

Troubleshooting Fundamentals – Phase 1.8

Objective

Host A is unable to communicate with Host D. Investigate and resolve the issue, if possible.

Topology

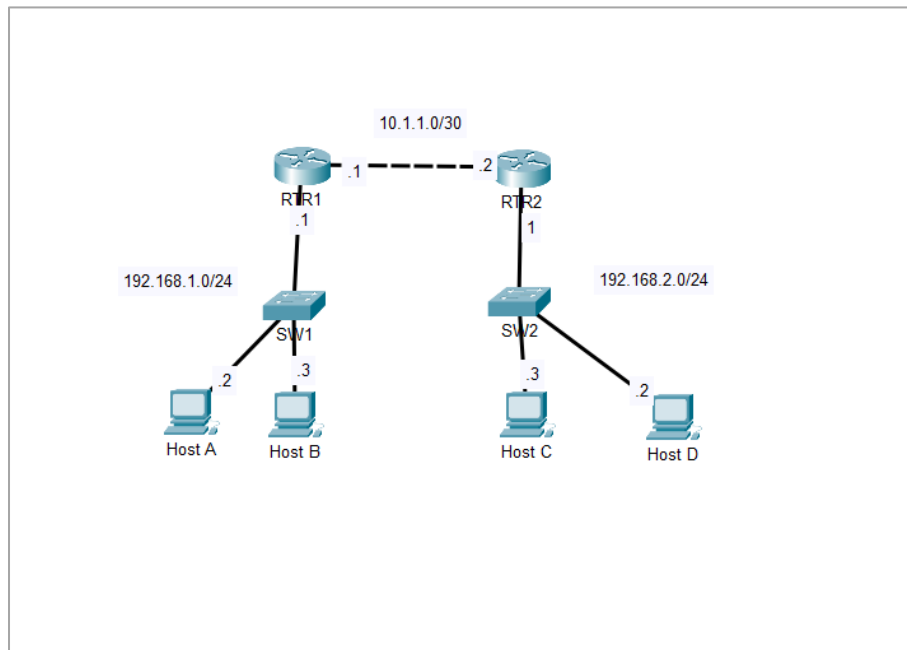


Figure 1 – Topology

Step 1 – Verify the Problem

From **Host A**, ping **Host D**.

Result: Ping fails.

```
Cisco Packet Tracer PC Command Line 1.0
C:\>ping 192.168.2.2

Pinging 192.168.2.2 with 32 bytes of data:

Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.168.2.2:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),

C:\>|
```

Figure 2 – Unsuccessful Ping from Host A to Host D

The issue is confirmed. Be mindful that a ping may fail due to administrative requirements in the form of ACLs.

Step 2 – Decide Where to Start

A useful way to approach troubleshooting is to ask: *what must be true for communication to work?* Each requirement becomes something that can be verified.

What needs to happen for Host A to have reachability to Host D?

- You could list out everything for Layers 1 and 2 but going straight to Layer 3 is a more logical and efficient starting point. Hosts A and D reside on different networks which means a Layer 3 issue is just as likely as a Layer 1 or 2 issue. Successfully validating Layer 3 also validates Layers 1 and 2. Should there be an issue validating Layer 3, the next step would be to check Layers 1 and 2, which directly impact Layer 3.

What needs to happen at Layer 3 for Host A to have reachability to Host D?

- Hosts A and D require correct IP assignments.
- Hosts A and D require gateways.
- The gateways require knowledge of the destination network in the form of a route.

Step 3 – Troubleshoot

- a. From Host A, ping the gateway. It would be reasonable to validate that Host A has an IP and gateway configured, but a successful ping will validate those components.

Observation:

- Host A has reachability to the gateway.

```
C:\>ping 192.168.1.1

Pinging 192.168.1.1 with 32 bytes of data:

Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255
Reply from 192.168.1.1: bytes=32 time<1ms TTL=255

Ping statistics for 192.168.1.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
    Approximate round trip times in milli-seconds:
        Minimum = 0ms, Maximum = 1ms, Average = 0ms

C:\>|
```

Figure 3 – Host A gateway ping output

- b. Log into the gateway, RTR1, and ping Host D's gateway. Host D may be the issue. Attempting to validate reachability to its gateway will help determine whether the entire destination network is unreachable or if the issue is isolated to Host D.

Observation: RTR1 does not have reachability to Host D's gateway. It is safe to assume the issue may be reachability to the entire destination network.

```
RTR1>
RTR1>en
RTR1#ping 192.168.2.1

Type escape sequence to abort.
Sending 5, 100-byte ICMP Echos to 192.168.2.1, timeout is 2 seconds:
.....
Success rate is 0 percent (0/5)

RTR1#|
```

Figure 4 – RTR1 ping output

- c. Issue a **tracert** to the destination. A tracert is not always necessary, but it can reveal where the issue may be occurring. Especially, if there are multiple hops to the destination network.

Observation: The tracert fails. A tracert could easily fail in a production network due to ICMP being blocked. That is not the case in this situation. This output suggests RTR1 may not have knowledge of the destination network, or the routing information may be incorrect.

The reasoning for that comes from the way a traceroute works. If RTR1 had knowledge of the destination network, it would have forwarded the traceroute probe toward RTR2. RTR2 would have responded with a TTL expired message. This is a reasonable expectation because RTR1 and RTR2 are directly connected. RTR2 is the next hop.

```
RTR1#traceroute 192.168.2.1
Type escape sequence to abort.
Tracing the route to 192.168.2.1

 1  *      *      *
 2  *      *
```

Figure 5 – RTR1 traceroute output

- d. Validate that RTR1 has a route to the destination.

Observation: RTR1 has a route to 192.168.2.0/24. The destination network is present in the routing table.

```
10.0.0.0/8 is variably subnetted, 2 subnets, 2 masks
C    10.1.1.0/29 is directly connected, GigabitEthernet0/0/0
L    10.1.1.1/32 is directly connected, GigabitEthernet0/0/0
    192.168.1.0/24 is variably subnetted, 2 subnets, 2 masks
C    192.168.1.0/24 is directly connected, GigabitEthernet0/0/1
L    192.168.1.1/32 is directly connected, GigabitEthernet0/0/1
S    192.168.2.0/24 [1/0] via 10.1.1.4
RTR1#
```

Figure 6 – RTR1 show ip route output

- e. Validate reachability to the next hop with a ping.

Observation: The ping fails. If you were paying attention to the diagram, you may have already known that the next hop was not 10.1.1.4. In this case, the ping would not have been necessary. The correct next hop is 10.1.1.2. This misconfiguration would prevent Host A from reaching Host D.

```
RTR1(config)#do sh run | sec ip route
ip route 192.168.2.0 255.255.255.0 10.1.1.4
RTR1(config)#
```

Figure 7 – RTR1 show run ip route output

The issue is that Hosts A's gateway has an incorrect next hop of 10.1.1.4 for the static route for the destination network. RTR2's IP on the shared network is 10.1.1.2 and the static route should have been pointing at that IP, instead. You should not expect this type of misconfiguration to occur frequently in a stable network that experiences few changes.

Step 4 – Correct and Validate

- a. On RTR1, remove the incorrect static route and configure the correct one.

```
no ip route 192.168.2.0 255.255.255.0 10.1.1.4
ip route 192.168.2.0 255.255.255.0 10.1.1.2
```

- b. On RTR1, observe the output of the traceroute. As previously predicted, RTR2 responded to the traceroute with its IP on the shared network. RTR2 also reveals that it is directly connected to the destination network. Had RTR2 relied on another hop to reach the destination, the expectation would have been to receive a response from that device as well; however, the traceroute ended with RTR2.

c.

```
RTR1# traceroute 192.168.2.1
Type escape sequence to abort.
Tracing the route to 192.168.2.1

 1  10.1.1.2          0 msec    0 msec    0 msec
RTR1#
```

Figure 8 – RTR1 traceroute output

- d. Re-test ping from Host A to Host D.

Result: Success.