Computer Networks: Open Shortest Path First (OSPF)

Bodhisatwa Mazumdar[1]

[1]Discipline of Computer Science & Engineering
Indian Institute of Technology Indore

April 10, 2019

Open Shortest Path First (OSPF)

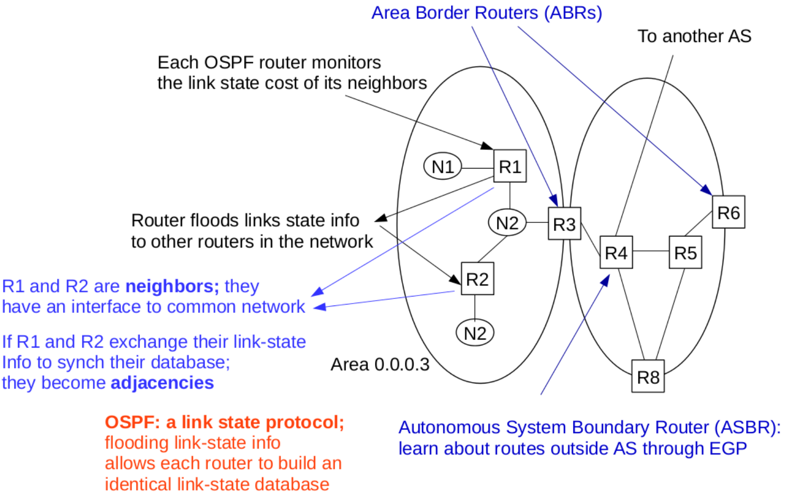

Figure : OSPF protocol in routers and networks; an IGP protocol that enables each router to learn entire network

OSPF Operation

Stages of OSPF operation

- Neighbors discovered through transmission of `Hello` messages.
- Designated routers are elected in multiaccess networks.
- Adjacencies are established; link-state databases are synchronized.
- Link state advertisements (LSAs) are exchanged by adjacent routers; they advertise inter-area and inter-AS routes.
- Routers use database info to generate routing tables.

OSPF Operation

Stages of OSPF operation

- Neighbors discovered through transmission of `Hello` messages.
- Designated routers are elected in multiaccess networks.
- Adjacencies are established; link-state databases are synchronized.
- Link state advertisements (LSAs) are exchanged by adjacent routers; they advertise inter-area and inter-AS routes.
- Routers use database info to generate routing tables.

OSPF Operation

Stages of OSPF operation

- Neighbors discovered through transmission of `Hello` messages.
- Designated routers are elected in multiaccess networks.
- Adjacencies are established; link-state databases are synchronized.
- Link state advertisements (LSAs) are exchanged by adjacent routers; they advertise inter-area and inter-AS routes.
- Routers use database info to generate routing tables.

OSPF Operation

Stages of OSPF operation

- Neighbors discovered through transmission of `Hello` messages.
- Designated routers are elected in multiaccess networks.
- Adjacencies are established; link-state databases are synchronized.
- Link state advertisements (LSAs) are exchanged by adjacent routers; they advertise inter-area and inter-AS routes.
- Routers use database info to generate routing tables.

OSPF Operation

Stages of OSPF operation

- Neighbors discovered through transmission of `Hello` messages.
- Designated routers are elected in multiaccess networks.
- Adjacencies are established; link-state databases are synchronized.
- Link state advertisements (LSAs) are exchanged by adjacent routers; they advertise inter-area and inter-AS routes.
- Routers use database info to generate routing tables.

OSPF Operation

Stages of OSPF operation

- Neighbors discovered through transmission of `Hello` messages.
- Designated routers are elected in multiaccess networks.
- Adjacencies are established; link-state databases are synchronized.
- Link state advertisements (LSAs) are exchanged by adjacent routers; they advertise inter-area and inter-AS routes.
- Routers use database info to generate routing tables.

OSPF Operation

Stages of OSPF operation

- Neighbors discovered through transmission of `Hello` messages.
- Designated routers are elected in multiaccess networks.
- Adjacencies are established; link-state databases are synchronized.
- Link state advertisements (LSAs) are exchanged by adjacent routers; they advertise inter-area and inter-AS routes.
- Routers use database info to generate routing tables.

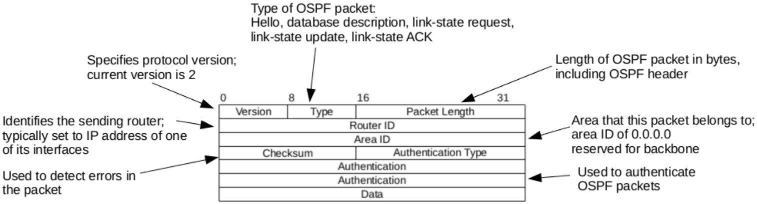

Figure : OSPF common header precedes each OSPF packet

Type 1: OSPF Hello Packets

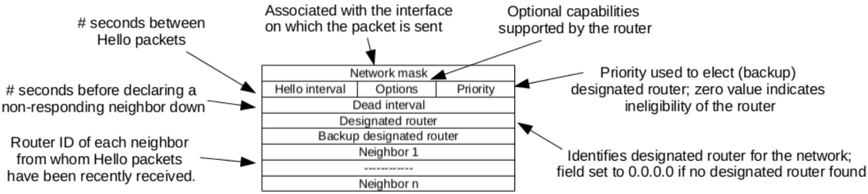

Figure : Hello packet fields

Type 1: OSPF Hello Packets

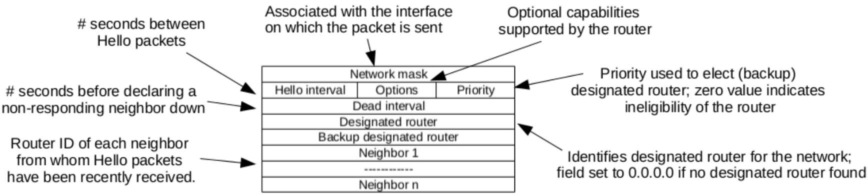

Figure : Hello packet fields

- Hello packets are sent to its neighbors periodically to discover, establish, and maintain neighbor relationships.
- Each router $R1$ broadcasts Hello packets periodically onto its network.
- When a router $R2$ receives a Hello packet, it replies with a Hello packet with router ID of each neighbor it has seen.
- If $R1$ observes that Hello packet sent by $R2$ contains its router ID field in the packet, it is assured that communication is bidirectional.
- After neighbor discovery, designated routers are elected in each multiaccess network.
- Election is based on highest value of priority and ID fields.

Type 1: OSPF Hello Packets

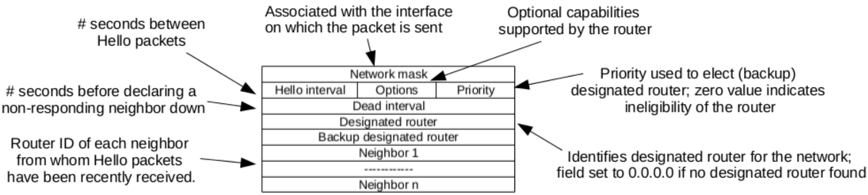

Figure : Hello packet fields

- Hello packets are sent to its neighbors periodically to discover, establish, and maintain neighbor relationships.
- Each router $R1$ broadcasts Hello packets periodically onto its network.
- When a router $R2$ receives a Hello packet, it replies with a Hello packet with router ID of each neighbor it has seen.
- If $R1$ observes that Hello packet sent by $R2$ contains its router ID field in the packet, it is assured that communication is bidirectional.
- After neighbor discovery, designated routers are elected in each multiaccess network.
- Election is based on highest value of priority and ID fields.

Type 1: OSPF Hello Packets

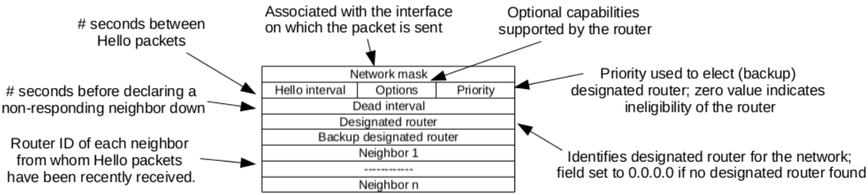

Figure : Hello packet fields

- Hello packets are sent to its neighbors periodically to discover, establish, and maintain neighbor relationships.
- Each router $R1$ broadcasts Hello packets periodically onto its network.
- When a router $R2$ receives a Hello packet, it replies with a Hello packet with router ID of each neighbor it has seen.
- If $R1$ observes that Hello packet sent by $R2$ contains its router ID field in the packet, it is assured that communication is bidirectional.
- After neighbor discovery, designated routers are elected in each multiaccess network.
- Election is based on highest value of priority and ID fields.

Type 1: OSPF Hello Packets

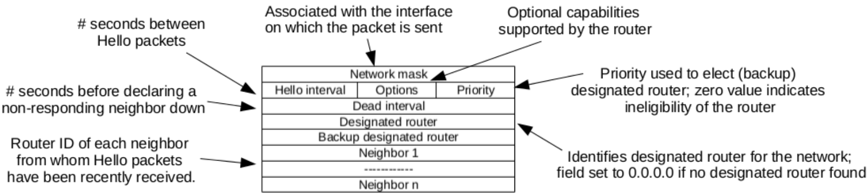

Figure : Hello packet fields

- Hello packets are sent to its neighbors periodically to discover, establish, and maintain neighbor relationships.
- Each router $R1$ broadcasts Hello packets periodically onto its network.
- When a router $R2$ receives a Hello packet, it replies with a Hello packet with router ID of each neighbor it has seen.
- If $R1$ observes that Hello packet sent by $R2$ contains its router ID field in the packet, it is assured that communication is bidirectional.
- After neighbor discovery, designated routers are elected in each multiaccess network.
- Election is based on highest value of priority and ID fields.

Type 1: OSPF Hello Packets

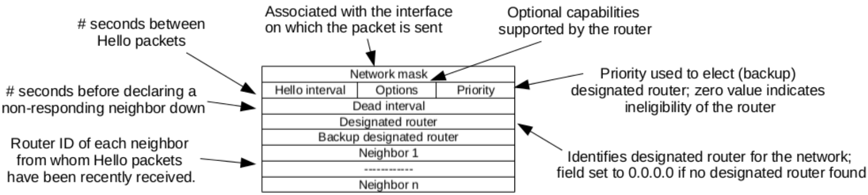

Figure : Hello packet fields

- Hello packets are sent to its neighbors periodically to discover, establish, and maintain neighbor relationships.
- Each router $R1$ broadcasts Hello packets periodically onto its network.
- When a router $R2$ receives a Hello packet, it replies with a Hello packet with router ID of each neighbor it has seen.
- If $R1$ observes that Hello packet sent by $R2$ contains its router ID field in the packet, it is assured that communication is bidirectional.
- After neighbor discovery, designated routers are elected in each multiaccess network.
- Election is based on highest value of priority and ID fields.

Type 1: OSPF Hello Packets

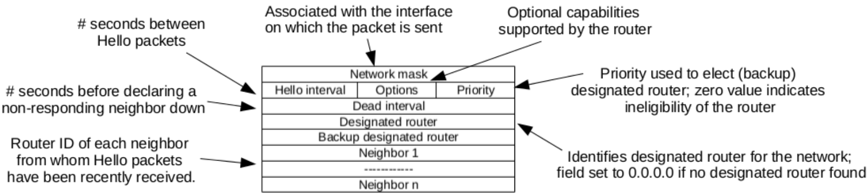

Figure : Hello packet fields

- Hello packets are sent to its neighbors periodically to discover, establish, and maintain neighbor relationships.
- Each router $R1$ broadcasts Hello packets periodically onto its network.
- When a router $R2$ receives a Hello packet, it replies with a Hello packet with router ID of each neighbor it has seen.
- If $R1$ observes that Hello packet sent by $R2$ contains its router ID field in the packet, it is assured that communication is bidirectional.
- After neighbor discovery, designated routers are elected in each multiaccess network.
- Election is based on highest value of priority and ID fields.

Type 2: OSPF Database Description Packets

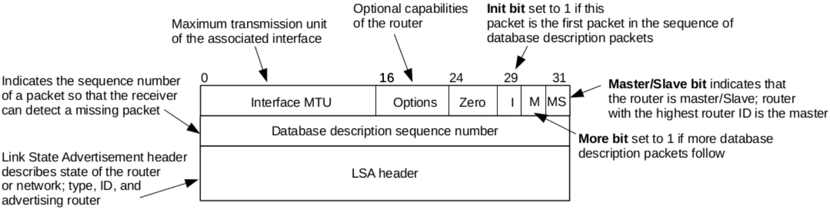

Figure : OSPF database description packet

Type 2: OSPF Database Description Packets

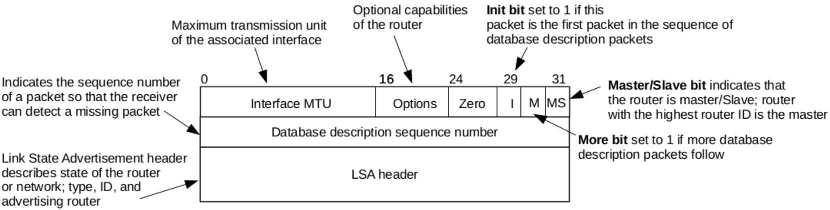

Figure : OSPF database description packet

Stage 2: Establishing adjacencies and synchronizing databases

- Involves establishing adjacencies between a subset of routers in AS.
- Once two neighboring routers establish connectivity, they exchange database description packets to synchronize their link state databases.
- One router acts as master; other as slave. Multiple packets can be used to describe the link state database.
- A database description packet can contain multiple *link state advertisement* (LSA) headers.

Type 2: OSPF Database Description Packets

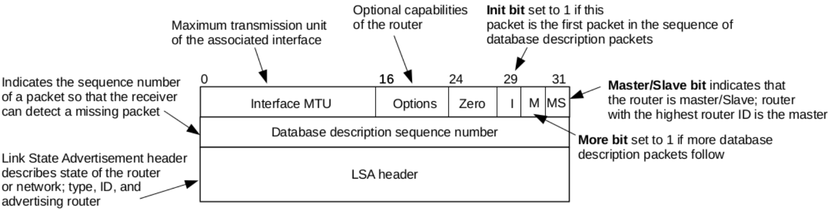

Figure : OSPF database description packet

Stage 2: Establishing adjacencies and synchronizing databases

- Involves establishing adjacencies between a subset of routers in AS.
- Once two neighboring routers establish connectivity, they exchange database description packets to synchronize their link state databases.
- One router acts as master; other as slave. Multiple packets can be used to describe the link state database.
- A database description packet can contain multiple *link state advertisement* (LSA) headers.

Type 2: OSPF Database Description Packets

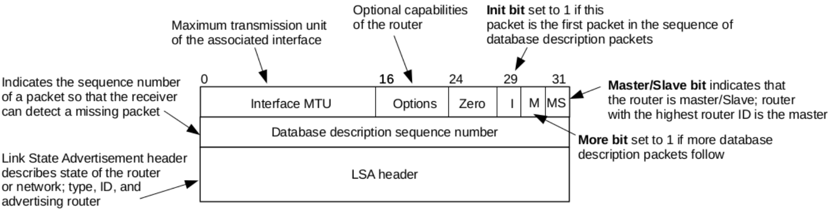

Figure : OSPF database description packet

Stage 2: Establishing adjacencies and synchronizing databases

- Involves establishing adjacencies between a subset of routers in AS.
- Once two neighboring routers establish connectivity, they exchange database description packets to synchronize their link state databases.
- One router acts as master; other as slave. Multiple packets can be used to describe the link state database.
- A database description packet can contain multiple *link state advertisement* (LSA) headers.

Type 2: OSPF Database Description Packets

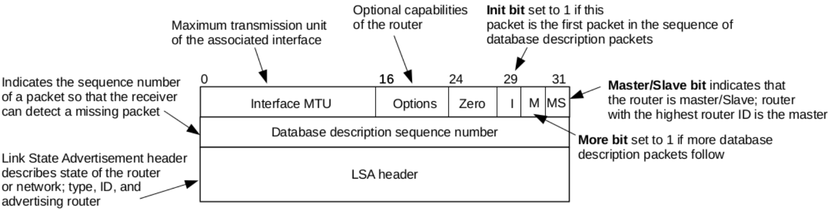

Figure : OSPF database description packet

Stage 2: Establishing adjacencies and synchronizing databases

- Involves establishing adjacencies between a subset of routers in AS.
- Once two neighboring routers establish connectivity, they exchange database description packets to synchronize their link state databases.
- One router acts as master; other as slave. Multiple packets can be used to describe the link state database.
- A database description packet can contain multiple *link state advertisement* (LSA) headers.

Type 2: OSPF Database Description Packets

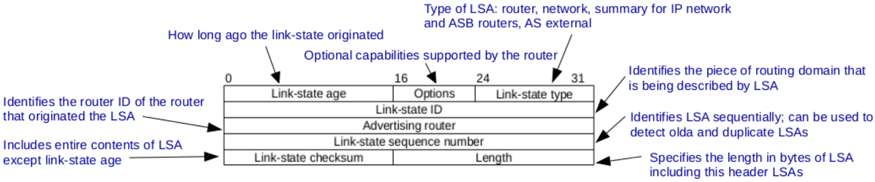

Figure : OSPF Link-State Advertisement (LSA) header fields

Type 2: OSPF Database Description Packets

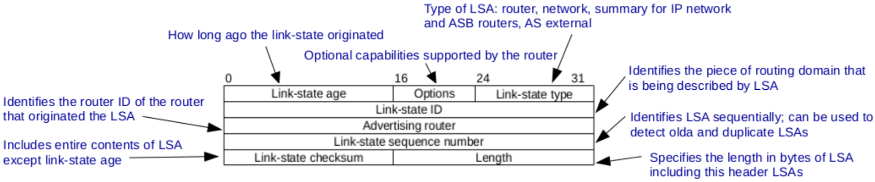

Figure : OSPF Link-State Advertisement (LSA) header fields

Type 2: OSPF Database Description Packets

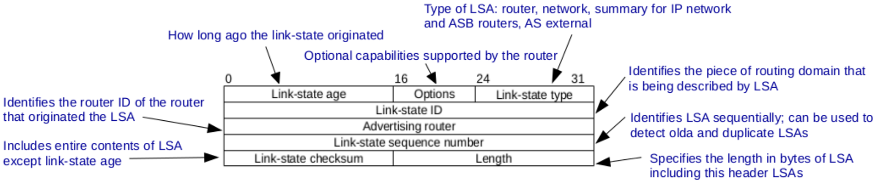

Figure : OSPF Link-State Advertisement (LSA) header fields

- Routers only send their LSA headers instead of entire database.
- Neighbor can then request LSAs that it does not have.
- *Router link advertisements*: Generated by all OSPF routers; it gives state of router links in the area only.
- *Network link advertisement*: Generated by the designated router; it lists the routers connected to broadcast and NBMA network.
- *Summary link advertisement*: Generated by ABRs; gives routes to destinations in other areas and routes to ASBRs.
- *AS external link advertisement*: Generated by ASBRs; describes routes to destinations outside the OSPF network.

Type 2: OSPF Database Description Packets

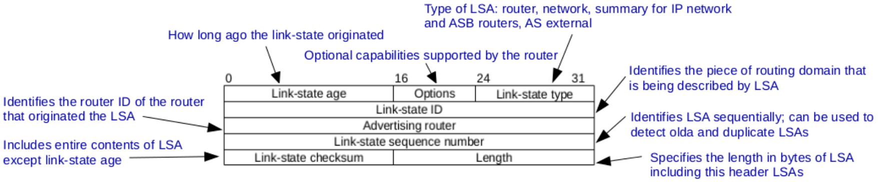

Figure : OSPF Link-State Advertisement (LSA) header fields

- Routers only send their LSA headers instead of entire database.
- Neighbor can then request LSAs that it does not have.
- *Router link advertisements*: Generated by all OSPF routers; it gives state of router links in the area only.
- *Network link advertisement*: Generated by the designated router; it lists the routers connected to broadcast and NBMA network.
- *Summary link advertisement*: Generated by ABRs; gives routes to destinations in other areas and routes to ASBRs.
- *AS external link advertisement*: Generated by ASBRs; describes routes to destinations outside the OSPF network.

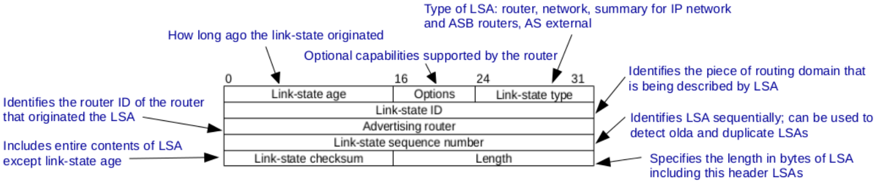

Figure : OSPF Link-State Advertisement (LSA) header fields

- Routers only send their LSA headers instead of entire database.
- Neighbor can then request LSAs that it does not have.
- *Router link advertisements*: Generated by all OSPF routers; it gives state of router links in the area only.
- *Network link advertisement*: Generated by the designated router; it lists the routers connected to broadcast and NBMA network.
- *Summary link advertisement*: Generated by ABRs; gives routes to destinations in other areas and routes to ASBRs.
- *AS external link advertisement*: Generated by ASBRs; describes routes to destinations outside the OSPF network.

Type 2: OSPF Database Description Packets

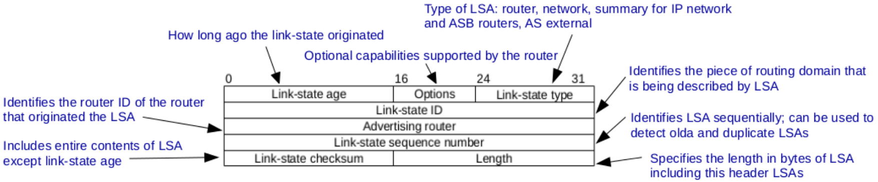

Figure : OSPF Link-State Advertisement (LSA) header fields

- Routers only send their LSA headers instead of entire database.
- Neighbor can then request LSAs that it does not have.
- *Router link advertisements*: Generated by all OSPF routers; it gives state of router links in the area only.
- *Network link advertisement*: Generated by the designated router; it lists the routers connected to broadcast and NBMA network.
- *Summary link advertisement*: Generated by ABRs; gives routes to destinations in other areas and routes to ASBRs.
- *AS external link advertisement*: Generated by ASBRs; describes routes to destinations outside the OSPF network.

Type 2: OSPF Database Description Packets

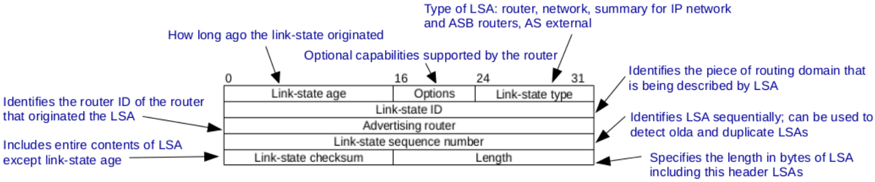

Figure : OSPF Link-State Advertisement (LSA) header fields

- Routers only send their LSA headers instead of entire database.
- Neighbor can then request LSAs that it does not have.
- *Router link advertisements*: Generated by all OSPF routers; it gives state of router links in the area only.
- *Network link advertisement*: Generated by the designated router; it lists the routers connected to broadcast and NBMA network.
- *Summary link advertisement*: Generated by ABRs; gives routes to destinations in other areas and routes to ASBRs.
- *AS external link advertisement*: Generated by ASBRs; describes routes to destinations outside the OSPF network.

Type 2: OSPF Database Description Packets

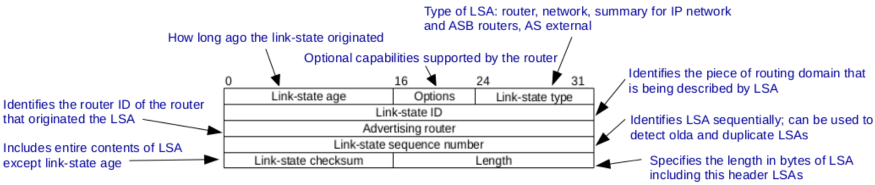

Figure : OSPF Link-State Advertisement (LSA) header fields

- Routers only send their LSA headers instead of entire database.
- Neighbor can then request LSAs that it does not have.
- *Router link advertisements*: Generated by all OSPF routers; it gives state of router links in the area only.
- *Network link advertisement*: Generated by the designated router; it lists the routers connected to broadcast and NBMA network.
- *Summary link advertisement*: Generated by ABRs; gives routes to destinations in other areas and routes to ASBRs.
- *AS external link advertisement*: Generated by ASBRs; describes routes to destinations outside the OSPF network.

Type 3: Link State Request packet

0	31
Link-state type	
Link-state ID	
Advertising Router	

Figure : OSPF link-state request packet

Type 3: Link State Request packet

0 … 31

Link-state type
Link-state ID
Advertising Router

Figure : OSPF link-state request packet

Stage 3: Propagation of link state information and building of routing tables

Type 3: Link State Request packet

0	31
Link-state type	
Link-state ID	
Advertising Router	

Figure : OSPF link-state request packet

Stage 3: Propagation of link state information and building of routing tables

- When a router wants to update link-state database, it sends *link state request packet* to its neighbor listing the LSAs it needs.
- An LSA request comprises links-state type, link-state ID, and the advertising router; the three fields are repeated for each link.
- **Response to link-state request**: If a router finds that its link-state has changed, it sends the new link-state info using the *link-state update message*.
- OSPF uses reliable flooding to ensure LSAs are updated correctly.

Type 3: Link State Request packet

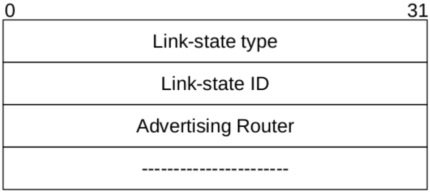

Figure : OSPF link-state request packet

Stage 3: Propagation of link state information and building of routing tables

- When a router wants to update link-state database, it sends *link state request packet* to its neighbor listing the LSAs it needs.
- An LSA request comprises links-state type, link-state ID, and the advertising router; the three fields are repeated for each link.
- **Response to link-state request**: If a router finds that its link-state has changed, it sends the new link-state info using the *link-state update message*.
- OSPF uses reliable flooding to ensure LSAs are updated correctly.

Type 3: Link State Request packet

0   31
Link-state type
Link-state ID
Advertising Router

Figure : OSPF link-state request packet

Stage 3: Propagation of link state information and building of routing tables

- When a router wants to update link-state database, it sends *link state request packet* to its neighbor listing the LSAs it needs.
- An LSA request comprises links-state type, link-state ID, and the advertising router; the three fields are repeated for each link.
- **Response to link-state request**: If a router finds that its link-state has changed, it sends the new link-state info using the *link-state update message*.
- OSPF uses reliable flooding to ensure LSAs are updated correctly.

Type 3: Link State Request packet

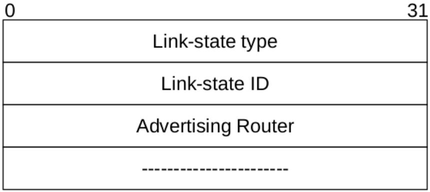

Figure : OSPF link-state request packet

Stage 3: Propagation of link state information and building of routing tables

- When a router wants to update link-state database, it sends *link state request packet* to its neighbor listing the LSAs it needs.
- An LSA request comprises links-state type, link-state ID, and the advertising router; the three fields are repeated for each link.
- **Response to link-state request**: If a router finds that its link-state has changed, it sends the new link-state info using the *link-state update message*.
- OSPF uses reliable flooding to ensure LSAs are updated correctly.

Stage 3: Propagation of link state information and building of routing tables

0 31

Number of LSAs
LSA 1

LSA n

Figure : OSPF link-state update packet

Stage 3: Propagation of link state information and building of routing tables

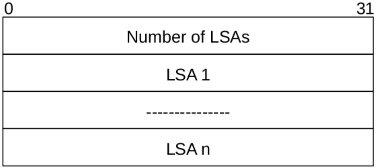

Figure : OSPF link-state update packet

- Suppose that a router's local state has changed, so the router need to update its LSA.
- The router issues a *link-state update* packet that invokes the flooding procedure.
- After receiving, a neighbor router examines LSAs in the update.
- Neighbor installs each LSA in update packet that is more recent tham the corresponding LSA in its database.
- Neighbor sends *LSA acknowledgement packet* back to the router; ACK packets comprise a list of LSA headers.
- Neighbor also prepare *link-state update packet* that contains LSA and floods it on all interfaces except the one on which it received the LSA.

Stage 3: Propagation of link state information and building of routing tables

0 … 31
Number of LSAs
LSA 1

LSA n

Figure : OSPF link-state update packet

- Suppose that a router's local state has changed, so the router need to update its LSA.
- The router issues a *link-state update* packet that invokes the flooding procedure.
- After receiving, a neighbor router examines LSAs in the update.
- Neighbor installs each LSA in update packet that is more recent tham the corresponding LSA in its database.
- Neighbor sends *LSA acknowledgement packet* back to the router; ACK packets comprise a list of LSA headers.
- Neighbor also prepare *link-state update packet* that contains LSA and floods it on all interfaces except the one on which it received the LSA.

Stage 3: Propagation of link state information and building of routing tables

0 ... 31
Number of LSAs
LSA 1

LSA n

Figure : OSPF link-state update packet

- Suppose that a router's local state has changed, so the router need to update its LSA.
- The router issues a *link-state update* packet that invokes the flooding procedure.
- After receiving, a neighbor router examines LSAs in the update.
- Neighbor installs each LSA in update packet that is more recent tham the corresponding LSA in its database.
- Neighbor sends *LSA acknowledgement packet* back to the router; ACK packets comprise a list of LSA headers.
- Neighbor also prepare *link-state update packet* that contains LSA and floods it on all interfaces except the one on which it received the LSA.

Stage 3: Propagation of link state information and building of routing tables

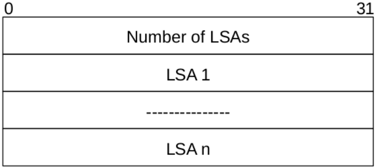

Figure : OSPF link-state update packet

- Suppose that a router's local state has changed, so the router need to update its LSA.
- The router issues a *link-state update* packet that invokes the flooding procedure.
- After receiving, a neighbor router examines LSAs in the update.
- Neighbor installs each LSA in update packet that is more recent tham the corresponding LSA in its database.
- Neighbor sends *LSA acknowledgement packet* back to the router; ACK packets comprise a list of LSA headers.
- Neighbor also prepare *link-state update packet* that contains LSA and floods it on all interfaces except the one on which it received the LSA.

Stage 3: Propagation of link state information and building of routing tables

0 ... 31
Number of LSAs
LSA 1

LSA n

Figure : OSPF link-state update packet

- Suppose that a router's local state has changed, so the router need to update its LSA.
- The router issues a *link-state update* packet that invokes the flooding procedure.
- After receiving, a neighbor router examines LSAs in the update.
- Neighbor installs each LSA in update packet that is more recent tham the corresponding LSA in its database.
- Neighbor sends *LSA acknowledgement packet* back to the router; ACK packets comprise a list of LSA headers.
- Neighbor also prepare *link-state update packet* that contains LSA and floods it on all interfaces except the one on which it received the LSA.

Stage 3: Propagation of link state information and building of routing tables

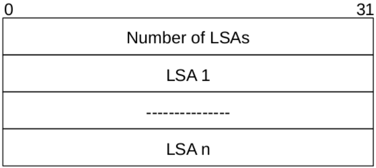

Figure : OSPF link-state update packet

- Suppose that a router's local state has changed, so the router need to update its LSA.
- The router issues a *link-state update* packet that invokes the flooding procedure.
- After receiving, a neighbor router examines LSAs in the update.
- Neighbor installs each LSA in update packet that is more recent tham the corresponding LSA in its database.
- Neighbor sends *LSA acknowledgement packet* back to the router; ACK packets comprise a list of LSA headers.
- Neighbor also prepare *link-state update packet* that contains LSA and floods it on all interfaces except the one on which it received the LSA.

Stage 3: Propagation of link state information and building of routing tables

- All routers eventually receive update LSA.
- After updating link-state database, the router needs to recompute the `shortest path algorithm` and modify `routing table` according to updated info.
- A router retransmits an LSA that has been sent to a neighbor peridocally until receiving an `acknowledgement` from the neighbor.
- OSPFs requires all originators of LSAs to refresh their LSAs after every 30 minutes.

Stage 3: Propagation of link state information and building of routing tables

- All routers eventually receive update LSA.
- After updating link-state database, the router needs to recompute the `shortest path algorithm` and modify `routing table` according to updated info.
- A router retransmits an LSA that has been sent to a neighbor peridocally until receiving an `acknowledgement` from the neighbor.
- OSPFs requires all originators of LSAs to refresh their LSAs after every 30 minutes.

Stage 3: Propagation of link state information and building of routing tables

- All routers eventually receive update LSA.
- After updating link-state database, the router needs to recompute the `shortest path algorithm` and modify `routing table` according to updated info.
- A router retransmits an LSA that has been sent to a neighbor peridocally until receiving an `acknowledgement` from the neighbor.
- OSPFs requires all originators of LSAs to refresh their LSAs after every 30 minutes.

Stage 3: Propagation of link state information and building of routing tables

- All routers eventually receive update LSA.
- After updating link-state database, the router needs to recompute the `shortest path algorithm` and modify `routing table` according to updated info.
- A router retransmits an LSA that has been sent to a neighbor peridocally until receiving an `acknowledgement` from the neighbor.
- OSPFs requires all originators of LSAs to refresh their LSAs after every 30 minutes.

ICMP message types

ICMP Type	Code	Description
0	0	echo reply (to ping)
3	0	destination network unreachable
3	1	destination host unreachable
3	2	destination protocol unrachable
3	3	destination port unreachable
3	6	destination network unknown
3	7	destination host unknown
4	0	source quench (congestion control)
8	0	echo request
9	0	router advertisement
10	0	router discovery
11	0	TTL expiry
12	0	IP header bad

IPv4 datagram format

Version	Header length	Type of service	Datagram length (bytes)
16-bit identifier		Flags	13-bit Fragmentation offset
Time-to-live	Upper-layer protocol	Header checksum	
32-bit Source IP address			
32-bit Destination IP address			
Options (if any)			
Data			

Table : IPv4 datagram format: each row is of 32-bit width.

IPv6 datagram format

Version	Traffic class	Flow label	
Payload length		Next hdr	Hop limit
Source address (128 bits)			
Destination address (128 bits)			
Data			

IPv6 datagram format

<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td colspan="2">Payload length</td><td>Next hdr</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source address (128 bits)</td></tr>
<tr><td colspan="4">Destination address (128 bits)</td></tr>
<tr><td colspan="4">Data</td></tr>
</table>

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

IPv6 datagram format

<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td colspan="2">Payload length</td><td>Next hdr</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source address (128 bits)</td></tr>
<tr><td colspan="4">Destination address (128 bits)</td></tr>
<tr><td colspan="4">Data</td></tr>
</table>

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

IPv6 datagram format

<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td colspan="2">Payload length</td><td>Next hdr</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source address (128 bits)</td></tr>
<tr><td colspan="4">Destination address (128 bits)</td></tr>
<tr><td colspan="4">Data</td></tr>
</table>

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

IPv6 datagram format

<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td colspan="2">Payload length</td><td>Next hdr</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source address (128 bits)</td></tr>
<tr><td colspan="4">Destination address (128 bits)</td></tr>
<tr><td colspan="4">Data</td></tr>
</table>

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

IPv6 datagram format

<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td colspan="2">Payload length</td><td>Next hdr</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source address (128 bits)</td></tr>
<tr><td colspan="4">Destination address (128 bits)</td></tr>
<tr><td colspan="4">Data</td></tr>
</table>

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

IPv6 datagram format

<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td colspan="2">Payload length</td><td>Next hdr</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source address (128 bits)</td></tr>
<tr><td colspan="4">Destination address (128 bits)</td></tr>
<tr><td colspan="4">Data</td></tr>
</table>

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

IPv6 datagram format

<table>
<tr><td>Version</td><td>Traffic class</td><td colspan="2">Flow label</td></tr>
<tr><td colspan="2">Payload length</td><td>Next hdr</td><td>Hop limit</td></tr>
<tr><td colspan="4">Source address (128 bits)</td></tr>
<tr><td colspan="4">Destination address (128 bits)</td></tr>
<tr><td colspan="4">Data</td></tr>
</table>

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

IPv6 datagram format

Version	Traffic class	Flow label	
Payload length		Next hdr	Hop limit
Source address (128 bits)			
Destination address (128 bits)			
Data			

Fields in IPv6

- *Version*: 4-bit field, IP version number. Carries a value of 6 in this field.
- *Traffic class*: 8-bit field similar to TOS field in IPv4.
- *Flow label*: 20-bit field used to identify a flow of datagrams.
- *Payload length*: 16-bit value; # bytes in IPv6 datagram following the 40-byte datagram header.
- *Next header*: Identifies the protocol to which the contents of this datagram will be delivered (TCP/UDP/ICMP).
- *Hop limit*: Contents of this field is decremented by one by each router that forwards the datagram.Packet discarded if hop limit reaches zero.
- *Source and destination addresses*: Various formats of IPv6 128-bit addresses.
- *Data*: Payload portion of IPv6 datagram. When datagram reaches its destination, the payload shall be removed from the IPv6 datagram and passed on to the protocol specified in the *Next header* field.

Fields in IPv4 but not in IPv6 datagram

- *Fragmentation and Reassembly*: IPv6 does not fragmentation and reassembly at routers; operations to be performed by only source and destination. If IPv6 datagram is big for a router, it sends "Packet Too Big" ICMP error message to the sender.

- *Header Checksum*: Too costly operation in IPv4. Checksum already computed in transport layer (for e.g. TCP, UDP) and link layer (for e.g. Ethernet). TTL field keeps changing across routers, checksum need to be recomputed at each each router.

- *Options*: No longer a part of IPv6 header. Mostly used as Next Header field in IPv4.

Fields in IPv4 but not in IPv6 datagram

- *Fragmentation and Reassembly*: IPv6 does not fragmentation and reassembly at routers; operations to be performed by only source and destination. If IPv6 datagram is big for a router, it sends "Packet Too Big" ICMP error message to the sender.

- *Header Checksum*: Too costly operation in IPv4. Checksum already computed in transport layer (for e.g. TCP, UDP) and link layer (for e.g. Ethernet). TTL field keeps changing across routers, checksum need to be recomputed at each each router.

- *Options*: No longer a part of IPv6 header. Mostly used as Next Header field in IPv4.

Fields in IPv4 but not in IPv6 datagram

- *Fragmentation and Reassembly*: IPv6 does not fragmentation and reassembly at routers; operations to be performed by only source and destination. If IPv6 datagram is big for a router, it sends "Packet Too Big" ICMP error message to the sender.
- *Header Checksum*: Too costly operation in IPv4. Checksum already computed in transport layer (for e.g. TCP, UDP) and link layer (for e.g. Ethernet). TTL field keeps changing across routers, checksum need to be recomputed at each each router.
- *Options*: No longer a part of IPv6 header. Mostly used as Next Header field in IPv4.

Link Layer

Terms and related aspects:

- Any network device that runs a link layer protocol is a **node**.
- Communication channels that connect adjacent nodes along communication paths are called **links**.
- **Basic service of any link layer**: Move a datagram from one to another node over a single communication link.

Link Layer

Terms and related aspects:

- Any network device that runs a link layer protocol is a **node**.
- Communication channels that connect adjacent nodes along communication paths are called **links**.
- **Basic service of any link layer**: Move a datagram from one to another node over a single communication link.

Link Layer

Terms and related aspects:

- Any network device that runs a link layer protocol is a **node**.
- Communication channels that connect adjacent nodes along communication paths are called **links**.
- **Basic service of any link layer**: Move a datagram from one to another node over a single communication link.

Services of a link layer

Possible services of a link layer protocol

- **Framing**:
  - Encapsulate each network-layer datagram within a link-layer frame before transmission over the link.
  - Frame consists of a data field, in which the network-layer datagram is inserted, and a number of header fields.

- **Link access**: A **medium access control (MAC)** protocol specifies the rules by which a frame is transmitted onto the link; a link can be a *point-to-point link* or a *broadcast link*.

- **Reliable delivery**:
  - If a link provides reliable delivery service, the transported link-layer frames does not have any error.
  - Can be achieved through acknowledgements and retransmissions.
  - A link-layer reliable delivery service is often used for links that are prone to high error rates, such as a wireless link, with the goal of correcting an error locally on the link.

Services of a link layer

Possible services of a link layer protocol

- **Framing**:
  - Encapsulate each network-layer datagram within a link-layer frame before transmission over the link.
  - Frame consists of a data field, in which the network-layer datagram is inserted, and a number of header fields.
- **Link access**: A **medium access control (MAC)** protocol specifies the rules by which a frame is transmitted onto the link; a link can be a *point-to-point link* or a *broadcast link*.
- **Reliable delivery**:
  - If a link provides reliable delivery service, the transported link-layer frames does not have any error.
  - Can be achieved through acknowledgements and retransmissions.
  - A link-layer reliable delivery service is often used for links that are prone to high error rates, such as a wireless link, with the goal of correcting an error locally on the link.

Services of a link layer

Possible services of a link layer protocol

- **Framing**:
  - Encapsulate each network-layer datagram within a link-layer frame before transmission over the link.
  - Frame consists of a data field, in which the network-layer datagram is inserted, and a number of header fields.
- **Link access**: A **medium access control (MAC)** protocol specifies the rules by which a frame is transmitted onto the link; a link can be a *point-to-point link* or a *broadcast link*.
- **Reliable delivery**:
  - If a link provides reliable delivery service, the transported link-layer frames does not have any error.
  - Can be achieved through acknowledgements and retransmissions.
  - A link-layer reliable delivery service is often used for links that are prone to high error rates, such as a wireless link, with the goal of correcting an error locally on the link.

Services of a link layer

Possible services of a link layer protocol

- **Error detection and correction**:
  - Link-layer hardware in a receiving node can incorrectly decide that a bit in a frame is zero when it was transmitted as a one, and vice versa.
  - Such bit errors are introduced by signal attenuation and electromagnetic noise.
  - Transmitting node includes error-detection bits in the frame, and the receiving node performs an error check.
  - Error detection in the link layer is usually more sophisticated and is implemented in hardware.
  - Error correction is similar to error detection, except that a receiver not only detects when bit errors have occurred in the frame but also determines exactly where in the frame the errors have occurred (and then corrects these errors).

Services of a link layer

Possible services of a link layer protocol

- **Error detection and correction**:
  - Link-layer hardware in a receiving node can incorrectly decide that a bit in a frame is zero when it was transmitted as a one, and vice versa.
  - Such bit errors are introduced by signal attenuation and electromagnetic noise.
  - Transmitting node includes error-detection bits in the frame, and the receiving node performs an error check.
  - Error detection in the link layer is usually more sophisticated and is implemented in hardware.
  - Error correction is similar to error detection, except that a receiver not only detects when bit errors have occurred in the frame but also determines exactly where in the frame the errors have occurred (and then corrects these errors).

Services of a link layer

Possible services of a link layer protocol

- **Error detection and correction**:
  - Link-layer hardware in a receiving node can incorrectly decide that a bit in a frame is zero when it was transmitted as a one, and vice versa.
  - Such bit errors are introduced by signal attenuation and electromagnetic noise.
  - Transmitting node includes error-detection bits in the frame, and the receiving node performs an error check.
  - Error detection in the link layer is usually more sophisticated and is implemented in hardware.
  - Error correction is similar to error detection, except that a receiver not only detects when bit errors have occurred in the frame but also determines exactly where in the frame the errors have occurred (and then corrects these errors).

Services of a link layer

Possible services of a link layer protocol

- **Error detection and correction**:
  - Link-layer hardware in a receiving node can incorrectly decide that a bit in a frame is zero when it was transmitted as a one, and vice versa.
  - Such bit errors are introduced by signal attenuation and electromagnetic noise.
  - Transmitting node includes error-detection bits in the frame, and the receiving node performs an error check.
  - Error detection in the link layer is usually more sophisticated and is implemented in hardware.
  - Error correction is similar to error detection, except that a receiver not only detects when bit errors have occurred in the frame but also determines exactly where in the frame the errors have occurred (and then corrects these errors).

Services of a link layer

Possible services of a link layer protocol

- **Error detection and correction**:
  - Link-layer hardware in a receiving node can incorrectly decide that a bit in a frame is zero when it was transmitted as a one, and vice versa.
  - Such bit errors are introduced by signal attenuation and electromagnetic noise.
  - Transmitting node includes error-detection bits in the frame, and the receiving node performs an error check.
  - Error detection in the link layer is usually more sophisticated and is implemented in hardware.
  - Error correction is similar to error detection, except that a receiver not only detects when bit errors have occurred in the frame but also determines exactly where in the frame the errors have occurred (and then corrects these errors).

Services of a link layer

Possible services of a link layer protocol

- **Error detection and correction**:
  - Link-layer hardware in a receiving node can incorrectly decide that a bit in a frame is zero when it was transmitted as a one, and vice versa.
  - Such bit errors are introduced by signal attenuation and electromagnetic noise.
  - Transmitting node includes error-detection bits in the frame, and the receiving node performs an error check.
  - Error detection in the link layer is usually more sophisticated and is implemented in hardware.
  - Error correction is similar to error detection, except that a receiver not only detects when bit errors have occurred in the frame but also determines exactly where in the frame the errors have occurred (and then corrects these errors).