Vector Packet Processing(VPP) IPSEC Implementation in LSDK 20.12

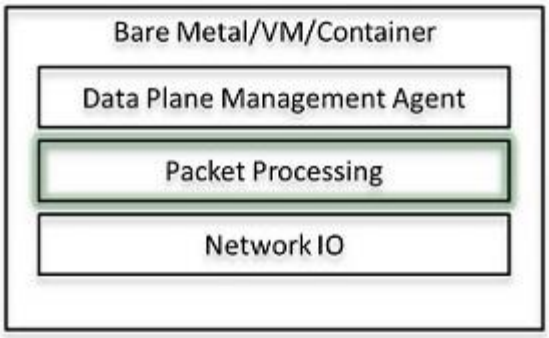
This document introduces Vector Packet Processing(VPP), creating VPP IPsec configuration scripts, building VPP v20.05 in LSDK 20.12, executing VPP IPsec on LS1046ARDB and LS2088ARDB platforms.

VPP introduction

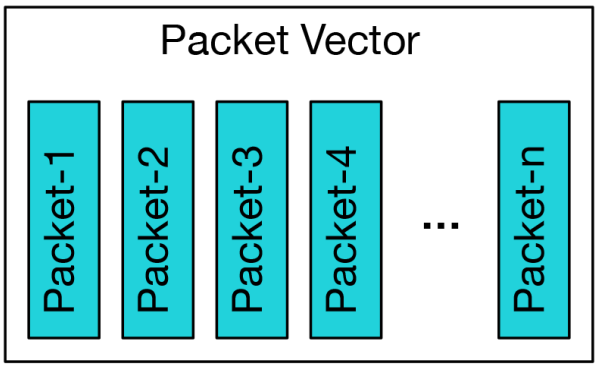
The VPP platform is an extensible framework that provides out-of-the-box production quality switch/router functionality. It is the open source version of Vector Packet Processing (VPP) technology: a high performance, packet-processing stack that can run on commodity CPUs.

The benefits of this implementation of VPP are its high performance, proven technology, its modularity and flexibility, and rich feature set.

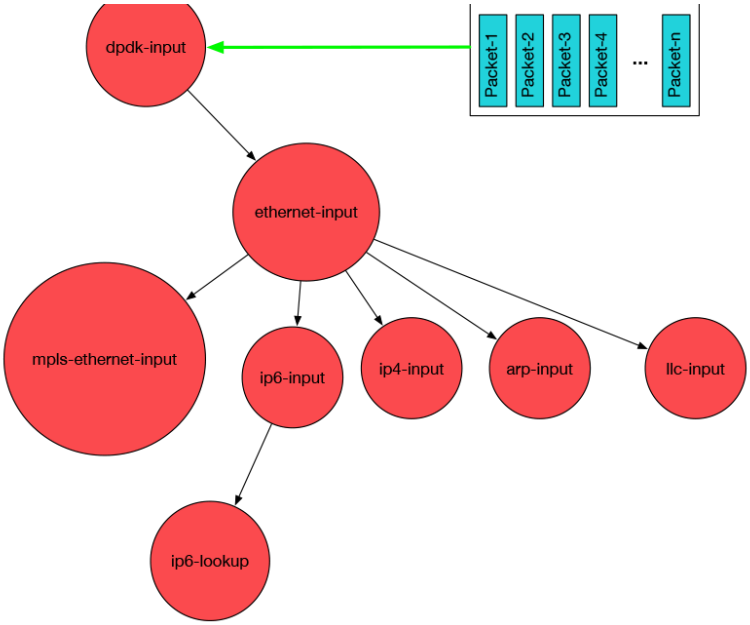
It is a modular design. The framework allows anyone to "plug in" new graph nodes without the need to change core/kernel code.



VPP reads the largest available vector of packets from the network IO layer.



VPP then processes the vector of packets through a Packet Processing graph.



Rather than processing the first packet through the whole graph, and then the second packet through the whole graph, VPP instead processes the entire vector of packets through a graph node before moving on to the next graph node.

Because the first packet in the vector warms up the instruction cache, the remaining packets tend to be processed at extreme performance. The fixed costs of processing the vector of packets are amortized across the entire vector. This leads not only to very high performance, but also statistically reliable performance. If VPP falls a little behind, the next vector contains more packets, and thus the fixed costs are amortized over a larger number of packets, bringing down the average processing cost per packet, causing the system to catch up. As a result, throughput and latency are very stable. If multiple cores are available, the graph scheduler can schedule (vector, graph node) pairs to different cores.

The graph node architecture of VPP also makes for easy extensibility. You can build an independent binary plugin for VPP from a separate source code base (you need only the headers). Plugins are loaded from the plugin directory. A plugin for VPP can rearrange the packet graph and introduce new graph nodes. This allows new features to be introduced via the plugin, without needing to change the core infrastructure code.

VPP IPSEC Use Case

VPP can perform the IPsec in “protocol offload” and “non-protocol offload” modes. DPDK support below 3 crypto drivers that VPP can use for IPsec:

ARMv8 crypto driver

OpenSSL Crypto driver

DPAA Sec driver

VPP IPSEC configuration scripts:

```
(board1) left.sh
#!/bin/bash

BOARD=$1
PORT=$2
if [ $BOARD == "LS1043ARDB" ]
then
    INT0=`vppctl show int |grep -oE ".*Ethernet0"`
    INT1=`vppctl show int |grep -oE ".*Ethernet1"`
    TEN0=`vppctl show int |grep -oE ".*Ethernet6"`
elif [ $BOARD == "LS1046ARDB" ]
then
    INT0=`vppctl show int |grep -oE ".*Ethernet0"`
    INT1=`vppctl show int |grep -oE ".*Ethernet1"`
    TEN0=`vppctl show int |grep -oE ".*Ethernet4"`
elif [ $BOARD == "LS2088ARDB" ] || [ $BOARD == "LS1088ARDB" ] || [ $BOARD ==
"LX2160ARDB" ]
then
    INT0=`vppctl show int |grep -oE ".*Ethernet0"`
    INT1=`vppctl show int |grep -oE ".*Ethernet1"`
fi

if [ $PORT == "2" ]
then
    vppctl set interface ip address $INT0 1.1.1.2/24
    vppctl set interface ip address $INT1 192.168.100.2/24
    vppctl set interface state $INT0 up
    vppctl set interface state $INT1 up
    vppctl ipsec sa add 10 spi 1001 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.2 tunnel-dst
192.168.100.3
    vppctl ipsec sa add 11 spi 1002 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.3 tunnel-dst
192.168.100.2
    vppctl ipsec spd add 1
    vppctl set interface ipsec spd $INT1 1
    vppctl set interface promiscuous on $INT1
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 10 local-ip-
range 1.1.1.3 - 1.1.1.3 remote-ip-range 2.1.1.3 - 2.1.1.3
```

```
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 11 local-ip-
range 1.1.1.3 - 1.1.1.3 remote-ip-range 2.1.1.3 - 2.1.1.3
    vppctl ip route add count 1 2.1.1.3/32 via 192.168.100.3 $INT1
    vppctl set ip arp static $INT1 192.168.100.3 00:22:22:22:22:23
    vppctl set ip arp static $INT0 1.1.1.3 00:22:22:22:22:28
    vppctl ipsec policy add spd 1 priority 100 inbound action bypass protocol 50
    vppctl ipsec policy add spd 1 priority 100 outbound action bypass protocol 50
elif [ $PORT == "4" ]
then
   vppctl set interface ip address $INT0 1.1.1.2/24
   vppctl set interface ip address $INT1 10.1.1.2/24
   vppctl set interface ip address $TEN0 192.168.100.2/24
   vppctl set interface state $INT0 up
   vppctl set interface state $INT1 up
   vppctl set interface state $TEN0 up
   vppctl ipsec sa add 10 spi 1001 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.2 tunnel-dst
192.168.100.3
   vppctl ipsec sa add 11 spi 1002 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.3 tunnel-dst
192.168.100.2
   vppctl ipsec spd add 1
   vppctl set interface ipsec spd $TEN0 1
   vppctl set interface promiscuous on $TEN0
   vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 10 local-ip-
range 1.1.1.3 - 1.1.1.3 remote-ip-range 2.1.1.3 - 2.1.1.3
   vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 11 local-ip-range
1.1.1.3 - 1.1.1.3 remote-ip-range 2.1.1.3 - 2.1.1.3
   vppctl ip route add count 1 2.1.1.3/32 via 192.168.100.3 $TEN0
   vppctl set ip arp static $TEN0 192.168.100.3 00:22:22:22:22:23
   vppctl set ip arp static $INT0 1.1.1.3 00:22:22:22:22:28
   vppctl ipsec policy add spd 1 priority 100 inbound action bypass protocol 50
   vppctl ipsec policy add spd 1 priority 100 outbound action bypass protocol 50
   vppctl ipsec sa add 40 spi 2001 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.4 tunnel-dst
192.168.100.5
   vppctl ipsec sa add 41 spi 2002 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
```

```
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.5 tunnel-dst 192.168.100.4
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 40 local-ip-range 1.1.1.3 - 1.1.1.3 remote-ip-range 2.1.1.4 - 2.1.1.4
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 41 local-ip-range 1.1.1.3 - 1.1.1.3 remote-ip-range 2.1.1.4 - 2.1.1.4
    vppctl ip route add count 1 2.1.1.4/32 via 192.168.100.3 $TEN0
    vppctl set ip arp static $TEN0 192.168.100.5 00:22:22:22:22:33
    vppctl set ip arp static $INT0 1.1.1.4 00:22:22:22:22:38
    vppctl ipsec sa add 50 spi 3001 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.2 tunnel-dst 192.168.100.3
    vppctl ipsec sa add 51 spi 3002 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.3 tunnel-dst 192.168.100.2
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 50 local-ip-range 10.1.1.3 - 10.1.1.3 remote-ip-range 20.1.1.3 - 20.1.1.3
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 51 local-ip-range 10.1.1.3 - 10.1.1.3 remote-ip-range 20.1.1.3 - 20.1.1.3
    vppctl ip route add count 1 20.1.1.3/32 via 192.168.100.3 $TEN0
    vppctl set ip arp static $INT1 10.1.1.3 00:22:22:22:22:38
    vppctl ipsec sa add 70 spi 4001 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.4 tunnel-dst 192.168.100.5
    vppctl ipsec sa add 71 spi 4002 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.5 tunnel-dst 192.168.100.4
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 70 local-ip-range 10.1.1.3 - 10.1.1.3 remote-ip-range 20.1.1.4 - 20.1.1.4
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 71 local-ip-range 10.1.1.3 - 10.1.1.3 remote-ip-range 20.1.1.4 - 20.1.1.4
    vppctl ip route add count 1 20.1.1.4/32 via 192.168.100.3 $TEN0
    vppctl set ip arp static $INT1 10.1.1.4 00:22:22:22:22:38
fi
vppctl show ip arp
vppctl show hardware-int
```

```
(board2) right.sh
#!/bin/bash

BOARD=$1
PORT=$2
if [ $BOARD == "LS1043ARDB" ]
then
    INT0=`vppctl show int |grep -oE ".*Ethernet0"`
    INT1=`vppctl show int |grep -oE ".*Ethernet1"`
    TEN0=`vppctl show int |grep -oE ".*Ethernet6"`
elif [ $BOARD == "LS1046ARDB" ]
then
    INT0=`vppctl show int |grep -oE ".*Ethernet0"`
    INT1=`vppctl show int |grep -oE ".*Ethernet1"`
    TEN0=`vppctl show int |grep -oE ".*Ethernet4"`
elif [ $BOARD == "LS2088ARDB" ] || [ $BOARD == "LS1088ARDB" ] || [ $BOARD ==
"LX2160ARDB" ]
then
    INT0=`vppctl show int |grep -oE ".*Ethernet0"`
    INT1=`vppctl show int |grep -oE ".*Ethernet1"`
fi

if [ $PORT == "2" ]
then
    vppctl set interface ip address $INT0 2.1.1.2/24
    vppctl set interface ip address $INT1 192.168.100.3/24
    vppctl set interface state $INT0 up
    vppctl set interface state $INT1 up
    vppctl ipsec sa add 20 spi 1001 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.2 tunnel-dst
192.168.100.3
    vppctl ipsec sa add 21 spi 1002 esp crypto-alg aes-cbc-128 crypto-key
4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.3 tunnel-dst
192.168.100.2
    vppctl ipsec spd add 1
    vppctl set interface ipsec spd $INT1 1
    vppctl set interface promiscuous on $INT1
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 20 local-ip-
range 2.1.1.3 - 2.1.1.3 remote-ip-range 1.1.1.3 - 1.1.1.3
```

```
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 21 local-ip-range 2.1.1.3 - 2.1.1.3 remote-ip-range 1.1.1.3 - 1.1.1.3
    vppctl ip route add count 1 1.1.1.3/32 via 192.168.100.2 $INT1
    vppctl set ip arp static $INT1 192.168.100.2 00:22:22:22:22:24
    vppctl set ip arp static $INT0 2.1.1.3 00:22:22:22:22:25
    vppctl ipsec policy add spd 1 priority 100 inbound action bypass protocol 50
    vppctl ipsec policy add spd 1 priority 100 outbound action bypass protocol 50
elif [ $PORT == "4" ]
then
    vppctl set interface ip address $INT0 2.1.1.2/24
    vppctl set interface ip address $INT1 20.1.1.2/24
    vppctl set interface ip address $TEN0 192.168.100.3/24
    vppctl set interface state $INT0 up
    vppctl set interface state $INT1 up
    vppctl set interface state $TEN0 up
    vppctl ipsec sa add 20 spi 1001 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.2 tunnel-dst 192.168.100.3
    vppctl ipsec sa add 21 spi 1002 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.3 tunnel-dst 192.168.100.2
    vppctl ipsec spd add 1
    vppctl set interface ipsec spd $TEN0 1
    vppctl set interface promiscuous on $TEN0
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 20 local-ip-range 2.1.1.3 - 2.1.1.3 remote-ip-range 1.1.1.3 - 1.1.1.3
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 21 local-ip-range 2.1.1.3 - 2.1.1.3 remote-ip-range 1.1.1.3 - 1.1.1.3
    vppctl ip route add count 1 1.1.1.3/32 via 192.168.100.2 $TEN0
    vppctl set ip arp static $TEN0 192.168.100.2 00:22:22:22:22:24
    vppctl set ip arp static $INT0 2.1.1.3 00:22:22:22:22:25
    vppctl ipsec policy add spd 1 priority 100 inbound action bypass protocol 50
    vppctl ipsec policy add spd 1 priority 100 outbound action bypass protocol 50
    vppctl ipsec sa add 30 spi 2001 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.4 tunnel-dst 192.168.100.5
    vppctl ipsec sa add 31 spi 2002 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key
```

```
4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.5 tunnel-dst 192.168.100.4
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 30 local-ip-range 2.1.1.3 - 2.1.1.3 remote-ip-range 1.1.1.4 - 1.1.1.4
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 31 local-ip-range 2.1.1.3 - 2.1.1.3 remote-ip-range 1.1.1.4 - 1.1.1.4
    vppctl ip route add count 1 1.1.1.4/32 via 192.168.100.2 $TEN0
    vppctl set ip arp static $TEN0 192.168.100.4 00:22:22:22:22:44
    vppctl set ip arp static $INT0 2.1.1.4 00:22:22:22:22:55
    vppctl ipsec sa add 60 spi 3001 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.2 tunnel-dst 192.168.100.3
    vppctl ipsec sa add 61 spi 3002 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.3 tunnel-dst 192.168.100.2
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 60 local-ip-range 20.1.1.3 - 20.1.1.3 remote-ip-range 10.1.1.3 - 10.1.1.3
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 61 local-ip-range 20.1.1.3 - 20.1.1.3 remote-ip-range 10.1.1.3 - 10.1.1.3
    vppctl ip route add count 1 10.1.1.3/32 via 192.168.100.2 $TEN0
    vppctl set ip arp static $INT1 20.1.1.3 00:22:22:22:22:55
    vppctl ipsec sa add 80 spi 4001 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.4 tunnel-dst 192.168.100.5
    vppctl ipsec sa add 81 spi 4002 esp crypto-alg aes-cbc-128 crypto-key 4a506a794f574265564551694d653768 integ-alg sha1-96 integ-key 4339314b55523947594d6d3547666b45764e6a58 tunnel-src 192.168.100.5 tunnel-dst 192.168.100.4
    vppctl ipsec policy add spd 1 priority 10 inbound action protect sa 80 local-ip-range 20.1.1.3 - 20.1.1.3 remote-ip-range 10.1.1.4 - 10.1.1.4
    vppctl ipsec policy add spd 1 priority 10 outbound action protect sa 81 local-ip-range 20.1.1.3 - 20.1.1.3 remote-ip-range 10.1.1.4 - 10.1.1.4
    vppctl ip route add count 1 10.1.1.4/32 via 192.168.100.2 $TEN0
    vppctl set ip arp static $INT1 20.1.1.4 00:22:22:22:22:55
fi
vppctl show ip arp
vppctl show hardware-int
```

Build VPP v20.05 in LSDK 20.12

1. Download / Clone Flexbuild for LSDK 20.12 and setup.

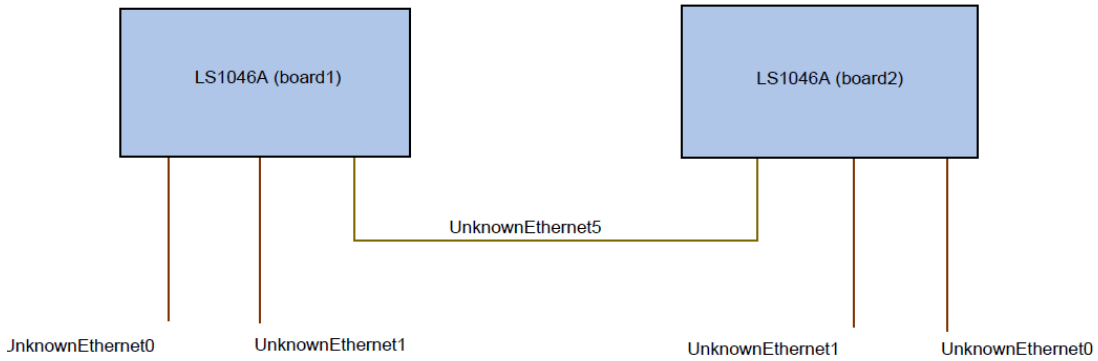
2. Enable VPP compilation in Flexbuild by doing following change:

```
diff --git a/configs/build_lsdk_internal.cfg b/configs/build_lsdk_internal.cfg
index 5413bc2..24efc3b 100644
--- a/configs/build_lsdk_internal.cfg
+++ b/configs/build_lsdk_internal.cfg
@@ -30,7 +30,7 @@ CONFIG_APP_SPC=y
 CONFIG_APP_CST=y
 CONFIG_APP_OPENSSL=y
 CONFIG_APP_DPDK=y
-CONFIG_APP_VPP=n
+CONFIG_APP_VPP=y
 CONFIG_APP_OVS_DPDK=y
 CONFIG_APP_PKTGEN_DPDK=y
 CONFIG_APP_AIOPSL=y
```

3. Now run following commands on prompt
   flexbuild$ source setup.env
   flexbuild$ flex-builder -c vpp -a arm64 ## This will clone the VPP repo
4. The above commands will clone the VPP repo,if already there, and start compilation. Stop the compilation using "CTRL+C"
5. Now run following commands to update VPP codebase.
   flexbuild$ cd packages/apps/networking/vpp/
   vpp$ git branch -a
   vpp$ git checkout -b v2005 remotes/origin/v2005
   vpp$ cd –
6. Build again using "***flex-builder -c vpp -a arm64***"
7. Debian Images can be found in **./build/rfs/rootfs_lsdk2012_ubuntu_main_arm64/usr/local/vpp/**

Execute VPP IPsec on LS1046RDB board

This setup will use two 1g port and one 10g port of both boards.1g ports should be connected to the Spirent whereas 10g ports of both boards should be connected back to back.



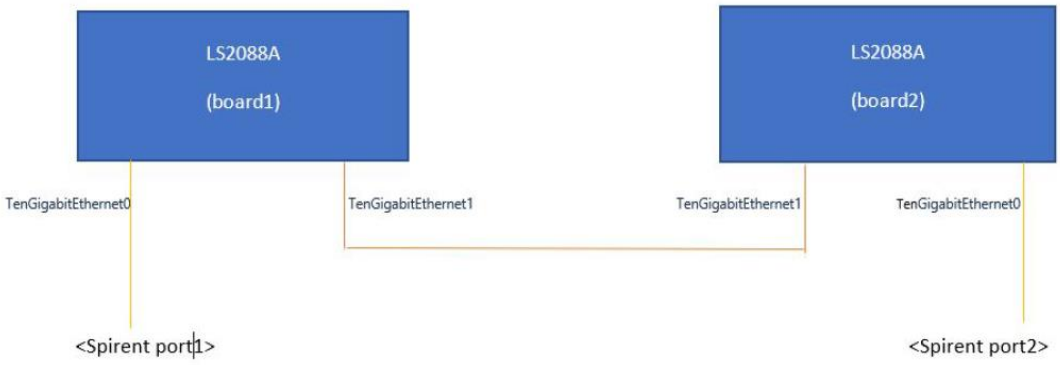
 # mkdir /mnt/hugepages
 # mount -t hugetlbfs none /mnt/hugepages
 # echo 256 > /proc/sys/vm/nr_hugepages
 # fmc -x
 # export DPAA_NUM_RX_QUEUES=1
 # cd /usr/local/dpdk/dpaa
 # fmc -c usdpaa_config_ls1046.xml -p usdpaa_poolicy_hash_ipv4_1queue.xml -a
 # cd –
 # vpp -c /etc/vpp/startup.conf.dpkg-new &
 # source /left.sh LS1046ARDB 2 (board1)
#source /right.sh LS1046ARDB 2(board2)

Configure traffic from the Spirent ports as the below:

Board1				Board2			
Port	Flow	SRC-IP	DEST-IP	Port	Flow	SRC-IP	DEST-IP
Unknown Eth0	1	1.1.1.3	2.1.1.3	Unknown Eth0	1	2.1.1.3	1.1.1.3
	2	1.1.1.3	2.1.1.4		2	2.1.1.3	1.1.1.4
Unknown Eth1	1	10.1.1.3	20.1.1.3	Unknown Eth1	1	20.1.1.3	10.1.1.3
	2	10.1.1.3	20.1.1.4		2	20.1.1.3	10.1.1.4

Execute VPP IPsec on LS2088ARDB board

This setup will use two 10G ports of both boards. One of the 10G ports should be connected to the Spirent whereas another 10g ports of both boards should be connected back-to-back.




```
# dpkg --get-selections 'vpp*'
# source /usr/local/dpdk/dpaa2/dynamic_dpl.sh dpmac.1 dpmac.2
# vpp -c /etc/vpp/startup.conf.dpkg-new &
# mkdir /mnt/hugepages
# mount -t hugetlbfs none /mnt/hugepages
# echo 256 > /proc/sys/vm/nr_hugepages
# source /left.sh LS2088ARDB 2 (baord1)
# source /right.sh LS2088ARDB 2 (board2)
```

Configure traffic from the Spirent ports as the below.

Board1				Board2			
Port	Flow	SRC-IP	DEST-IP	Port	Flow	SRC-IP	DEST-IP
Unknown Eth0	1	1.1.1.3	2.1.1.3	Unknown Eth0	1	2.1.1.3	1.1.1.3
	2	1.1.1.3	2.1.1.4		2	2.1.1.3	1.1.1.4
Unknown Eth1	1	10.1.1.3	20.1.1.3	Unknown Eth1	1	20.1.1.3	10.1.1.3
	2	10.1.1.3	20.1.1.4		2	20.1.1.3	10.1.1.4