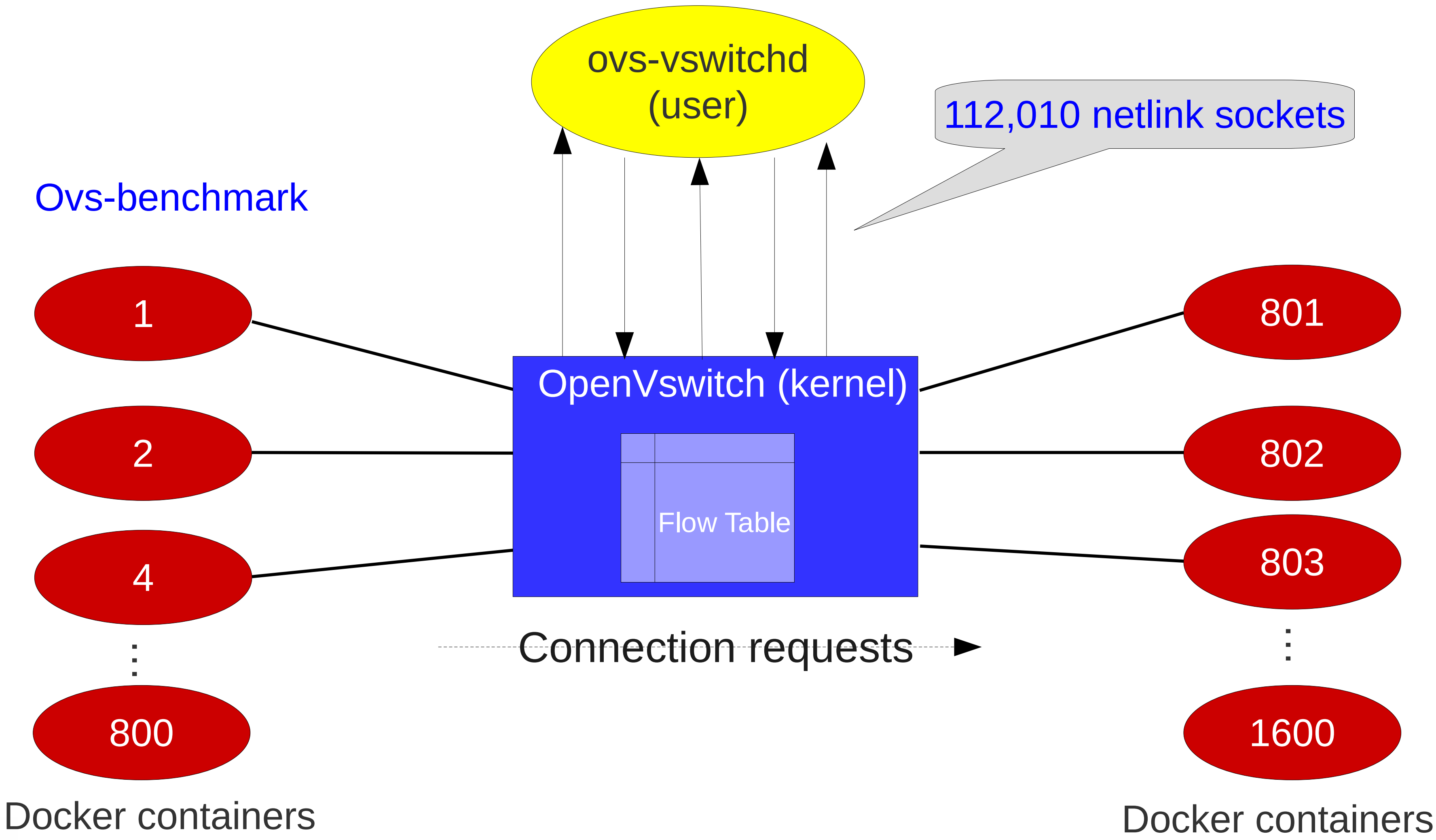
Linux Bridge compared to OpenVswitch



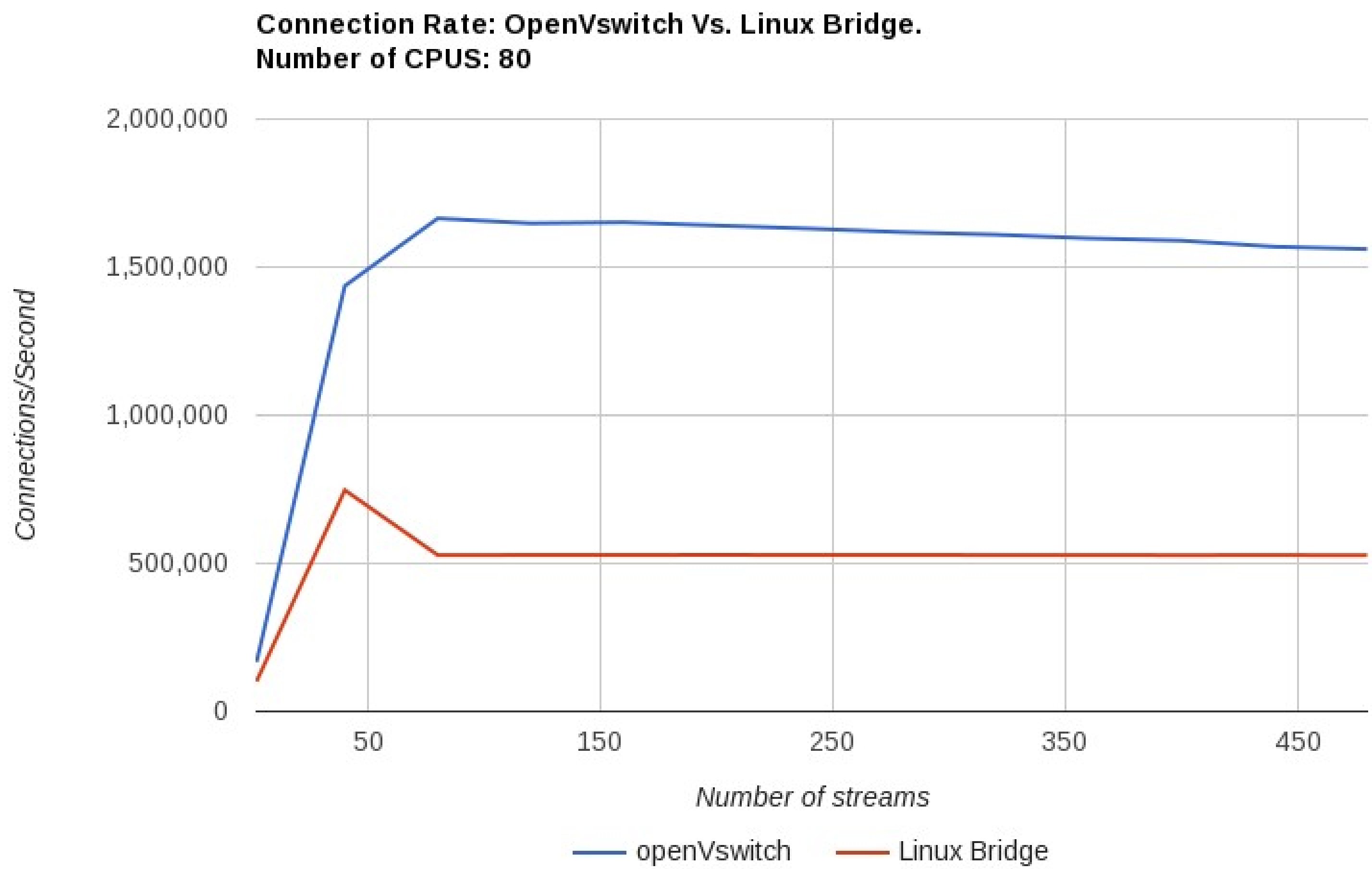
Issues

- Linux Bridge
  - Limited to a maximum of 1024 ports.
  - Linux/net/bridge/br_private.h
    - #define BR_PORT_BITS 10
    - #define BR_PORT_BITS 14 /* (16k ports) */
- OpenVswitch
  - At over 400 streams iperf sporadically failed to establish a connection returning the error "No route to host".

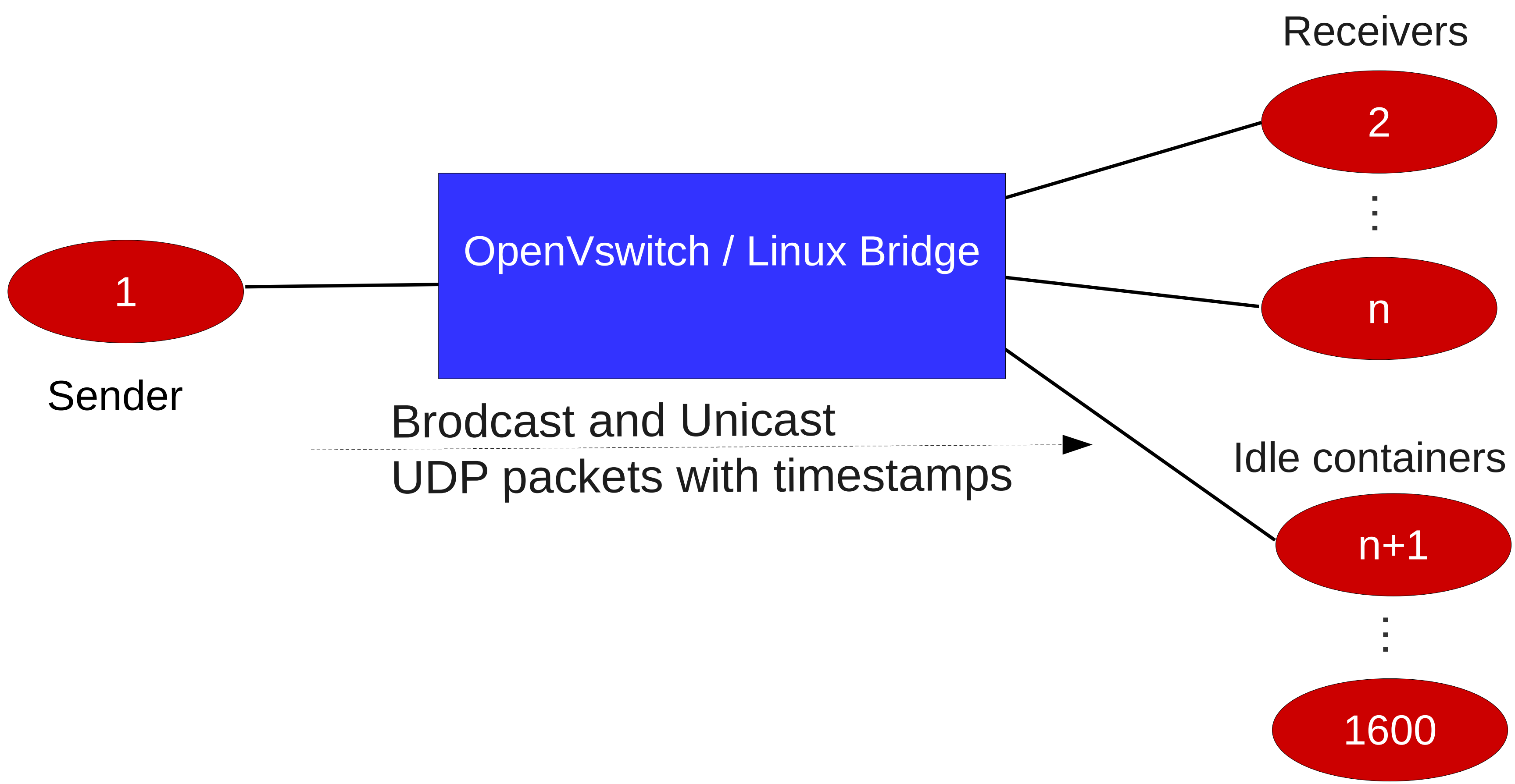
Measuring TCP Connection Rate



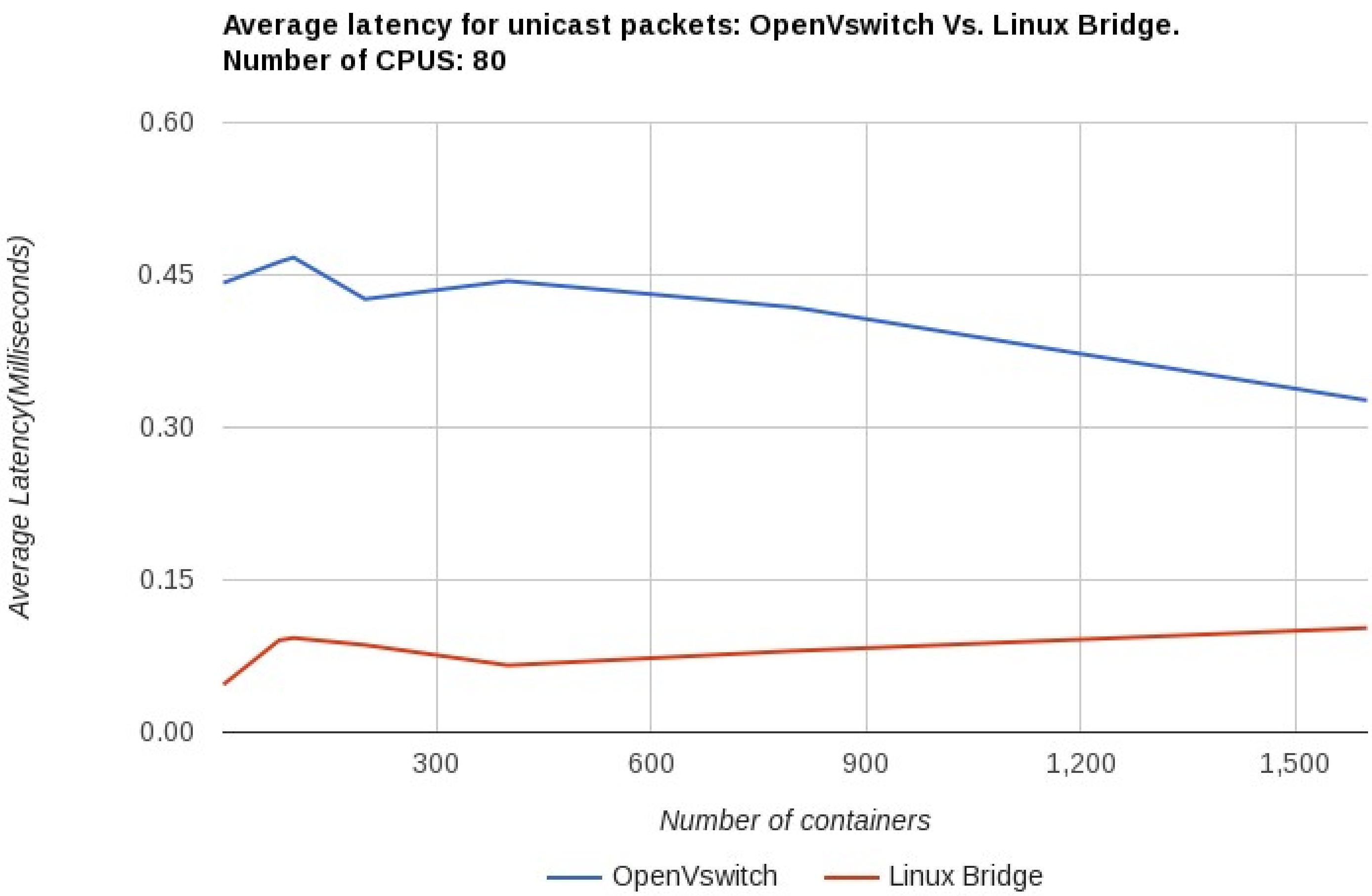
TCP Connection Rate



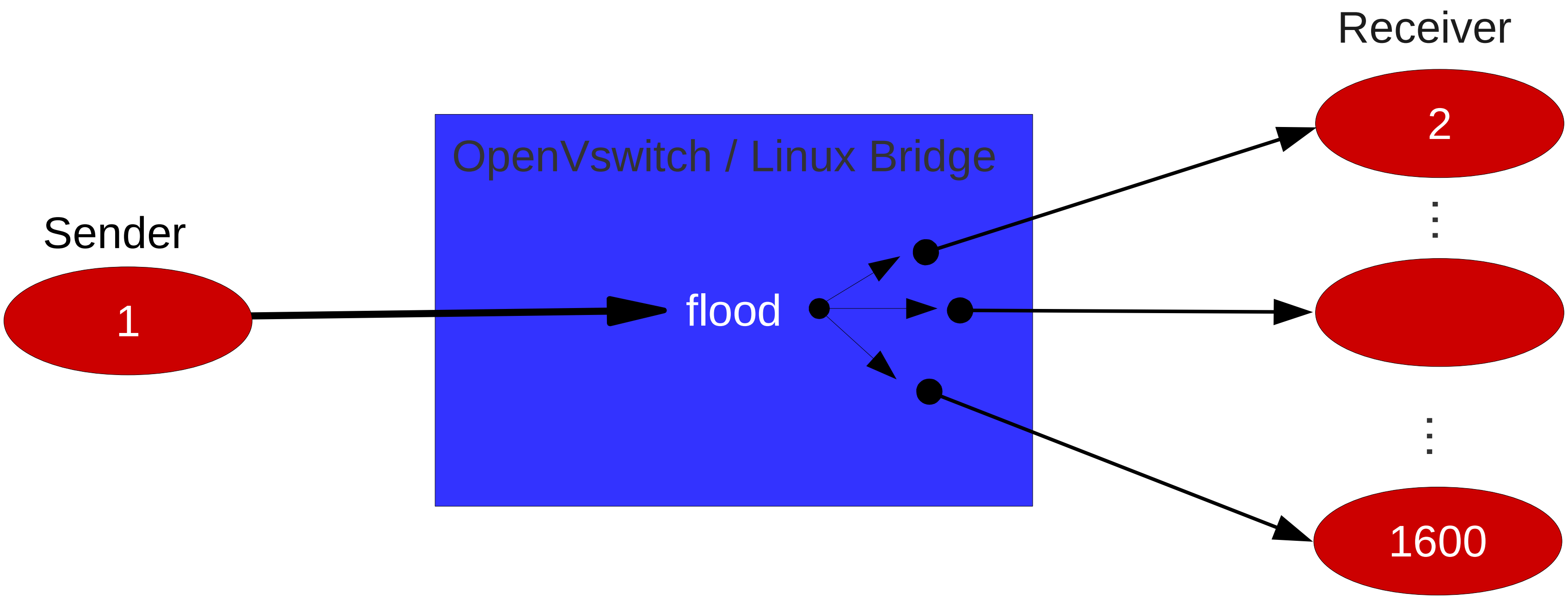
Measuring latency of unicast and brodcast packets.



Unicast Latency

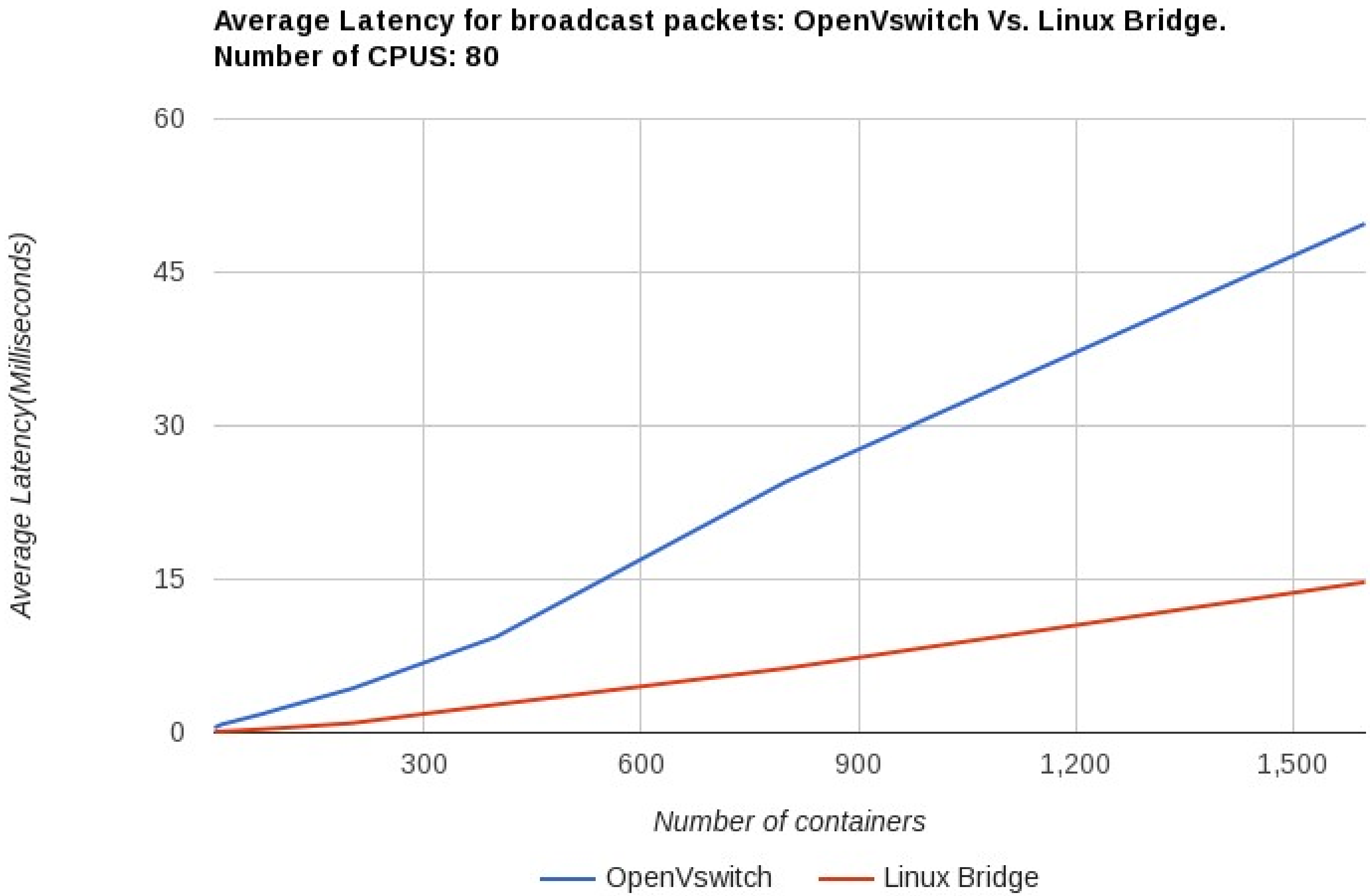


Packet Flooding.

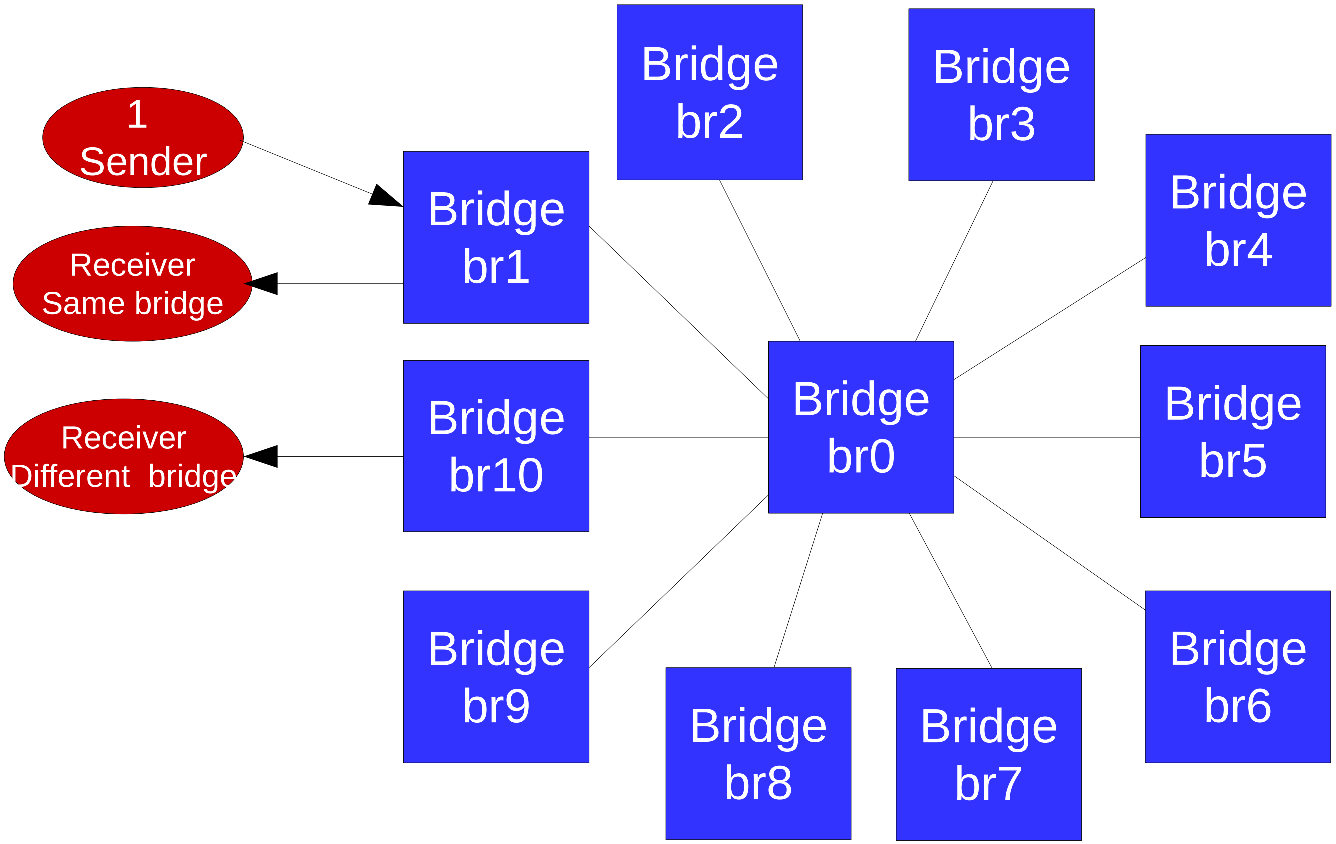


sysctl -w net.core.netdev_max_backlog=10000

Broadcast Packet Latency

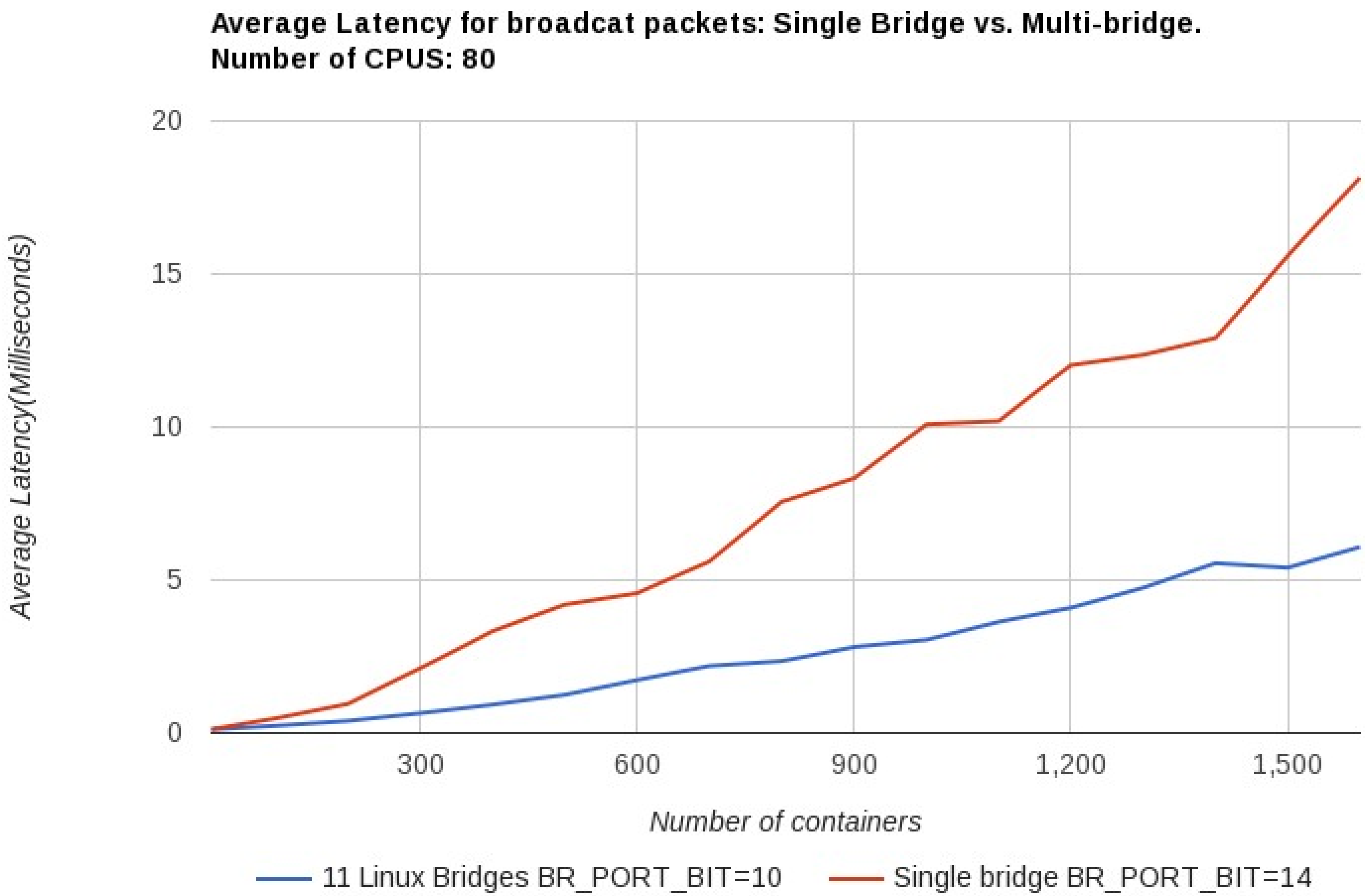


Multiple Bridge Configuration - Eleven Bridges



Up to 1600 Containers were evenly distributed between br1 through br10

Broadcast Latency – single vs. multiple bridges



Tuning

- Prevent broadcast packet loss due to overflowing per-cpu queue:

```
sysctl -w net.core.netdev_max_backlog=10000
```

- Prevent Neighbor (ARP) table overflows

```
sysctl -w net.ipv4.neigh.default.gc_thresh1=1024
sysctl -w net.ipv4.neigh.default.gc_thresh2=2048
sysctl -w net.ipv4.neigh.default.gc_thresh3=4096
```

dmesg: neighbour: arp_cache: neighbor table overflow!

Increasing the nofiles limit for ovs-vswitchd

- Ovs-vswitchd creates many netlink sockets to communicate with the kernel. 112K sockets in our set-up.
- Each netlink socket requires a file descriptor.
- The value of nofiles for the ovs-vswitchd must be at least: (Number of cpus * Number of switch ports).

```
$ prlimit -p \
 `cat /var/run/openvswitch/ovs-vswitchd.pid` \
--nofile=200000
```

In Summary.....

- Throughput becomes a function of CPU (loadaverage).
- Consider how contention for CPU will affect throughput.
- Massive layer 2 domains will have a large broadcast latency.
- I saw a decreased latency when using multiple bridges.

Legal Statement

This work represents the view of the authors and does not necessarily represent the view of IBM.

IBM is a registered trademark of International Business Machines Corporation in the United States and/or other countries.

Linux is a registered trademark of Linus Torvalds.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, and service names may be trademarks or service marks of others.