Troubleshoot OSPF Neighbors Stuck in Exstart/Exchange State

Contents

Introduction

Prerequisites

- Requirements
- Components Used
- Conventions

Background Information

Exstart State

Exchange State

Neighbors Stuck in Exstart/Exchange State

Solution

Related Information

Introduction

This document describes how to troubleshoot situations in which Open Shortest Path First (OSPF) neighbors are stuck in Exstart and Exchange states.

Prerequisites

Requirements

It is recommended that the user is familiar with basic OSPF operation and configuration, in particular, OSPF neighbor states.

Components Used

The information in this document is based on these software and hardware versions:

- Cisco 2503 routers
- Cisco IOS®Software Release 12.2(24a) to run on both routers

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.

Conventions

For more information on document conventions, refer to the Cisco Technical Tips Conventions.

Background Information

OSPF states for adjacency formation there are Down, Init, 2-way, Exstart, Exchange, Loading and Full. There can be a number of reasons why the Open Shortest Path First (OSPF) neighbors are stuck in Exstart/Exchange state. This document focuses on an MTU mismatch between OSPF neighbors that result in Exstart/Exchange state. For more details on how to troubleshoot OSPF refer to Troubleshoot Common Problems with OSPF.

Exstart State

After two OSPF neighboring routers establish bi-directional communication and complete DR/BDR election (on multi-access networks), the routers transition to the Exstart state. In this state, the neighboring routers establish a Primary/Subordinate relationship and determine the initial database descriptor (DBD) sequence number to use while the DBD packets are exchanged.

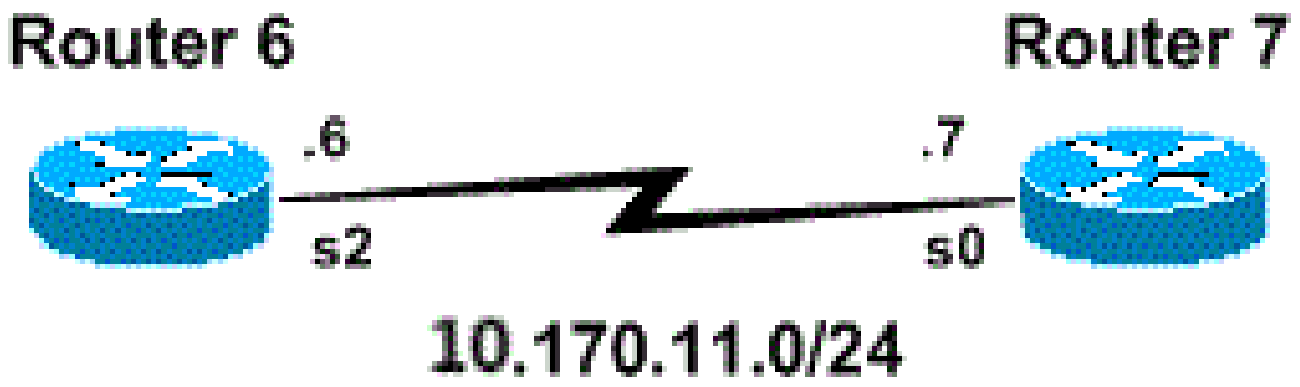
Exchange State

Once the Primary/Subordinate relationship has been negotiated (the router with the highest Router-ID becomes the Primary), the neighbor routers transition into the Exchange state. In this state, the routers exchange DBD packets, which describe their entire link-state database. The routers also send link-state request packets, which request more recent link-state advertisements (LSA) from neighbors.

Although OSPF neighbors transition through the Exstart/Exchange states during the normal OSPF adjacency-building process, it is not normal for OSPF neighbors to be stuck in this state. The next section describes the most common reason that OSPF neighbors get stuck in this state. Refer to Understand OSPF Neighbor States to learn more about the different OSPF states.

Neighbors Stuck in Exstart/Exchange State

The problem occurs most frequently when you attempt to run OSPF between a Cisco router and another vendor router. The problem occurs when the maximum transmission unit (MTU) settings for neighboring router interfaces do not match. If the router with the higher MTU sends a packet larger that the MTU set on the neighboring router, the neighbor router ignores the packet. When this problem occurs, the output of the show ip ospf neighbor command displays output similar to what is shown in this figure.



This section describes an actual recreation of this problem.

Router 6 and Router 7 in this figure are connected via Frame Relay and Router 6 has been configured with 5 static routes redistributed into OSPF. The serial interface on Router 6 has the default MTU of 1500, while the serial interface on Router 7 has an MTU of 1450. Each router configuration is shown in the table (only necessary configuration information is shown):

Router 6 Configuration	Router 7 Configuration
interface Serial2 !--- MTU is set to its default value of 1500. no ip address no ip directed-broadcast encapsulation frame-relay no ip mroute-cache frame-relay lmi-type ansi ! interface Serial2.7 point-to-point ip address 10.170.10.6 255.255.255.0 no ip directed-broadcast frame-relay interface-dlci 101 ! router ospf 7 redistribute static subnets network 10.170.10.0 0.0.0.255 area 0 ! ip route 192.168.0.10 255.255.255.0 Null0 ip route 192.168.10.10 255.255.255.0 Null0 ip route 192.168.10.0 255.255.255.0 Null0 ip route 192.168.37.10 255.255.255.0 Null0 ip route 192.168.38.10 255.255.255.0 Null0	interface Serial0 mtu 1450 no ip address no ip directed-broadcast encapsulation frame-relay frame-relay lmi-type ANSI ! interface Serial0.6 point-to-point ip address 172.16.7.11 255.255.255.0 no ip directed-broadcast frame-relay interface-dlci 110 ! router ospf 7 network 172.16.11.6 0.0.0.255 area 0

The **show ip ospf neighbor** command output for each router is:

```
<#root>

router-6#

show ip ospf neighbor


Neighbor ID     Pri   State           Dead Time   Address         Interface
172.16.7.11       1   EXCHANGE/  -    00:00:36    172.16.7.11    Serial2.7
router-6#

router-7#

show ip ospf neighbor


Neighbor ID     Pri   State           Dead Time   Address         Interface
10.170.10.6       1   EXSTART/  -     00:00:33    10.170.10.6    Serial0.6
router-7#
```

The problem occurs when Router 6 sends a DBD packet larger than 1450 bytes (Router 7's MTU) while in the exchange state. Use the debug ip packet and debug ip ospf adj commands on each router to see the OSPF adjacency process as it takes place. The output from Router 6 and 7 from steps 1 to 14 is:

1. Router 6 debug output:

```
<<<ROUTER 6 IS SENDING HELLOS BUT HEARS NOTHING, STATE OF NEIGHBOR IS DOWN>>>

00:03:53: OSPF: 172.16.7.11 address 172.16.7.11 on
Serial2.7 is dead
00:03:53: OSPF: 172.16.7.11 address 172.16.7.11 on
Serial2.7 is dead, state DOWN
```

2. Router 7 debug output:

```
<<<OSPF NOT ENABLED ON ROUTER7 YET>>>
```

3. Router 6 debug output:

```
<<<ROUTER 6 SENDING HELLOS>>>

00:03:53: IP: s=10.170.10.6 (local), d=224.0.0.5
(Serial2.7), len 64, sending broad/multicast, proto=89
00:04:03: IP: s=10.170.10.6 (local), d=224.0.0.5
(Serial2.7), Len 64, sending broad/multicast, proto=89
```

4. Router 7 debug output:

```
<<<OSPF NOT ENABLED ON ROUTER7 YET>>>
```

5. Router 7 debug output:

```
<<<OSPF ENABLED ON ROUTER 7, BEGINS SENDING HELLOS AND BUILDING A ROUTER LSA>>>

00:17:44: IP: s=172.16.7.11 (local), d=224.0.0.5
(Serial0.6), Len 64, sending broad/multicast, proto=89
00:17:44: OSPF: Build router LSA for area 0,
router ID 172.16.7.11, seq 0x80000001
```

6. Router 6 debug output:

```
<<<RECEIVE HELLO FROM ROUTER7>>>

00:04:04: IP: s=172.16.7.11 (Serial2.7), d=224.0.0.5,
Len 64, rcvd 0, proto=89
00:04:04: OSPF: Rcv hello from 172.16.7.11 area 0 from
Serial2.7 172.16.7.11
00:04:04: OSPF: End of hello processing
```

7. Router 6 debug output:

```
<<<ROUTER 6 SEND HELLO WITH ROUTER7 ROUTERID IN THE HELLO PACKET>>>

00:04:13: IP: s=10.170.10.6 (local), d=224.0.0.5
(Serial2.7), Len 68, sending broad/multicast, proto=89
```

8. Router 7 debug output:

```
<<<ROUTER 7 RECEIVES HELLO FROM ROUTER6 CHANGES STATE TO 2WAY>>>

00:17:53: IP: s=10.170.10.6 (Serial0.6), d=224.0.0.5,
Len 68, rcvd 0, proto=89
00:17:53: OSPF: Rcv hello from 10.170.10.6 area 0 from
Serial0.6 10.170.10.6
00:17:53: OSPF: 2 Way Communication to 10.170.10.6 on
Serial0.6, state 2WAY
```

9. Router 7 debug output:

```
<<<ROUTER 7 SENDS INITIAL DBD PACKET WITH SEQ# 0x13FD>>>

00:17:53: OSPF: Send DBD to 10.170.10.6 on Serial0.6
seq 0x13FD opt 0x2 flag 0x7 Len 32
00:17:53: IP: s=172.16.7.11 (local), d=224.0.0.5
(Serial0.6), Len 52, sending broad/multicast, proto=89
00:17:53: OSPF: End of hello processing
```

10. Router 6 debug output:

```
<<<ROUTER 6 RECEIVES ROUTER7'S INITIAL DBD PACKET CHANGES STATE TO 2-WAY>>>

00:04:13: IP: s=172.16.7.11 (Serial2.7), d=224.0.0.5,
Len 52, rcvd 0, proto=89
00:04:13: OSPF: Rcv DBD from 172.16.7.11 on Serial2.7
seq 0x13FD opt 0x2 flag 0x7 Len 32 mtu 1450 state INIT
```

```
00:04:13: OSPF: 2 Way Communication to 172.16.7.11 on
Serial2.7, state 2WAY
```

11. Router 6 debug output:

```
<<<ROUTER 6 SENDS DBD PACKET TO ROUTER 7 (PRIMARY/SUBORDINATE NEGOTIATION - ROUTER 6 IS SUB

00:04:13: OSPF: Send DBD to 172.16.7.11 on Serial2.7
seq 0xE44 opt 0x2 flag 0x7 Len 32
00:04:13: IP: s=10.170.10.6 (local), d=224.0.0.5 (Serial2.7),
Len 52, sending broad/multicast, proto=89
00:04:13: OSPF: NBR Negotiation Done. We are the SLAVE
```

12. Router 7 debug output:

```
<<<RECEIVE ROUTER 6'S INITIAL DBD PACKET (MTU MISMATCH IS RECOGNIZED)>>>

00:17:53: IP: s=10.170.10.6 (Serial0.6), d=224.0.0.5,
Len 52, rcvd 0, proto=89
00:17:53: OSPF: Rcv DBD from 10.170.10.6 on Serial0.6
seq 0xE44 opt 0x2 flag 0x7 Len 32 mtu 1500 state EXSTART
00:17:53: OSPF: Nbr 10.170.10.6 has larger interface MTU
```

13. Router 6 debug output:

```
<<<SINCE ROUTER 6 IS SUBORDINATE SEND DBD PACKET WITH LSA HEADERS, SAME SEQ# (0x13FD) TO ACK

00:04:13: OSPF: Send DBD to 172.16.7.11 on Serial2.7
seq 0x13FD opt 0x2 flag 0x2 Len 1472
00:04:13: IP: s=10.170.10.6 (local), d=224.0.0.5 (Serial2.7),
Len 1492, sending broad/multicast, proto=89
```

14. Router 7 debug output:

```
<<<NEVER RECEIVE ACK TO ROUTER7'S INITIAL DBD, RETRANSMIT>>>

00:17:54: IP: s=172.16.7.11 (local), d=224.0.0.5
(Serial0.6), Len 68, sending broad/multicast, proto=89
00:18:03: OSPF: Send DBD to 10.170.10.6 on Serial0.6
seq 0x13FD opt 0x2 flag 0x7 Len 32 00:18:03: OSPF:
Retransmitting DBD to 10.170.10.6 on Serial0.6 [1]
```

At this point, Router 6 continues to try to ACK the initial DBD packet from Router 7.

```
00:04:13: IP: s=172.16.7.11 (Serial2.7), d=224.0.0.5,
Len 68, rcvd 0, proto=89
00:04:13: OSPF: Rcv hello from 172.16.7.11 area 0 from
Serial2.7 172.16.7.11
00:04:13: OSPF: End of hello processing

00:04:18: IP: s=172.16.7.11 (Serial2.7), d=224.0.0.5,
Len 52, rcvd 0, proto=89
00:04:18: OSPF: Rcv DBD from 172.16.7.11 on Serial2.7
seq 0x13FD opt 0x2 flag 0x7 Len 32 mtu 1450 state EXCHANGE

00:04:18: OSPF: Send DBD to 172.16.7.11 on Serial2.7
seq 0x13FD opt 0x2 flag 0x2 Len 1472
00:04:18: IP: s=10.170.10.6 (local), d=224.0.0.5
(Serial2.7), Len 1492, sending broad/multicast, proto=89

00:04:23: IP: s=10.170.10.6 (local), d=224.0.0.5
(Serial2.7), Len 68, sending broad/multicast, proto=89

00:04:23: IP: s=172.16.7.11 (Serial2.7), d=224.0.0.5,
Len 52, rcvd 0, proto=89
00:04:23: OSPF: Rcv DBD from 172.16.7.11 on Serial2.7
seq 0x13FD opt 0x2 flag 0x7 Len 32 mtu 1450 state EXCHANGE
```

Router 7 never gets an ACK from Router 6 because the DBD packet from Router 7 is too large for the Router 7 MTU. Router 7 repeatedly retransmits the DBD packet.

```
0:17:58: IP: s=172.16.7.11 (local), d=224.0.0.5
(Serial0.6), Len 52, sending broad/multicast, proto=89
00:18:03: OSPF: Send DBD to 10.170.10.6 on Serial0.6
seq 0x13FD opt 0x2 flag 0x7 Len 32 00:18:03: OSPF:
Retransmitting DBD to 10.170.10.6 on Serial0.6 [2]

00:18:03: IP: s=172.16.7.11 (local), d=224.0.0.5
(Serial0.6), Len 52, sending broad/multicast, proto=89
00:18:03: IP: s=10.170.10.6 (Serial0.6), d=224.0.0.5,
Len 68, rcvd 0, proto=89
00:18:03: OSPF: Rcv hello from 10.170.10.6 area 0 from
Serial0.6 10.170.10.6
00:18:03: OSPF: End of hello processing

00:18:04: IP: s=172.16.7.11 (local), d=224.0.0.5
(Serial0.6), Len 68, sending broad/multicast, proto=89

00:18:03: OSPF: Send DBD to 10.170.10.6 on Serial0.6
seq 0x13FD opt 0x2 flag 0x7 Len 32 00:18:03: OSPF:
Retransmitting DBD to 10.170.10.6 on Serial0.6 [3]

00:18:08: IP: s=172.16.7.11 (local), d=224.0.0.5
(Serial0.6), Len 52, sending broad/multicast, proto=89
router-7#
```

Because Router 6 has a higher MTU, it continues to accept the DBD packet from Router 7, attempts to acknowledge them, and remains in the EXCHANGE state.

Because Router 7 has a lower MTU, it ignores the DBD packets along with ACK from Router 6, continues to retransmit the initial DBD packet, and remains in the EXSTART state.

In step 9 and 11, Router 7 and Router 6 send their first DBD packets with flag 0x7 set as part of Primary/Subordinate negotiation. After Primary/Subordinate determination, Router 7 is elected as Primary because of its higher Router-ID. The flags in steps 13 and 14 clearly shows that Router 7 is Primary (Flag 0x7) and Router 6 is Subordinate (Flag 0x2).

In step 10, Router 6 receives the Router 7 initial DBD packet and transitions its state to 2-way.

In step 12, Router 7 receives the Router 6 initial DBD packet and recognizes an MTU mismatch. (Router 7 is able to recognize a MTU mismatch because Router 6 includes its interface MTU in the interface MTU field of the DBD packet). The Router 6 initial DBD is rejected by Router 7. Router 7 retransmits the initial DBD packet.

Step 13 shows that Router 6, as subordinate, adopts the Router 7 sequence number and sends its second DBD packet that contains the headers of its LSAs, which increases the size of the packet. However, Router 7 never receives this DBD packet because it is larger than the Router 7 MTU.

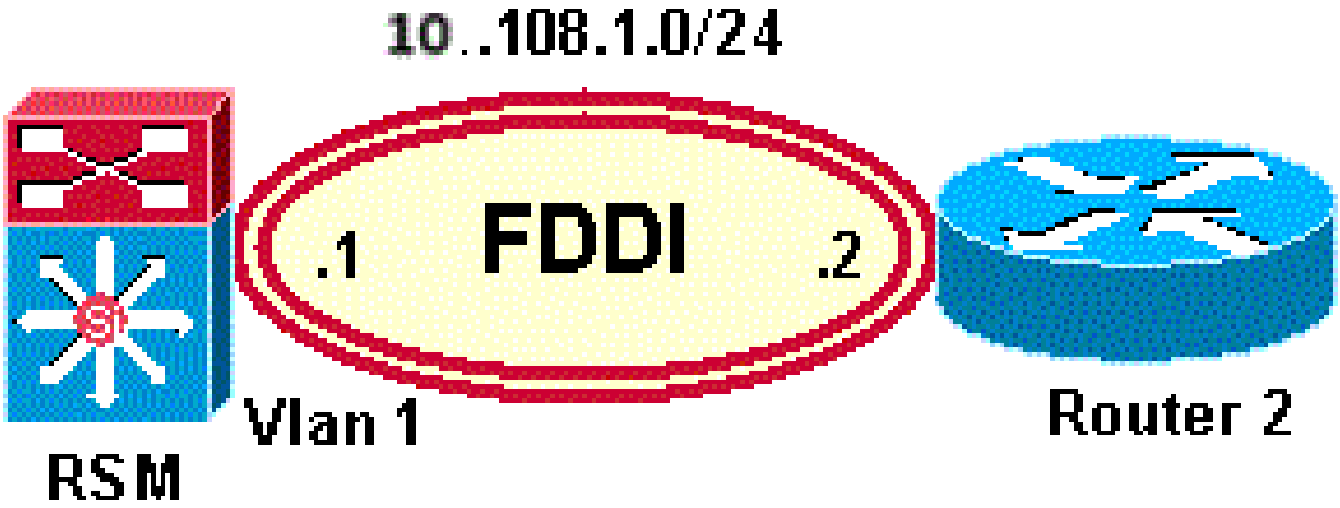
After step 13, Router 7 continues to retransmit the initial DBD packet to Router 6, while Router 6 continues to send DBD packets that adhere to the Primary sequence number. This loop continues indefinitely, which prevents either router from transition out of the exstart/exchange state.

Solution

Since the problem is caused by mismatched MTUs, the solution is to change either router MTU to match the neighbor MTU.

Note: Cisco IOS Software Release 12.0(3) introduced interface MTU mismatch detection. This detection involves the OSPF that advertises the interface MTU in the DBD packets, which is in accordance with the OSPF RFC 2178, appendix G.9. When a router receives a DBD packet that is advertised, a MTU larger than the router can receive, the router ignores the DBD packet and the neighbor state remains in Exstart. This prevents the formation of an adjacency. To fix this problem, ensure that the MTU are the same on both ends of a link.

In Cisco IOS Software 12.01(3), the **ip ospf mtu-ignore** interface configuration command was also introduced to turn off the MTU mismatch detection; however, this is only needed in rare instances, as shown in this diagram:



The previous diagram shows a Fiber Distributed Data Interface (FDDI) port on a Cisco Catalyst 5000 with a Route Switch Module (RSM) connected to a FDDI interface on Router 2. The Virtual LAN (VLAN) on the RSM is a virtual Ethernet interface with a MTU of 1500, and the FDDI interface on Router 2 has a MTU of 4500. When a packet is received on the FDDI port of the switch, it goes to the backplane and the FDDI to Ethernet conversion/fragmentation happens within the switch itself. This is a valid setup, but with MTU mismatch detection feature, OSPF adjacency is not formed in between the router and the RSM. Since FDDI and Ethernet MTU are different, this **ip ospf mtu-ignore** command is useful on the VLAN interface of the RSM to stop OSPF detection of MTU mismatch and forms the adjacency.

It is important to notice that MTU mismatch, although the most common, is not the only reason that OSPF neighbors get stuck in the Exstart/Exchange state. The problem is most frequently caused by the inability to successfully exchange DBD packets. However, the root cause could be any of the these:

- MTU mismatch
- Unicast is broken. In the Exstart state, the router sends a unicast packet to the neighbor to elect Primary and Subordinate. This is true unless you have a point-to-point link, in which case it sends a multicast packet. These are the possible causes:
    - Wrong virtual circuit (VC) mapping in an Asynchronous Transfer Mode (ATM) or Frame Relay environment in highly redundant network.
    - MTU problem, which means the routers can only ping a packet of a certain length.
    - Access list blocks the unicast packet.
    - NAT runs on the router and translates the unicast packet.
- Neighbor between PRI and BRI/dialer.
- Both routers have the same Router-ID (mis-configuration).

In addition, the OSPF RFC 2328, section 10.3, states that the Exstart/Exchange process is initiated for any of these events (any of which could be caused by internal software problems):

- Sequence Number Mismatch.
    - Unexpected DD Sequence number.
    - "I" bit is set unexpectedly.
    - Option field different from the last option field received in the DBD packet.
- BadLSReq
    - Neighbor sends unrecognized LSA during exchange process.
    - Neighbor requested an LSA during exchange process that cannot be found.

When OSPF does not form neighbors, consider the factors mentioned previously, such as the physical media and network hardware, in order to troubleshoot the problem.

Related Information

- Understand OSPF Neighbor States
- Troubleshoot OSPF Neighbor Problems
- Cisco Technical Support & Downloads