Troubleshoot Router Issues on Enterprise Network

Contents

Introduction

This document describes how to identify, troubleshoot and resolve latency issues in Enterprise Networks using Cisco routers.

Prerequisites

Requirements

There are no specific prerequisites or requirements for this document.

Components Used

This document is not restricted to specific software version and hardware type, but commands are applicable to Cisco IOS® XE routers such as ASR 1000, ISR 4000 and Catalyst 8000 families.

The information in this document was created from the devices in a specific lab environment. All of the devices used in this document started with a cleared (default) configuration. If your network is live, ensure

that you understand the potential impact of any command.

Background Information

This document describes a basic guide to understand, isolate and troubleshoot general latency issues, gives useful commands/debugs to detect the root causes and best practices. Keep in mind that all possible variables and scenarios can not be considered and deeper analysis depends on specific situations.

Latency Definition

In general terms, and quoting the strict definition for store and forward devices (on RFC 1242), latency is the time interval starting when the last bit of the input frame reaches the input port and ending when the first bit of the output frame is seen on the output port.

Network latency can simply refers to delay on data transfer across the network. For practical issues, this definition is just the starting point; you need to define the latency problem you are talking about on every specific case, although it seems obvious, the first step needed to solve a problem, and becomes really important, is to define it.

Latency Usage

Many applications requires low latency for real-time communication and business operations; with the hardware and software improvements everyday, more applications are available for mission-critical computation, online meeting applications, streaming among others; in the same way, network traffic continues to grow and the need for optimized network designs and better devices performance increases as well.

Besides giving better user experience and deliver the minimum required for latency-sensitive applications, effectively identify and reduce latency issues on a network can save a lot of time and resources highly valuable on a network.

Approaching Latency Issues

The hard part of these type of issues is the number of variables you must have into consideration plus there can not be a single point of failure. Hence, definition of latency becomes an important key to solve it and some aspects you must take into consideration to have a useful problem description are the next ones.

1. Expectation and Detection

It is important to differentiate a desired latency, the expected or baseline working latency and the current one. Depending on design, providers or devices on the network, sometimes you can not achieve the desired latency, it is a good procedure to measure the real one under normal conditions but you need to be consistent on measurement methods to avoid misleading numbers; IP SLAs, and network analyzer tools can help on this regards.

One of the most used and basic tools to identify latency by applications or even IP SLA is via ICMP or ping:

<#root>

```
Router#
```

ping

```
 198.51.100.1
Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 198.51.100.1, timeout is 2 seconds:
!!!!!
Success rate is 100 percent (5/5),
round-trip min/avg/max
 =
2/109/541 ms
```

Besides checking reachability, ping tells the Round Trip Time (RTT) from source to destination; the minimum (2), average (109) and maximum (541) in milliseconds. This means, the duration from when the router sends the request to when it receives the reply from device destination. However, it does not show how many hops or deeper information but it is an easy and fast way to detect a problem.

2. Isolation

Same as ping, traceroute can be used as the starting point for isolation, it discovers hops and RTT per hop:

<#root>

```
Router#
traceroute
 198.51.100.1
Type escape sequence to abort.
Tracing the route to 198.51.100.1
VRF info: (vrf in name/id, vrf out name/id)
  1 10.0.3.1 5 msec 6 msec 1 msec
  2 10.0.1.1 1 msec 1 msec 1 msec
  3 10.60.60.1 1 msec 1 msec 1 msec
  4 10.90.0.2
362 msec 362 msec 362 msec
<<<< you can see the RTT of the three probes only on both hops
  5 10.90.1.2
363 msec 363 msec 183 msec
  6 10.90.7.7 3 msec 2 msec 2 msec
```

Traceroute operates by sending out a packet with a TimeTo Live (TTL) of 1. First hop sends back an ICMP error message indicating that the packet could not be forwarded because the TTL expired and RTT is measured, second packet is then resent with a TTL of 2, and the second hop returns the TTL expired. This process continues until the destination is reached.

On the example, you can now narrow down to two specific host and you can start from there on our isolation.

Despite these are useful commands that can easily identify a problem, they do not take into consideration

another variables such as protocols, packet markings and sizes (although you can set them as second step), different IP sources, destinations among multiple factors.

Saying latency can be a very broad concept and you often see only the symptom on an application, browsing, call or specific tasks. One of the first things to limit is to understand the impact and define the problem in more detail, answer the next questions and elements can help for this dimensioning:

- Does latency affect only specific kind of traffic or application? Example: Only UDP, TCP, ICMP...
- If so, does this traffic have unique identifiers? Example: Specific QoS marking, only determined packet sizes, IP options...
- How many users or sites are affected? Example: Only one specific subnet, one or two end hosts, a whole site connected to one or many devices...
- Is there specific timestamps identified? Example: does this occur only during peak hours, any time pattern or complete random...
- Design aspects. Example: Traffic passing through a specific device, maybe many devices but connecting to only one provider, traffic doing load balance but affected one path...

There are many other considerations but crossing the different answers (and even tests that can be done to answer them) can effectively isolate and limit the scope to proceed with the troubleshooting. As an example; only one application (same kind traffic) affected on all branches passing through different providers ending on the same data center at peak hours. On this case, you do not start checking all access switches in all branches, instead, you focus on collecting more information on the data center and inspect further on that side,

Monitoring tools and some automation you can have on the network helps a lot on this isolation as well, really depends on the resources you have and unique situations.

Troubleshoot Common Causes

Once you limit the scope of the troubleshoot, you can start checking specific causes, for instance, on the traceroute example provided, you can isolate to two different hops and then, narrow down to possible causes.

Platform Related

High CPU

One of the common causes can be a device with high CPU making delay on process all packets. For routers, the most useful and basic command to check on routers are

Overall performance for router:

<#root>

```
Router#
```

show platform resources

```
**State Acronym: H - Healthy, W - Warning, C - Critical
Resource                Usage                Max            Warning         Critical        State
-------------------------------------------------------------------------------------------------
RP0 (ok, active)                                                                            H
```

```
Control Processor       1.15%                100%            80%             90%             H


DRAM                    3631MB(23%)          15476MB         88%             93%             H

  bootflash               11729MB(46%)         25237MB         88%             93%             H
  harddisk                1121MB(0%)           225279MB        88%             93%             H
ESP0(ok, active)                                                                             H
 QFP                                                                                         H
  TCAM                    8cells(0%)           131072cells     65%             85%             H
  DRAM                    359563KB(1%)         20971520KB      85%             95%             H
  IRAM                    16597KB(12%)         131072KB        85%             95%             H

CPU Utilization         0.00%                100%            90%             95%             H


Crypto Utilization      0.00%                100%            90%             95%             H

  Pkt Buf Mem (0)         1152KB(0%)           164864KB        85%             95%             H
  Pkt Buf CBlk (0)        14544KB(1%)          986112KB        85%             95%             H
```

Useful to see memory and CPU utilization at once, it is divided on Control plane and Data plane (QFP) same as thresholds for each one. Memory itself, does not create a latency problem, however, if there is no more DRAM memory for control plane, Cisco Express Forwarding (CEF) is disabled and induces a high CPU usage which can produces latency, that is why it is important to keep numbers under healthy state. Basic guide for memory troubleshooting is out of the scope but refer useful link on Related information section.

If high CPU is detected for Control Processor, QFP CPU or Crypto utilization, you can use the next commands:

For control plane:

show process cpu sorted

<#root>

```
Router#
show processes cpu sorted
CPU utilization for five seconds:
99%/0%

; one minute: 13%; five minutes: 3%
 PID Runtime(ms)     Invoked      uSecs   5Sec   1Min   5Min TTY Process
  65        1621         638       2540 89.48%  1.82%  0.41%  0 crypto sw pk pro
   9         273          61       4475  1.56%  0.25%  0.05%   0 Check heaps
  51         212          64       3312  0.72%  0.21%  0.05%   0 Exec
 133         128          16       8000  0.60%  0.08%  0.01%   0 DBAL EVENTS
 473          25          12       2083  0.48%  0.04%  0.00%   0 WSMAN Process
  84        1173         353       3322  0.36%  0.07%  0.02%   0 IOSD ipc task
  87          23          12       1916  0.24%  0.02%  0.00%   0 PuntInject Keepa
```

```
  78         533          341       1563  0.12%  0.29%  0.07%   0 SAMsgThread
 225          25         1275         19  0.12%  0.00%  0.00%   0 SSS Feature Time
 386           4            4       1000  0.12%  0.00%  0.00%   0 Crypto WUI
 127         204        18810         10  0.12%  0.02%  0.00%   0 L2 LISP Punt Pro
```

If control plane CPU is high (this example is at 99% because of processes), need to isolate the process and, depends on it, proceed with isolation (can be punted packets for us like ARP or control network packets, can be any routing protocol, multicast, NAT, DNS, crypto traffic or any service).

Depends on your traffic flow, this can cause a problem on further processing, if traffic is not destined to the router you can focus on data plane:

For data plane:

show platform hardware qfp active datapath utilization [summary]

<#root>

```
Router#
```

```
show platform hardware qfp active datapath utilization
```

```
  CPP 0: Subdev 0
```

```
5 secs
```

```
        1 min        5 min       60 min
Input:  Priority (pps)            0            0            0            0
                 (bps)            0            0            0            0
    Non-Priority (pps)          231          192           68            6
                 (bps)       114616        95392        33920         3008
           Total (pps)          231          192           68            6
                 (bps)       114616        95392        33920         3008
Output: Priority (pps)            0            0            0            0
                 (bps)            0            0            0            0
    Non-Priority (pps)            3            2            2            0
                 (bps)        14896         9048         8968         2368
```

```
Total (pps)
```

```
         3323         2352          892            0
```

```
(bps)
```

```
        14896         9048         8968         2368
```

```
Processing: Load (pct)
```

```
3
```

```
            3            3            3
```

```
Crypto/IO
```

```
Crypto: Load (pct)           0
```

```
            0            0            0
```

```
RX: Load (pct)            0            0            0            0
TX: Load (pct)            1            1            0            0
    Idle (pct)           99           99           99           99
```

If data plane is high (identified by Processing Load number reaching 100%), need to see amount of traffic passing through the router (Total packet per seconds and bits per seconds) and throughput performance of the platform (you can have an idea on specific data sheet).

In order to determine if this traffic is expected or not, packet capture (EPC) or any monitoring feature such as Netflow can be used for further analysis, some checks are:

- Is traffic valid and expected to pass this router?
- Identify abnormal traffic flows or higher rates.
- If you have high packet per second numbers, look for size of the packets. Determine if this is expected to see or if you have a fragmentation issue.

If all traffic is expected, you can be reaching a platform limitation, then, look for the features running on your router as a second part for analysis via **show running-config**, mostly on the interfaces, identIfy any unnecessary features and disable them or balance traffic to release CPU cycles.

However, if there is no indication of a platform limit, another useful tool to corroborate if router is adding delay on packets is the FIA trace, you can see the exact process time spent for each packet and the features taking most of processing. The complete high CPU troubleshooting is out of scope of this document but refer to links on Related Information section.

Traffic Related

MTU and Fragmentation

Maximum Transmission Unit (MTU) is the maximum packet length to be transmitted which depends on the number of octets that physical links can convey.When upper-layer protocols submits data to the underlying IP, and the resulting length of IP packet is greater than the path MTU, the packet is divided into fragments. This lower sizes on the network causes more processing and different treatment on some cases and that is why you must avoid it as possible.

For some features such as NAT or Zone Based Firewall, virtual reassembly is required to "have the whole packet", applies what is needed, forwards its fragments, and discards the reassembled copy. This process adds CPU cycles and it is prone to errors.

Some applications does does not rely on fragmentation, one of the most basic test to check MTU is a ping with a no fragment option and test different packet sizes: **ping ip-address df-bit size number.** If ping is unsuccessful, fix MTU over the path as drop occurs and causes further issues.

Features, such as Policy-based Routing and equal cost multipath on a network with fragmented packets can create delay issues and more errors mostly on high data rates, inducing high assemble times, duplicate IDs and corrupted packets, if some of this problems are identified, please look to resolve this fragmentation as possible. One command to check if you have fragments and any potential issues is **show ip traffic**:

<#root>

```
Router#
```

```
show ip traffic
```

```
IP statistics:
  Rcvd:  9875429 total, 14340254 local destination
         0 format errors, 0 checksum errors, 0 bad hop count
         0 unknown protocol, 0 not a gateway
         0 security failures, 0 bad options, 0 with options
  Opts:  0 end, 0 nop, 0 basic security, 0 loose source route
         0 timestamp, 0 extended security, 0 record route
         0 stream ID, 0 strict source route, 0 alert, 0 cipso, 0 ump
         0 other, 0 ignored


Frags:



150 reassembled

, 0

timeouts

,

0 could not reassemble

         0

fragmented

, 600

fragments

, 0

could not fragment

         0 invalid hole
  Bcast: 31173 received, 6 sent
  Mcast: 0 received, 0 sent
  Sent:  15742903 generated, 0 forwarded
  Drop:  0 encapsulation failed, 0 unresolved, 0 no adjacency
         0 no route, 0 unicast RPF, 0 forced drop, 0 unsupported-addr
         0 options denied, 0 source IP address zero
<output omitted>
```

From the output above, bold words on Frags section refer to:

- Reassembled: Number of packets reassembled.
- Timeouts: Every time the reassemble time for a packet fragment expires.
- Could not reassemble: Number of packets that could not be reassembled.
- Fragmented: Number of packets exceeding MTU and subject for fragmenting.
- Fragments: Number of chunks into which packets were fragmented.
- Could not fragment: Number of packets exceeding MTU but could not be fragmented.

If fragmentation is used and you have timeouts or could not reassemble counters increase, one way to corroborate issues caused by the platform, is via QFP drops, using same command as explained later on drops section: **show platform hardware qfp active statistics drop.** Look for errors such as: TcpBadfrag, IpFragErr, FragTailDrop, ReassDrop, ReassFragTooBig, ReassTooManyFrags, ReassTimeout or related ones. Each case can have different causes like not getting all fragments, duplicated, CPU congestion among others. Again, useful tools for further analysis and potential fix can be a FIA trace and configuration check.

TCP offers Maximum Segment Size (MSS) mechanism to solve this problem but it can induce latency if incorrect, non MSS negotiated or wrong Path MTU discovered.

As UDP does not have this fragmentation mechanism you can rely on manual implementation of PMTD or any application-layer solution, you can enable them (when applicable) to send packets shorter than 576 bytes which is the smaller effective MTU for sending number as per RFC1122 in aids to avoid fragmentation.

Design related

More than a troubleshoot suggestion, this section briefly describes two more key components that can add to latency problems and they require an extensive discussion and analysis out of the scope of this document.

Suboptimal Routing

Suboptimal routing in networking refers to a situation where data packets are not being directed through the most efficient or shortest path available in a network. Instead, these packets are taking a route that is less efficient, possibly resulting in increased latency, congestion, or affecting network performance. IGPs choose always the best paths, which means the lower cost, but it does not necessarily is the cheapest one or the lowest delay path (best can be the one with a higher bandwidth).

Suboptimal routing can occur for issues with routing protocols; either configuration or any situation like race conditions, dynamic changes (topology changes or link failures), intended traffic engineering based on company policies or cost, redundancies or failovers (going to the backup path under certain condition) among others situations.

Tools like traceroutes or monitoring appliance can help to identify this situation for specific flows, if this is the case, and depends on many other factors, satisfy application demands and lower latency can requires routing re-design or traffic engineering.

Quality of Service (QoS)

By configuring the quality of service (QoS), you can provide preferential treatment to specific types of traffic at the expense of other traffic types. Without QoS, the device offers best-effort service for each packet, regardless of the packet contents or size. The device sends the packets without any assurance of reliability, delay bounds, or throughput.

If QoS is in place, it becomes really important to identify if router marks, re-marks or just classifies the packets, check the configuration and **show policy-map [name_of_policy_map | session | interface interface_id]** helps to understand classes affected by high rates, drops or packets wrongly classified.

Implementing QoS is a heavy duty task that requires serious analysis and is outside of the scope of this document, but it is strongly recommended to consider this to prioritize time sensitive applications and solve or prevent many latency and application issues.

Other Performance Issues

Other conditions can add slowness, session reconnect or general bad performance that you need to check, some of them are:

Drops

An issue directly related with processing on a device is packet drops, you need to check on input and output

side from interface perspective:

<#root>

```
Router#sh interfaces GigabitEthernet0/0/1
GigabitEthernet0/0/1 is up, line protocol is up
  Hardware is vNIC, address is 0ce0.995d.0000 (bia 0ce0.995d.0000)
  Internet address is 10.10.1.2/24
  MTU 1500 bytes, BW 1000000 Kbit/sec, DLY 10 usec,
     reliability 255/255, txload 1/255, rxload 1/255
  Encapsulation ARPA, loopback not set
  Keepalive set (10 sec)
  Full Duplex, 1000Mbps, link type is auto, media type is Virtual
  output flow-control is unsupported, input flow-control is unsupported
  ARP type: ARPA, ARP Timeout 04:00:00
  Last input 00:00:19, output 00:08:33, output hang never
  Last clearing of "show interface" counters never


 Input queue: 0/375/6788/0 (size/max/drops/flushes); Total output drops: 18263

  Queueing strategy: fifo
  Output queue: 0/40 (size/max)
  5 minute input rate 114000 bits/sec, 230 packets/sec
  5 minute output rate 0 bits/sec, 0 packets/sec
     193099 packets input, 11978115 bytes, 0 no buffer
     Received 0 broadcasts (0 IP multicasts)
     0 runts, 0 giants, 0 throttles


1572 input errors

,

12 CRC

, 0 frame,

1560 overrun

, 0 ignored
     0 watchdog, 0 multicast, 0 pause input
     142 packets output, 11822 bytes, 0 underruns
     Output 0 broadcasts (0 IP multicasts)
     0 output errors, 0 collisions, 0 interface resets
     23 unknown protocol drops
     0 babbles, 0 late collision, 0 deferred
     0 lost carrier, 0 no carrier, 0 pause output
     0 output buffer failures, 0 output buffers swapped out
Router#
```

On input side you have:

- Input queue drops: Each interface owns an input queue (this is a software buffer that can be modified) which incoming packets are placed to await processing by the Routing Processor (RP). if rate for incoming packets placed on the input queue exceeds the rate at which the RP can process the packets you can have drops increment. However, be aware that only control packets and "For us" traffic is placed, therefore, if latency is seen on passing through traffic, even if you have sporadic drops, this must not be a cause.

- Overruns: This occurs when receiver hardware is unable to hand received packets to a hardware buffer because the input rate exceeds the ability of the receiver to handle the data. This number can indicate a problem with the rate and performance of router, capture traffic only for this interface and look for traffic spikes. A common workaround is to enable flow control but this can add to delay packets. This can be also an evidence for bottlenecks and oversubscription.
- CRCs: Occurs because of physical issues, check cabling, ports and SFPs correctly connected and proper function.

On output side you have:

- Output queue drops: Each interface owns an output queue where is outgoing packets to be sent on the interface are placed. Sometimes the rate for outgoing packets placed on the output queue by the RP exceeds the rate at which the interface can send the packets, This can cause performance issue and latency problems if there is no QoS in place, otherwise, you can have this number increasing because of certain policy applied and advise to check or implement QoS configuration to protect and assure intended or critical traffic.

Finally, drops on QFP is directly related to high processing that can cause latency, check via **show platform hardware qfp active statistics drop**:

<#root>

```
Router#

show platform hardware qfp active statistics drop

Last clearing of QFP drops statistics : never

-------------------------------------------------------------------------
Global Drop Stats                          Packets                 Octets
-------------------------------------------------------------------------
Disabled                                         2                    646
Ipv4NoAdj                                   108171                6706602
Ipv6NoRoute                                     10                    560
```

Causes depend on code, FIA trace helps to corroborate or discard if the traffic affected by latency is dropped at this point.

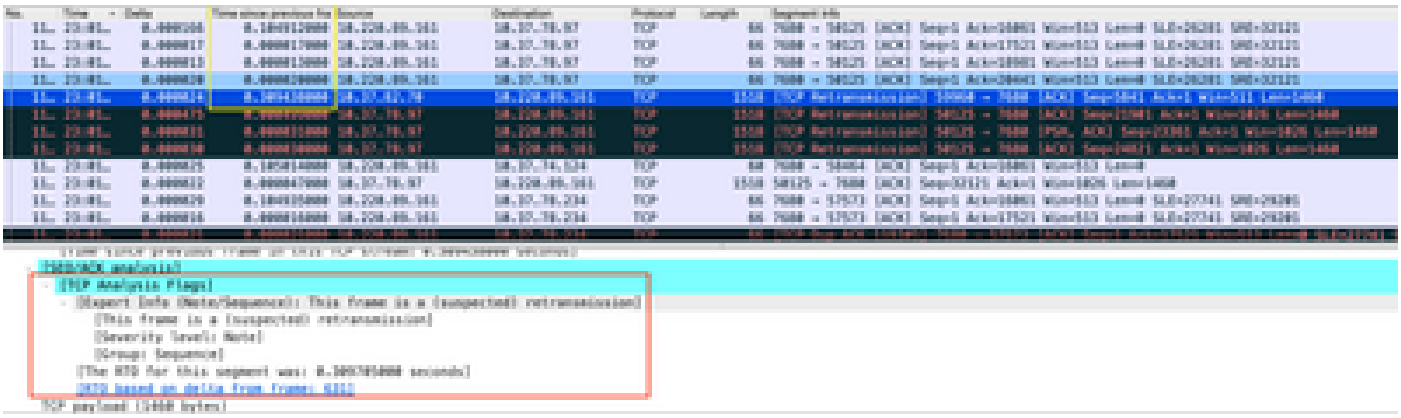
TCP Retransmission

TCP retransmission is a symptom or can be a consequence due to an underlay problem such as packet loss. This problem can induce to slowness and bad performance on application.

The Transmission Control Protocol (TCP) uses a retransmission timer to ensure data delivery in the absence of any feedback from the remote data receiver. The duration of this timer is referred to as RTO (retransmission timeout). When the retransmission timer expires, sender retransmits the earliest segment that has not been acknowledged by the TCP receiver and RTO is increased.

Some retransmissions can not be eliminated completely, if they are minimal, it can not reflect a problem. However, as you can infer, more retransmission seen, more latency on the TCP session and needs to be addressed.

Packet capture analyzed in Wireshark can corroborate the problem as next example:



TCP Conversation Capture

If there are retransmissions, use the same capture method on the router ingress and egress direction to check all packets send and received. Of course, doing this on every hop can represent a tremendous effort so detailed analysis on capture is needed for TCP, looking at TTLs, times from previous frames on same TCP stream to understand from which direction (server or client) you have this delay or lack of response to direct your troubleshooting.

Oversubscription and Bottlenecks

Oversubscription happens when the required resources (bandwidth) are greater than the actual available ones. Commands to identify if you have this problem on a router have been already covered on previous section.

Consequence of this situation, bottlenecks can happen when traffic flows are slowed down because of insufficient bandwidth or hardware capacity. It is important to identify if this happens in short period of time or it is long term situation to apply solutions.

There is no specific advise to solve it but some of the options are balance traffic to different platform, segment the network or upgrade to more robust devices based on current needs and future growth analysis.

Related Information

- **IP SLAs ICMP Echo Operations**
- **Memory Troubleshooting**
- **Troubleshoot with the Cisco IOS-XE Datapath Packet Trace Feature**
- **Troubleshoot Packet Drops on ASR 1000 Series Service Routers.**
- **Qos Related information**
- **QoS Configuration on routers**
- **Cisco Technical Support & Downloads**