Inbuilt Traffic Generator for Network Diagnostics

A traffic generator is a tool that is used to

- inject traffic onto a network for other devices to consume
- test the network to optimize performance, and
- troubleshoot network issues.

The Cisco 8000 Series Fixed Routers and Line Cards provides an inbuilt traffic generator in the Network Processing Unit (NPU) . This chapter shows you how to set up and run the inbuilt traffic generator.

Overview of the Inbuilt Traffic Generator

Table 1: Feature History Table

Feature Name	Release Information	Description
Inbuilt Traffic Generator for Network Diagnostics	Release 24.2.11	By introducing an inbuilt traffic generator in the Network Processing Unit (NPU) of line cards (LCs) of distributed systems and route processors (RPs) of fixed routers, we've ensured that the traffic generator is always available for network diagnostics. You also don't face compatibility issues because the traffic generator is inbuilt and easy to maintain. Previously, connecting an external traffic generator was necessary to inject packets to test networks. This feature introduces these changes: **CLI:** • **diagnostic packet-generator create** • **diagnostic packet-generator start** • **diagnostic packet-generator stop** • **diagnostic packet-generator delete** • **show diagnostic packet-generator status**

The inbuilt traffic generator is implemented within the Cisco Silicon One Application-Specific Integrated Circuit (ASIC), the Network Processing Unit (NPU) of linecards, and fixed routers of the Cisco 8000 Series Routers.

Caution Don’t run the inbuilt traffic generator on a live network unless you are fully aware of the impact of packets injected. Injecting packets into a live network may result in network outages.

The inbuilt traffic generator works in three modes:

- **Ingress mode:** In this mode, the traffic generator injects ingress packets to an interface as if received from an upstream router. The NPU processes these packets normally and sends them to the egress interface based on looking up the Forwarding Information Base (FIB) programmed in the NPU. The NPU may encapsulate the packet and rewrite the packet headers before sending the packet out of the egress interface.
- **Egress mode:** In this mode, the traffic generator injects packets directly to an egress interface. If the egress interface is on another line card, the router sends the injected packet through the fabric and to another NPU where the egress interface is located. The router sends these packets as-is from the egress interface. So, you must be careful to formulate these packets with a valid Layer 2 header so that the downstream routers can process the packets.
- **Raw mode:** This mode is an advanced traffic generator mode where the user has full control. In this mode, the user can control the packets within the NPU by defining the inject headers of the packet.

Caution Improper use of raw mode could cause unexpected behavior, such as NPU lock-up. It is recommended that only Cisco engineers create traffic generator instances in raw mode.

Restrictions of the Inbuilt Traffic Generator

The following restrictions apply to the inbuilt traffic generator:

- The traffic generator does not support Link Layer Topology Discovery (LLTD) protocol for a scapy packet.
- The packet counters displayed in the show command outputs of the traffic generator CLIs does not increment if the router drops the packet.
- On linecards and fixed routers with Q100 based Silicon One ASICs, the traffic generator does not support capturing ingress packets.
- The inbuilt traffic generator does not retain the instance or the data if the line card or slot gets reloaded.
- When running the traffic generator in ingress mode, the **input packets** counter does not reflect an increase in the **show interface** command output for the ingress interface. However, the **input packets** counter of the **show interface accounting** command output does show an increment for the same ingress interface.
- Conversely, in egress mode, the **output packets** counter is incremented in the **show interface** command output for the egress interface. But the **output packets** counter of the **show interface accounting** command output does not display this increment for the same egress interface.

Guidelines for the Inbuilt Traffic Generator

Follow these guidelines while using the inbuilt traffic generator:

- You can create and control multiple traffic generator sessions at the same time, each with a different name.

- You can turn on packet capture when you create the traffic generator session. In this case, the router punts the traffic packets to the CPU instead of sending them out on the wire.
- All variations of the Cisco Silicon One ASIC support this inbuilt traffic generator.
- Per NPU, you can configure a maximum of 585 flows with packet size smaller or equal to 112 bytes and upto 8 flows for larger packets.
- You can configure a maximum traffic rate of 6.8 Mpps or 33.2 Gbps.
- You can configure upto 8 different configurable transmit rates shared across flows, that is, from 1 pps to 6.8 Mpps. Since the hardware doesn't support every single rate from 1 to 6.8Mpps, it will round off any unsupported rate you configured to the nearest hardware rate, with a maximum difference of 2%. You can view the rate applied in hardware with the **show diagnostic packet-generator status** command.
- You can configure a maximum packet size of 608 bytes on line cards and routers with the Q100 or Q200 based Silicon One ASICs.
- You can view per flow transmit statistics.
- The traffic generator can inject packets in egress, or ingress, or both directions at the same time.

Guidelines to Define Packets for the Traffic Generator

Few guidelines to keep in mind while defining packets to be generated by the traffic generator:

- You can define packets either at command line or within a file. The maximum length for packets at command line is 255 characters. For larger packets, define the packets within a file.
- While defining the packet within a file, you can use a pcap (packet capture) file with a .pcap suffix, or a text file.

  Pcap is a common format for storing packet captures. A pcap file includes an exact copy of every byte of every packet as seen on the network, including OSI layers 2-7.

  The packets defined in a text file can either be a hexadecimal string or as a python scapy script. Scapy is a packet manipulation tool, written in python, that can forge or decode packets, send them on the wire, capture them, and so on.
- Ensure to remove white-spaces and line-breaks from the hexadecimal or scapy specification of the packet.

Set-up and Run the Inbuilt Traffic Generator

The following steps show you how to set-up and run the inbuilt traffic generator:

1. **Create traffic generator instance:** In this step, you'll create the traffic generator instance with the required rate, duration, traffic generator mode, packet format, location, and so on, using the **diagnostic packet-generator create** command.

   By default, packet capture is turned off. You can turn on packet capture in this step using the optional keyword **capture**.

   The following examples show you how to create the traffic generator instance in ingress and egress mode:

   - **Ingress mode:**

```
Router# diagnostic packet-generator create t1 rate 100 duration 60 packet
IP(src="32.0.0.1",dst="22.0.0.1",ttl=64)/UDP()/Raw(load="a"*100) ingress interface
FourHundredGigE0/0/0/1 capture location 0/RP0/CPU0
OK
```

- **Egress mode:**

```
Router# diagnostic packet-generator create t1 rate 100 duration 60 packet
Ether(src="A:B:C:D:E:F",dst="1:2:3:4:5:6")/IP(src="32.0.0.1",dst="109.0.0.101",ttl=64)/Raw(load="f"*100)
 egress interface fourHundredGigE0/0/0/0 capture location 0/RP0/CPU0
OK
```

2. **Start traffic generator instance:** The traffic generator is in an inactive state until you execute the **diagnostic packet-generator start** command to start injecting packets.

```
Router# diagnostic packet-generator start t1 location 0/RP0/CPU0
OK
```

3. **Verify the status of the traffic generator:** Execute the **show diagnostic packet-generator status** command to view the traffic generator status and the packet generated.

```
Router# show diagnostic packet-generator status t1 location 0/RP0/CPU0
0/RP0/CPU0:
Name            Run_State    Type    Capture  Set_Rate(pps)  Applied_Rate(pps)
Duration(sec)  TC  Phy_Interface  NPU  Slice  IFG  Packets        Bytes
==========================================================================================
T1              Running      Ingress True     100            101                60
       0   FH0/0/0/1      0    4      1    209            45144
------------------------------------------------------------------------------------------
Packet Details:
###[ Ethernet ]###
  dst       = 78:bf:d2:07:10:08
  src       = 00:00:00:00:00:01
  type      = IPv4
###[ IP ]###
     version   = 4
     ihl       = 5
     tos       = 0x0
     len       = 128
     id        = 1
     flags     =
     frag      = 0
     ttl       = 64
     proto     = udp
     chksum    = 0x446b
     src       = 32.0.0.1
     dst       = 22.0.0.1
     \options   \
###[ UDP ]###
        sport     = domain
        dport     = domain
        len       = 108
        chksum    = 0xc3a5
###[ DNS ]###
           id        = 24929
           qr        = 0
           opcode    = 12
           aa        = 0
           tc        = 0
           rd        = 1
           ra        = 0
           z         = 1
           ad        = 1
```

```
           cd        = 0
           rcode     = format-error
           qdcount   = 24929
           ancount   = 24929
           nscount   = 24929
           arcount   = 24929
           qd        = ''
           an        = ''
           ns        = ''
           ar        = ''
###[ Raw ]###
              load      =
'aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa'
```

4. **View captured packet:** If you have enabled packet capture, the router punts the packets, that are ready to be sent out, to the CPU of the egress line card. You can view the captured packets with the **show captured packets** command. This CLI limits the display of the packet to 256 bytes.

```
Router# show captured packets traps all location 0/RP0/CPU0
Captured packets are also saved in /harddisk:/nputrap_pcap_traps_all_loc_RP0.txt
***** Packet 1 (len 236) *****
Captured at: 2024-01-17 18:51:33.486

Internal header + packet:
4e41500100014e41500100117102080000a0c00ffff003cbffff3ffff0000028e
0ca000000000000000004e41500100014e415001001181006632710208031 43c
000008ffff000030003c00001112131415161705000000000123456789abd00ca
fedead008100000007103262000000c00000000000000000000000000078bfd20710
08000000000000108004500008000010000401 1446b200000011600000100 3500
35006cc3a561616161616161616161616161616161616161616161616161616161
6161616161616161616161616161616161616161616161616161616161616161
616161616161616161616161

DMAC: 4e:41:50:01:00:01
SMAC: 4e:41:50:01:00:11
etype: NPU-PUNT
Punt header:
next_hdr: 0x1 (Ethernet)
fwd_hdr_type: 0x1 (Ethernet)
next header offset: 0
punt_source: 0xa (OUTBOUND_MIRROR)
Trap code: 12 (Refer to the traps output)
Ingress port details:
--------------------------
src_sys_port: 0xffff
src_logical_port: 0x3ffff
is_l2_punt: Yes
--------------------------
Egress port details:
--------------------------
dst_sys_port: 0x3c
dst_logical_port: 0x3ffff
--------------------------
switch_or_vrf_id: 0x0 (default)
timestamp: 0x28e0ca000000000
counterstamp: 0x0



Ethernet payload

4e41500100014e41500100118100663271020803143c000008ffff000030003c
```

```
000011121314151617050000000001234567 89abd00cafedead00810000007103
26200000c000000000000000000000000078bfd207100800000000000001080045
000080000100004011446b2000000116000001003500350 06cc3a56161616161
6161616161616161616161616161616161616161616161616161616161616161616161
6161616161616161616161616161616161616161616161616161616161616161616161
6161

DMAC: 4e:41:50:01:00:01
SMAC: 4e:41:50:01:00:11
etype: 802.1Q
802.1Q hdr [User field:03 CFI:00 VLAN-ID:1586]
etype: NPU-PUNT

***** Packet 1 end *****
```

5. **Stop and delete traffic generator instance:** After completing the traffic testing, execute the **diagnostic packet-generator stop** command to stop injecting packets. Following that, free up resources by deleting the traffic generator instance with the **diagnostic packet-generator delete** command.

```
Router# diagnostic packet-generator stop t1 location 0/RP0/CPU0
OK
Router# diagnostic packet-generator delete t1 location 0/RP0/CPU0
OK
```