HKG18-415: TSN and Linux

Khoronzhuk Ivan
What is TSN and Intentions?

- TSN is the IEEE802.1Q defined standard technology to provide deterministic messaging on standard Ethernet. TSN technology is centrally managed and delivers guarantees of delivery and minimized jitter using time scheduling for those real-time applications that require determinism.

- Generic configuration interfaces when possible
  - Linux kernel, rich open source community, thus review
  - Allows to reuse tools like ethtool, iproute2, tc, vconfig, bridge, lldp, etc
  - Use generic interfaces for driver configuration, netdev, ethtool, switchdev
  - No need to maintain out of tree drivers

- Main criteria - guaranteed delivery, best-effort coexists
Map socket traffic to network level priority used to identify class of traffic at the end
Done with socket SO_PRIORITY option
or
net_prio cgroups to map all sockets of application (allows to set priority for application)

Map network level priority to LK traffic class used to identify traffic class within LK
Done with MQPRIO Qdisc

Map LK traffic class to LK sw queue of used to identify tx queue within LK
Done with MQPRIO mapping
or
internally in the driver
with rdo_setup_tc for MQPRIO

Map sw LK queue to internal hw queues (dma) used to direct sw tx queues to hw tx queues
internally in the driver
usually 1 to 1,
each channel with separate set of descs
Map network level prio to L2 level prio (8021P)
or map network level prio to destination MAC address
used to identify class of traffic within network
(including internal h/w switch)

set vlan egress priority
SI0CSIF/VLAN ioctl
SET_VLAN_EGRESS_PRIORITY_CMD
vconfig set_egress_map
(test with cat /proc/net/vlan/eth0.X)
specific configuration in h/w binding prio to MAC addresses

Map L2 level prio (8021P) to TSN class of traffic
used to split traffic in the TSN capable network

Based on switchdev
Allows to map traffic between switch ports
On receive end point can be mapped in backward order based on vlan tag
Ingress traffic to rx queue mapping

**PCP VLAN**
- pcp_vlan_tag -> rx queue
- Done with ethtool_ntuple_vlan_filtering
- ||
- coupled rx/tx queues

**Mapping**
- Mapping is done by packet itself
- + RSS and RPS mapping if needed
tx shaper (bandwidth) configuration

Set rates for appropriate tx queues of interface. An interface can use it in different purposes.

Done with:
```
echo R > /sys/class/net/eth0/queues/tx-N/tx_maxrate
```
Queues rates are accumulated to tc according map.

Set rates for classes of traffic
*In most cases needed to solve shaping on link level*

**TC rates from host side**
Can set with CBS Qdisc offload using .ndo_setup_tc
```
TC_SETUP_QDISC_MQPRIO type
```
 || any TSN related shaper, TAS, TBS, CQ..

**OR**

**TC rates from other ports (switch mode)**
should be accumulated rate from other ports
Different variants of traffic pinning and shapers conf

**Socket to queue pinning**
- tc options replace dev env4 handle: handle 100:1
- parent net info: queue_num: 10
- map: 221022222222222222222
- queues 1:0:1:0:1:0:1

**Traffic type to queue pinning**
- tc or iptables

**Application to queue pinning**
- net prio cgroups
- allow set priority for every socket from application

**Rest of traffic via LK stack**

**TSN traffic bypass LK TCP/IP stack**
- can be configured with vfi
- see: https://www.kernel.org/doc/Documentation/vfio.txt

$L$ tc options replace dev env4 parent 100:1
- cbs sockd: -1474 honest 30
- pktsk queue: -298300000 idle skq 298300000

socket(s), set priority option, to choose class of traffic

$s:♃_priority

socket, skb queues

socket(s), set priority option, to choose class of traffic

TNS related network scheduler,
CBS, TAS
offload possible

Kernel space

if vfi

Cisco Qdisc
- cbs qdisc
- tc 0.7
- q7
- q0

Rate limit configuration for interface tx queues.
Several queues can be part of same traffic class, and rate set with:
echo R > /sys/class/net/eth0/queues/tx-nh/maxrate
Driver also can decide how to map queues to tc

Bandwidth configuration for tc.
- It has to be set to sum of input queues rates.
- Can set with CBS Qdisc offload using
- _ndo_setup_to TC_SETUP_QDISC_CBS type

In switch mode has to take into account path thru traffic from other ports. So shaping should be set for each port, that's possible with switchdev exposing each port as network

Linaro connect
Hong Kong 2018
TSSDN model
Challenges

- Scale impact, a lot of control path messages, LLDP, PTP, NETCONF
- Only secure access to the network and secure configuration
  - mistiming is forbidden
  - topology issue can lead to incorrect latency
  - data plane flooding, sophisticated rx policy
- Network redundancy, can be solved in h/w at NIC level, but configuration
  - Separate netspace from best effort traffic
  - RSS, CPU pinning
  - Bypass some part of stack, eBFP
  - XDP eBPF and AF_XDP (still under development)
State, plans

- TSN yocto layer +
  - Last in kernel changes
  - Package group needed for TSN system configuration
  - In kernel changes (CBS, TAS, TBS, optimization)
  - System configuration changes, documentation
  - Driver tsn extensions for shaper configuration, switchdev..

- More latency measurements
- LAVA environment for TSN based on yocto layer
- Mininet configuration instead of h/w? For TSSDN test purposes only
- Identifying data model entries: latency, jitter, tc, bw, route and others
State, plans for TI parts

- cpsw CBS shaper patches (need only upstream)
- netcp multiqueue patches (need only upstream)
- netcp Ethtool ntuples configuration (need add support)
- cpsw ethtool ntuples configuration (need add support)
- cpsw add coupled queues and ingress configuration
- cpsw swithdev support (not sure if need, but should be)
- Test TAS shapers support
- extend tisdk, probably
● TAS (time aware shaper) mainline activity
  ○ [https://patchwork.kernel.org/patch/10029815/](https://patchwork.kernel.org/patch/10029815/)
● TBS (time based packet scheduler)
  ○ [https://lwn.net/Articles/744797/](https://lwn.net/Articles/744797/)
● AF_XDP (one more try to avoid SKB allocation and apply ZC)
  ○ [https://lwn.net/Articles/745934/](https://lwn.net/Articles/745934/)
● Latency measurements for am572, netcp, x86_64 i210
  ○ [https://projects.linaro.org/secure/attachment/12934/TSN%20measurements%20am572x%20k2g.pdf](https://projects.linaro.org/secure/attachment/12934/TSN%20measurements%20am572x%20k2g.pdf)
  ○ [https://git.linaro.org/people/ivan.khoronzhuk/tsn_latencies.git/tree/](https://git.linaro.org/people/ivan.khoronzhuk/tsn_latencies.git/tree/)
  ○ [https://projects.linaro.org/secure/attachment/13379/x86_lat.pdf](https://projects.linaro.org/secure/attachment/13379/x86_lat.pdf)
● Cpsw, netcp mq, cbs shaper configuration
  ○ [https://git.linaro.org/people/ivan.khoronzhuk/tsn_kernel.git/log/?h=cpsw_avb_v1](https://git.linaro.org/people/ivan.khoronzhuk/tsn_kernel.git/log/?h=cpsw_avb_v1)
  ○ [https://git.linaro.org/people/ivan.khoronzhuk/tsn_kernel.git/log/?h=mq_netcp_v6](https://git.linaro.org/people/ivan.khoronzhuk/tsn_kernel.git/log/?h=mq_netcp_v6)
Related base protocols

- IEEE 802.1Qbu and IEEE 802.3br Frame Preemption
- IEEE Std 802.1Qbv-2015 Enhancements for Scheduled Traffic
- IEEE Std 802.1Qca-2015 Path Control and Reservation
- IEEE 802.1Qcc Central configuration, enhancements and perf improvements
- IEEE 802.1Qci Time-based ingress policing
- IEEE 802.1CB
- IEEE Std 802.1AS-2011 Timing and Synchronization, gPTP
- IEEE Std 802.1Qat-2010 Stream Reservation Protocol (SRP)
- IEEE Std 802.1Qav-2009 Forwarding and Queueing Enhancements (for CBS)
- IEEE Std 802.1BA-2009 Audio Video Bridging (AVB) Systems
Thank You

#HKG18
HKG18 keynotes and videos on: connect.linaro.org
For further information: www.linaro.org, ivan.khoronzhuk@linaro.org