Advanced Computer Networks

Internal Routing - Link State protocols

Prof. Andrzej Duda
duda@imag.fr

http://duda.imag.fr

1

Contents

- Link state
  - flooding topology information
  - finding the shortest paths (Dijkstra)
  - areas - hierarchical routing
- OSPF
  - neighbor discovery - Hello protocol
  - database synchronization
  - link state updates
  - examples

2

Link State Routing

- Principles
  - estimate metrics with neighbors
    - bandwidth, delay, cost (fixed by administrator)
  - build a packet with the metrics of all neighbors
  - flood to all routers
  - compute the shortest path to all destinations (Dijkstra)
  - update if modification of topology
- Used in OSPF (Open Shortest Path First) and PNNI (ATM routing protocol)

3

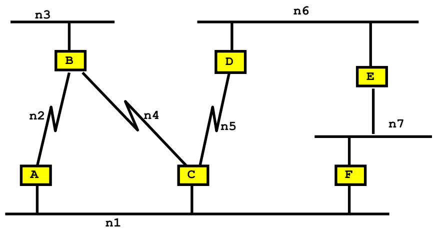
Topology Database Synchronization

- Neighbouring nodes synchronize before starting any relationship
  - Hello protocol; keep alive
  - initial synchronization of database
  - description of all links (no information yet)
- Once synchronized, a node accepts link state advertisements
  - contain a sequence number, stored with record in the database
  - only messages with new sequence number are accepted
  - accepted messages are flooded to all neighbours
  - sequence number prevents anomalies (loops or blackholes)

4

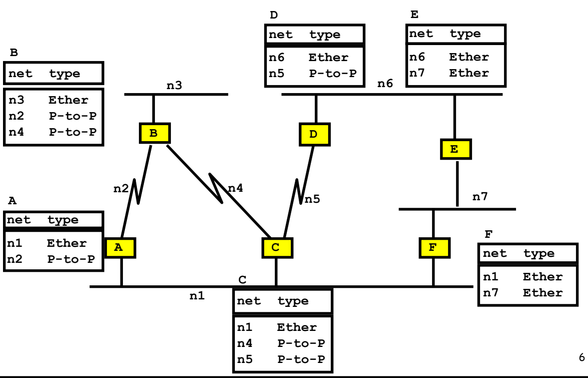
Example network

- Each router knows directly connected networks



5

Initial routing tables



6

Flooding

- The local metric information is flooded to all routers
- After convergence, all routers have the same information

A

net	cost
n1	10
n2	100

B

net	cost
n3	10
n2	100
n4	100

C

net	cost
n1	10
n4	100
n5	100

D

net	cost
n6	10
n5	100

E

net	cost
n6	10
n7	10

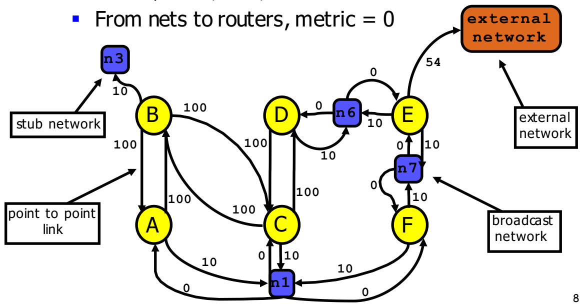
F

net	cost
n1	10
n7	10

7

Topology graph

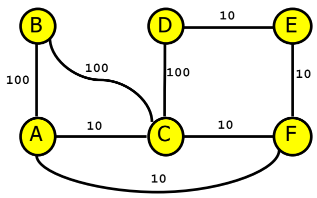
- Arrows to nets with a given metric
  - except P-to-P, stub, and external networks
- From nets to routers, metric = 0



8

Simplified graph

- Only arrows with metrics between routers
- Execute the SPF (Shortest Path First - Dijkstra) algorithm on the graph



9

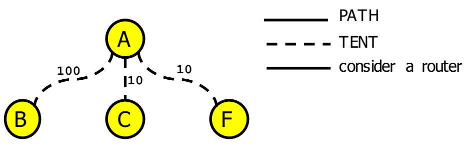
SPF at *A*

1. Initialization
   1. *PATH* variable: router *A* (the best path to destination)
   2. *TENT* variable: *empty* (tentative paths)
2. For each router *N* in *PATH*
   1. for each neighbor *M* of *N*
      1. *c(A, M) = c(A, N) + c(N, M)*
   2. if *M* is not in *PATH* nor in *TENT* with a better cost, insert *M* with *N* in *TENT*
3. If *TENT* is empty, end. Otherwise take the entry with the best cost from *TENT*, insert it into *PATH* and go to 2.

At the end PATH contains the tree of best paths to all destinations

10

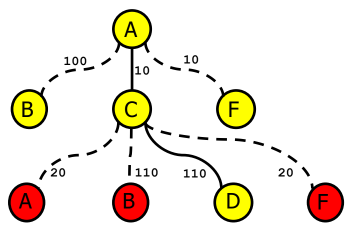
Executing SPF




```
Before: TENT: A-B(100), A-C(10), A-F(10)
        PATH: A
After:  TENT: A-B(100), A-F(10)
        PATH: A-C(10)
```

11

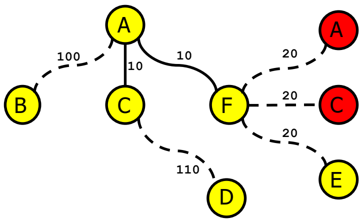
Executing SPF




```
Before: TENT: A-B(100), A-F(10), C-D(110)
        PATH: A-C(10)
After:  TENT: A-B(100), C-D(110)
        PATH: A-C(10), A-F(10)
```

12

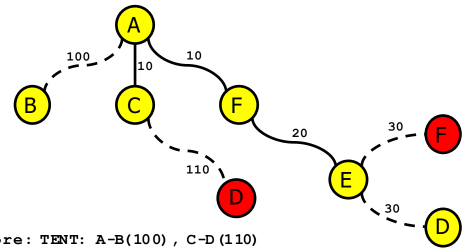
Executing SPF




```
Before: TENT: A-B(100), C-D(110), F-E(20)
        PATH: A-C(10), A-F(10)
After:  TENT: A-B(100), C-D(110)
        PATH: A-C(10), A-F(10), F-E(20)
```

13

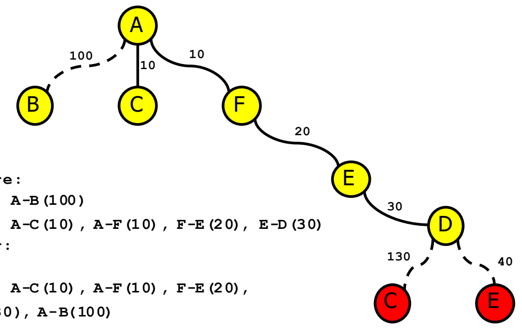
Executing SPF




```
Before: TENT: A-B(100), C-D(110)
        TENT: A-B(100), E-D(30)
        PATH: A-C(10), A-F(10), F-E(20)
After:  TENT: A-B(100)
        PATH: A-C(10), A-F(10), F-E(20), E-D(30)
```

14

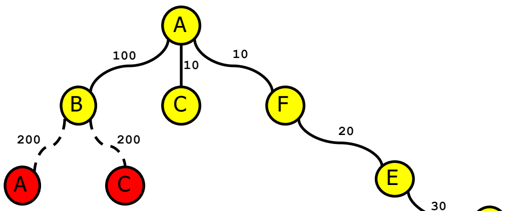
Executing SPF




```
Before:
TENT: A-B(100)
PATH: A-C(10), A-F(10), F-E(20), E-D(30)
After:
TENT:
PATH: A-C(10), A-F(10), F-E(20),
E-D(30), A-B(100)
```

15

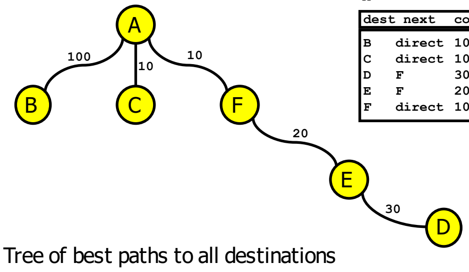
Executing SPF




```
Before: TENT:
        PATH: A-C(10), A-F(10), F-E(20),
              E-D(30), A-B(100)
After:  TENT:
        PATH: A-C(10), A-F(10), F-E(20),
              E-D(30), A-B(100)
```

16

Result



A

dest	next	cost
B	direct	100
C	direct	10
D	F	30
E	F	20
F	direct	10

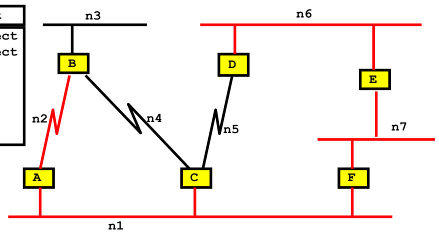
- Tree of best paths to all destinations

17

Routing table of A

A

net	next
n1	direct
n2	direct
n3	B
n4	C
n5	C
n6	F
n7	F



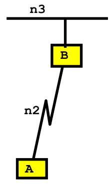
18

Towards OSPF

- OSPF (Open Shortest Path First)
  - Link State protocol
  - Link State information: LSA (Link State Advertisement)
  - different sub-protocols: Hello, Database Description, Link State flooding
- It allows to
  - separate hosts and routers
  - consider different types of networks
    - broadcast (Ethernet), NBMA (ATM, X.25), point-to-point (PPP)
  - divide large networks into several areas
  - independent route computing in each area

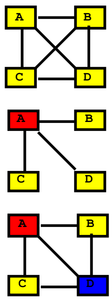
Separate hosts and routers

- Link should be described in the DB
  - link between a router and each host, but LANs in most cases: advertize the link to the "stub network"
  - link of the form of a broadcast network (Ethernet)
    - IP address of the subnetwork (stub network)
    - e.g. `n3` identified by `128.88.38/24`
  - link to a neighbor router
    - IP address of the neighbor router
    - e.g. `n2` identified by `176.44.23.254`
    - no IP address assigned to the interface
      – interface index



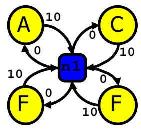
Designated routers

- Number of neighbors
  - if *n* routers, *n*(*n*-1)/2 neighbors
- Election of a designated router on a LAN
  - *n*-1 neighbors
  - flooding
    - advertise to `224.0.0.6` (all designated routers)
    - flooded to `224.0.0.5` (all routers)
  - back-up designated router
    - listens to advertisements, but does not flood
    - failure of the designated router detected by Hello
      – back-up becomes designated router



Virtual networks

- LAN represented as a virtual network
  - less entries in the DB
  - real cost to `n1`, zero to routers



NBMA networks and P-to-P

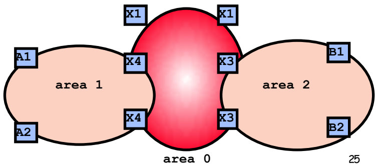
- NBMA (Non Broadcast, Multiple Access)
  - several hosts, but no broadcast
- Virtual circuits between all hosts - each link appears in the database
- Managed as broadcast networks
  - designated and back-up router
  - permanent virtual circuits only to them
- Flooding
  - designated router sends a copy of update to all routers

Divide large networks

- Why divide large networks?
- Cost of computing routing tables
  - update when topology changes
  - SPF algorithm
    - *n* routers, *k* links
    - complexity O(*n***k*)
  - size of DB, update messages grows with the network size
- Limit the scope of updates and computational overhead
  - divide the network into several areas
  - independent route computing in each area
  - inject aggregated information on routes into other areas

Hierarchical Routing

- A large OSPF domain can be configured into *areas*
  - one *backbone area* (area 0)
  - non backbone areas (areas numbered other than 0)
- All inter-area traffic goes through area 0
  - strict hierarchy
- Inside one area: link state routing as seen earlier
  - one topology database per area



25

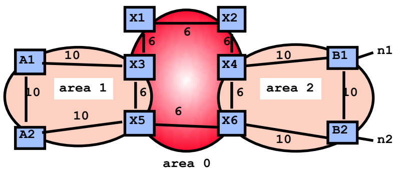
Principles

- Routing method used in the higher level:
  - *distance vector*
  - no problem with loops - one backbone area
- Mapping of higher level nodes to lower level nodes
  - area border routers (inter-area routers) belong to two areas
- Inter-level routing information
  - summary link state advertisements (LSA) from other areas are injected into the local topology databases

26

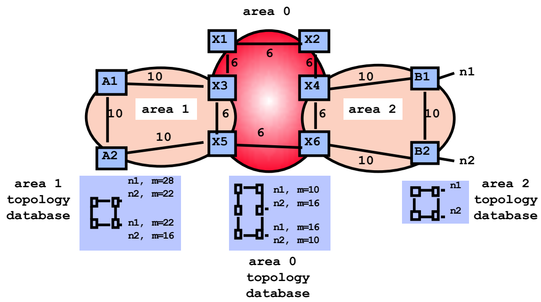
Example

- Assume networks **n1** and **n2** become visible at time 0. Show the topology databases at all routers



27

Solution



28

Explanations

- All routers in area 2 propagate the existence of n1 and n2, directly attached to B1 (resp. B2).
- Area border routers X4 and X6 belong to area 2, thus they can compute their distances to n1 and n2
- Area border routers X4 and X6 inject their distances to n1 and n2 into the area 0 topology database (item 3 of the principle). The corresponding summary LSA is propagated to all routers of area 0.
- All routers in area 0 can now compute their distance to n1 and n2, using their distances to X4 and X6, and using the principle of distance vector (item 1 of the principle).

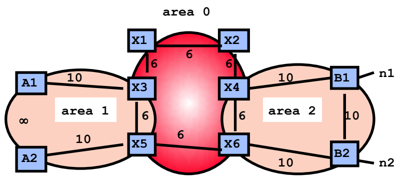
29

Comments

- Distance vector computation causes none of the RIP problems
  - strict hierarchy: no loop between areas
- External and summary LSA for all reachable networks are present in all topology databases of all areas
  - most LSAs are external
  - can be avoided in configuring some areas as terminal: use `default` entry to the backbone
- Area partitions require specific support

30

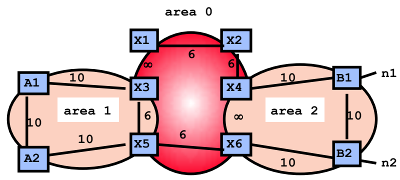
Problems - link failure



- Link A1-A2 fails, Area 1 is partitioned
  - X3 has a route to A1, X5 to A2
  - one cannot pass to X5 a packet to A1 and to X3 a packet to A2
- Solution
  - X3 and X5 will advertise only routes to connected networks (X3 advertizes A1, X5 advertizes A2)

31

Problems - partitioned backbone

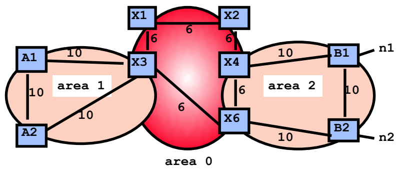


- No connectivity between areas via backbone
  - e.g. X2 to A2
- There is a route through Area 2
- Virtual link
  - X4 and X6 configure a virtual link through Area 2
  - virtual link entered into the database, metric = sum of links

32

Stub area

- Many networks are connected only via one router
- Stub area
  - all external networks aggregated into `default` route
  - e.g. route to `n1, n2` or any other network in Area 0 and 2 goes through X3



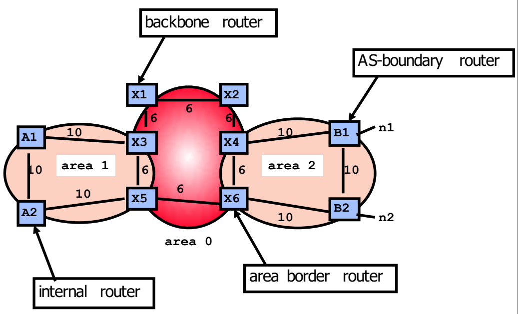
33

Classification of routers

- Internal routers
  - a router with all directly connected networks belonging to the same area
- Area border routers
  - attached to multiple areas
  - condense LSA of their attached areas for distribution to the backbone
- Backbone routers
  - a router that has an interface to the backbone area
- AS boundary routers
  - exchange routing information with routers belonging to other AS

34

Classification of routers



35

OSPF protocol

- On top of IP (protocol type = 89)
- Multicast
  - `224.0.0.5` - all routers of a link
  - `224.0.0.6` - all designated and backup routers
- Sub-protocols
  - **Hello** to identify neighbors, elect a designated and a backup router
  - **Database description** to diffuse the topology between adjacent routers
  - **Link State** to request, update, and ack the information on a link (LSA - Link State Advertisement)
- LSA
  - tagged with the router Id and checksum
  - 5 different types

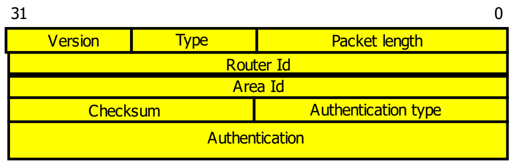
36

OSPF protocol PDUs

- OSPF protocol type = 1
  - Hello
- OSPF protocol type = 2
  - Database description
- OSPF protocol type = 3
  - Link State Request
- OSPF protocol type = 4
  - Link State Update
- OSPF protocol type = 5
  - Link State Ack

37

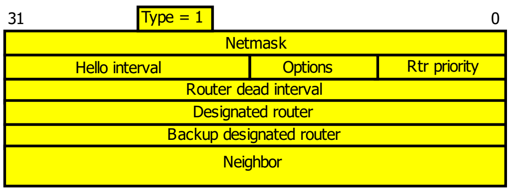
Common OSPF header



- Type
  - Hello, DB-desc, LS-request, LS-update, LS-ack
- Router Id
  - the smallest IP address of a router
- Area Id
  - chosen by administrator (4 bytes), (0 = backbone)
- Authentication
  - same as RIP v2 (password or MD5)

38

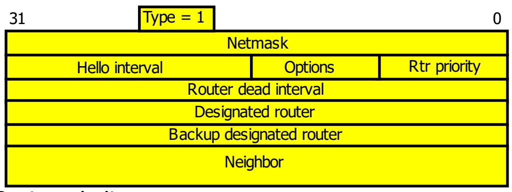
Hello protocol



- Hello interval
  - 10 sec
- Options (router capability)
  - bit E - set if the attached area capable of processing AS-external LSAs (E=0 in a stub area)
- Router dead interval
  - 40 sec

39

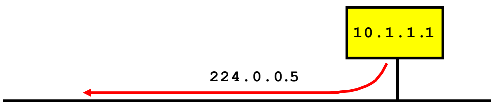
Hello protocol



- Router priority
  - used during election, the greater the better
  - if equal priority, the router with larger Id wins,
  - if 0, the router does not participate in election
- Designated/backup router
  - `0.0.0.0` if not known
- Neighbors known by the sender

40

Example

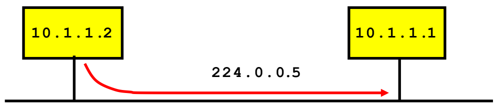



```
OSPFv2-hello 44:
area 0.0.0.1 E mask 255.255.255.0 int 10 pri 5
   dead 40 dr 10.1.1.1 nbrs
```

- `224.0.0.5` - to all routers of a link
- Router `10.1.1.1` with priority 5, prefix `10.1.1.0/24`
- Area 1, not stub area (bit E), interval 10 sec, dead interval 40, it proposes itself as designated router, no neighbors

41

Example

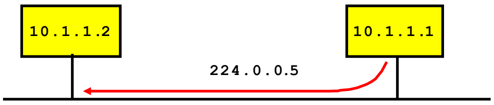



```
OSPFv2-hello 44:
area 0.0.0.1 E mask 255.255.255.0 int 10 pri 4
   dead 40 nbrs
```

- Router `10.1.1.2` with priority 4, prefix `10.1.1.0/24`
- Area 1, not stub area (bit E), interval 10 sec, dead interval 40, no neighbors

42

Example

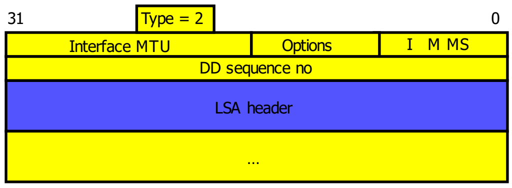



```
OSPFv2-hello 44:
area 0.0.0.1 E mask 255.255.255.0 int 10 pri 5 dead 40 dr
   10.1.1.1 nbrs 10.1.1.2
```

- Router `10.1.1.1` becomes designated
- Router `10.1.1.2` appears as a neighbor
  - bi-directional connectivity
  - can start synchronizing LS databases

43

Database Description protocol



- Interface MTU
  - max size on interface
- Options (router capability)
  - bit E - set if the attached area capable of processing AS-external LSAs
- Flags
  - I: Init bit - first description packet
  - M - More bit
  - MS - Master/Slave bit, Master if 1
- DD sequence no
  - chosen, incremented only by master
- List of link state DB pieces

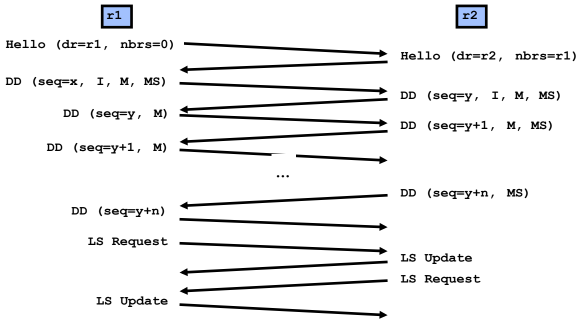
44

Database Description protocol

- Unicast packets between a router and its neighbor
- Master/slave relationship - election of the Master
  - router with larger Id becomes Master
- Master sends packets to slave (polls)
- Slave acknowledges by echoing the sequence number
- If lost packet, master retransmits
- Exchange finished when bit M=0 sent by both routers

45

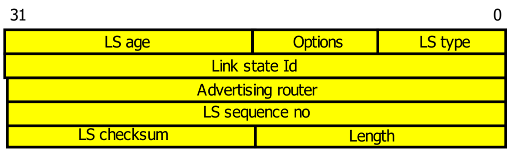
DD protocol



- `r2` is designated router (priority), `r2` has larger router Id than `r1`

46

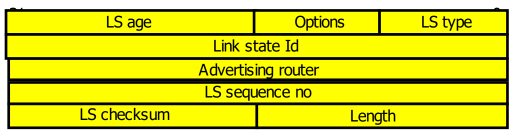
LSA header



- LS age
  - time in sec since the LSA was originated
- Options (router capability)
  - bit E - set if the attached area capable of processing AS-external LSAs
- LS type
  - type of LSA
  - see below

47

LSA header

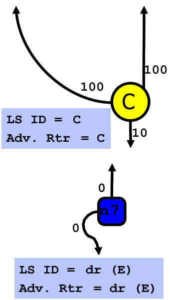


- Link state Id (depends on LSA type, see later)
  - identifies the portion of the network described by the LSA
- Advertising router
  - router that originates the LSA
- LS sequence no
  - identifies an advertisement, detects old or duplicates
  - from -N+1 to N-1, starts with `x80000001`, incremented
  - when attains `x7FFFFFFF`, LSA is aged (sent with age=MaxAge)

48

LSA types

Type	Description
1. **router-LSA**	Originated by all routers. Describes the collected states of the router's interfaces to an area. Flooded throughout a single area only.
2. **network-LSA**	Originated for broadcast and NBMA networks by the Designated Router. Contains the list of routers connected to the network (m=0). Flooded throughout a single area only.



49

LSA types

Type	Description
3. **summary-LSA**	Originated by area border routers. Describes a route to a destination outside the area, yet still inside the AS (i.e., an inter-area route). Type 3 summary-LSAs describe routes to networks. Flooded through out the LSA's associated area.
4. **summary-LSA**	Type 4 summary-LSAs describe routes to AS boundary routers.
5. **AS-external-LSA**	Originated by AS boundary routers. Describes a route to a destination in another Autonomous System. Default routes for the AS can also be described by AS-external-LSAs. Flooded through-out the AS.

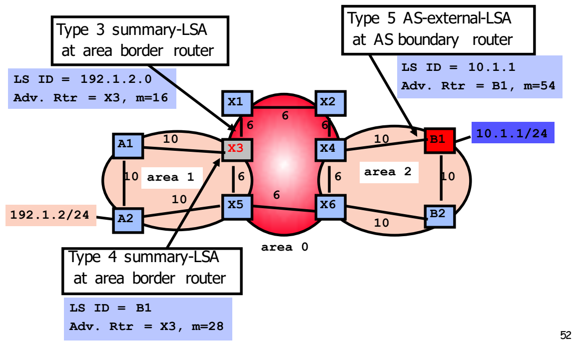
50

LS Id, Advertising Router

Type	LS Id	Adv. Router
1. **router-LSA**	The originating router's Router ID.	the same
2. **network-LSA**	The IP interface address of the network's Designated Router.	the same
3. **summary-LSA**	The destination network's IP address.	area border router
4. **summary-LSA**	The Router ID of the described AS boundary router.	area border router
5. **AS-external-LSA**	The destination network's IP address.	AS boundary router

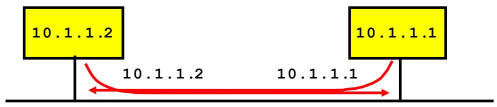
51

Types of LSA



52

Example

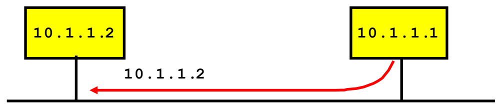



```
10.1.1.2: OSPFv2-dd 32: area 0.0.0.1 E I/M/MS S B
10.1.1.1: OSPFv2-dd 32: area 0.0.0.1 E I/M/MS S 1973
```

- 10.1.1.2 has seq. no. B
- 10.1.1.1 has seq. no. 1973
- First packet, more to follow, want to be master
- 10.1.1.2 wins (larger id)

53

Example

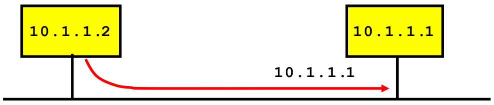



```
10.1.1.1: OSPFv2-dd 112: area 0.0.0.1 E M S B
{E S 80000002 age 3:09 rtr 10.1.1.1} TYPE 1
{E S 80000001 age 2:49 sum 10.1.2.0 abr 10.1.1.1} TYPE 3
{E S 80000003 age 2:44 sum 10.1.100.0 abr 10.1.1.1} TYPE 3
{E S 80000001 age 2:59 abr 10.1.1.1 rtr 10.1.1.1} TYPE 4
```

- Database of 10.1.1.1
  - type 1 - router-LSA
  - type 3 - summary-LSA - networks outside the area
  - type 4 - summary-LSA - route to AS boundary router

54

Example

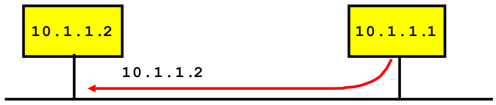



```
10.1.1.2: OSPFv2-dd 52: area 0.0.0.1 E MS S C
{S 80000002 age 5 rtr 10.1.1.2} TYPE 1
```

- Ack of the previous message
  - increment of seq. no. (from B to C)
- Database of `10.1.1.2`
  - type 1 - router-LSA
- No more information

55

Example

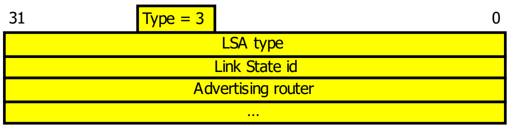



```
10.1.1.1: OSPFv2-dd 32: area 0.0.0.1 E S C
```

- `10.1.1.1` has no more information
- End of database synchronization
  - each routers knows the database of the other

56

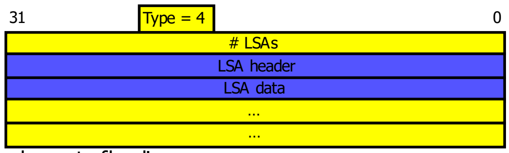
Link State Request



- Request of a database entry (described previously)
  - LSA header for DB description
- Entry identified uniquely by
  - LSA type, Link State id, Advertising router
- Current instance defined by
  - LS sequence no, LS checksum, and LS age

57

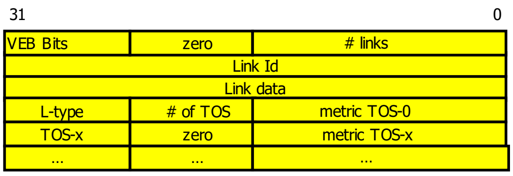
Link State Update (LSU)



- Implements flooding
  - multicast/broadcast on physical networks
  - acked with LS-Ack packets: list of LSA headers
- LSA header (same as for DB description)
  - Link State Id, Advertising router that depend on one of 5 LSA types
- LSA data
  - different formats for 5 LSA types (3 and 4 have the same format)

58

LSA data type 1 - Router-LSA



- VEB Bits
  - V - endpoint of virtual link
  - E - AS boundary router
  - B - area border router
- L-type determines Link Id and Link data
- # links
  - total collection of router's interfaces to the area
- # TOS
  - no. of different TOS (0 - no additional)
- TOS (Type of Service)
  - TOS-0 default metric

59

Router-LSA: Link Id and Link data

L-type/Nature	Link Id	Link data
1. P-to-P	Neighbor router Id	Interface index
2. Connection to transit network	Desig. Router addr.	IP router addr.
3. Connection to stub network	Subnetwork addr.	Netmask
4. Virtual link	Neighbor router Id	IP router addr.

60

TOS and metric

- TOS
  - mapping of 4 IP TOS bits to a decimal integer
  - 0 - normal service
  - 2 - minimize monetary cost
  - 4 - maximize reliability
  - 8 - maximize throughput
  - 16 - minimize delay
- Metric
  - time to send 100 Mb over the interface
  - C = 10^8/bandwidth
  - 1 if greater than 100 Mb/s
  - can be configured by administrator

61

LSA Data type 2 - Network-LSA

31 … 0

Netmask
Attached router 1
...

- Attached routers
  - router id of each of the routers attached to the network
  - Designated Router includes itself

62

LSA Data type 3/4 - Summary-LSA

31 … 0

Netmask	
zero	metric
TOS-x	metric TOS-x
...	...

- Originated by area border routers
- Describe inter-area destinations
  - Type 3, if the destination is an IP network (Link State Id is an IP network number)
  - Type 4, if the destination is an AS boundary router (Link State Id is the AS boundary router Id)
- One LSA per destination

63

LSA Data type 3/4 - Summary-LSA

31 … 0

Netmask	
zero	metric
TOS-x	metric TOS-x
...	...

- For stub areas, Type 3 can be used to describe a default route
  - Link State Id is the default destination (0.0.0.0)
  - Netmask set to 0.0.0.0
- Netmask
  - destination network's IP address mask
  - not meaningful for Type 4 (must be 0)
- TOS like for router-LSA (type 1)

64

LSA Data type 5 - AS-external-LSA

31 … 0

Netmask	
E, zero	metric
Forwarding address	
External route tag	
E, TOS-x	metric TOS-x
Forwarding address	
External route tag	

- Originated by AS boundary routers and describe destinations external to the AS (e.g. acquired from BGB)
  - Link State Id is an IP network number (can describe a default destination 0.0.0.0)

65

LSA Data type 5 - AS-external-LSA

31 … 0

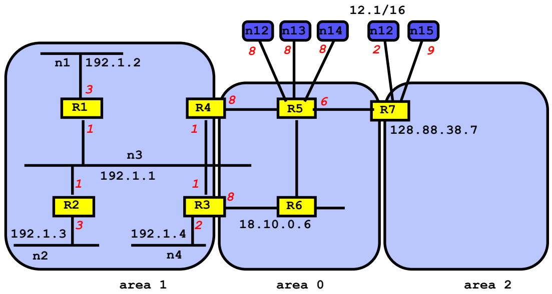
Netmask	
E, zero	metric
Forwarding address	
External route tag	
E, TOS-x	metric TOS-x
Forwarding address	
External route tag	

- Bit E - type of external metric
  - E = 0, comparable with internal metrics
  - E = 1, not comparable with internal metrics (>any internal metric)
- Forwarding address
  - data traffic will be forwarded to this address
- External route tag
  - defined by external routers (outside the scope of OSPF)

66

Example network

- Router address: router number (R3 - `192.1.4.3` and `192.1.1.3`)



67

Router-LSA

```
Router R3 for the area 1
LS age = 0, LS Type = 1
LS Id = 192.1.1.3
Adv. router = 192.1.1.3
bit E = 0, bit B = 1        ; area border router
#links = 2
   Link ID = 192.1.1.4      ;IP address of Desig. Rtr.
   Link Data = 192.1.1.3    ;R3's IP interface to net
   Type = 2                 ;connects to transit network
   # TOS metrics = 0
   metric =  1
   Link ID = 192.1.4.0      ;IP Network number
   Link Data = 0xffffff00   ;Network mask
   Type = 3                 ;connects to stub network
   # TOS metrics = 0
   metric =  2
```

68

Router-LSA

```
Router R3 for the backbone
LS age = 0, LS Type = 1
LS Id = 192.1.1.3
Adv. router = 192.1.1.3
bit E = 0, bit B = 1       ;area border router
#links = 1
   Link ID = 18.10.0.6          ;Neighbor's Router ID
   Link Data = 0.0.0.3          ;interface index (3rd)
   Type = 1                     ;connects to router
   # TOS metrics = 0
   metric =  8
```

69

Network-LSA

```
R4 on behalf of Network n3
   LS age = 0, LS type = 2,
   Link State ID = 192.1.1.4          ;IP address of Desig. Rtr.
   Adv. Router = 192.1.1.4            ;R4's Router ID
   Network Mask = 0xffffff00
          Attached Router = 192.1.1.4     ;Router ID
          Attached Router = 192.1.1.1     ;Router ID
          Attached Router = 192.1.1.2     ;Router ID
          Attached Router = 192.1.1.3     ;Router ID
```

70

Summary-LSA

```
Summary-LSA for Network n1 by Router R4 into the backbone

   LS age = 0, LS type = 3
   Link State ID = 192.1.2.0        ;n1's IP network number
   Adv. Router = 192.1.1.4          ;R4's ID
   Network Mask = 0xffffff00
   metric = 4

Summary-LSA for AS boundary router R7 by Router R4 into Area 1

   LS age = 0, LS type = 4
   Link State ID = 128.88.38.7      ;R7's ID
   Adv. Router = 192.1.1.4           ;R4's ID
   Network Mask = 0xffffff00
   metric = 14
```

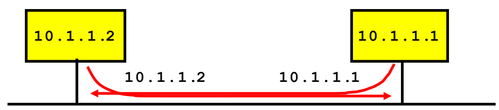
71

AS-external-LSA

```
AS-external-LSA for Network n12 by Router R7
   LS age = 0, LS type = 5
   Link State ID = 12.1.0.0              ;n12's IP network number
   Advertising Router = 128.88.38.7      ;Router R7's ID
   bit E = 1                             ;metric>than internal
   Network Mask = 0xffff0000
   metric = 2
   Forwarding address = 0.0.0.0        ;packets for external
                                     ;destination n12 should
                                    ;be forwarded to Adv.
                                    ;router - R7
```

72

Example - LS Request



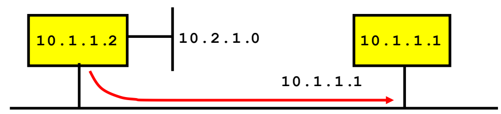

```
10.1.1.2: OSPFv2-ls_req 72: area 0.0.0.1
{rtr 10.1.1.1}
{sum 10.1.2.0 abr 10.1.1.1}
{sum 10.1.100.0 abr 10.1.1.1}
{abr 10.1.1.1 rtr 10.1.1.1}

10.1.1.1: OSPFv2-ls_req 36: area 0.0.0.1
{rtr 10.1.1.2}
```

- Routers ask for missing information

73

Example - LS Update

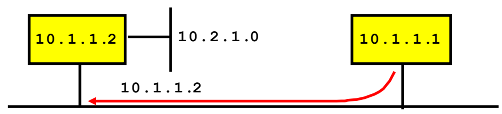



```
10.1.1.2: OSPFv2-ls_upd 76: area 0.0.0.1
{S 80000002 age 6 rtr 10.1.1.2
  {net 10.1.1.0 mask 255.255.255.0 tos 0 metric 1}
  {net 10.2.1.0 mask 255.255.255.0 tos 0 metric 1}}
```

- The contents of router-LSA (Type 1), L-Type 3 of Link Data - connection to stub network (network address and netmask)
- Age has changed (from 5 to 6)

74

Example - LS Update

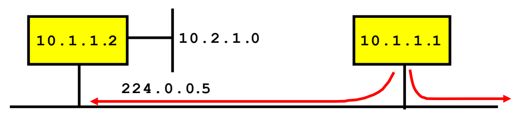



```
10.1.1.1: OSPFv2-ls_upd 148: area 0.0.0.1
{E S 80000002 age 3:10 rtr 10.1.1.1 B
  {net 10.1.1.0 mask 255.255.255.0 tos 0 metric 1}}
{E S 80000001 age 2:50 sum 10.1.2.0 abr 10.1.1.1 mask
  255.255.255.0 tos 0 metric 20}
{E S 80000003 age 2:45 sum 10.1.100.0 abr 10.1.1.1 mask
  255.255.255.0 tos 0 metric 10}
{E S 80000001 age 3:01 abr 10.1.1.1 rtr 10.1.1.1 tos 0
  metric 16777215}
```

- One link in area 1, two networks in another area
  - 10.1.2/24, 10.1.100/24

75

Example - LS Update

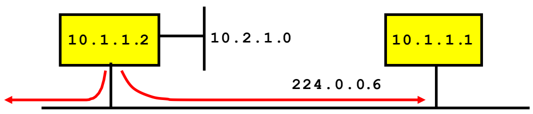



```
10.1.1.1: OSPFv2-ls_upd 56: area 0.0.0.1
{E S 80000002 age 1:00:00 abr 10.1.1.1 rtr 10.1.1.1
  tos 0 metric 16777215}
```

- 10.1.1.1 makes this information obsolete
  - MaxAge = 1:00:00 (1 hour)

76

Example - LS Update

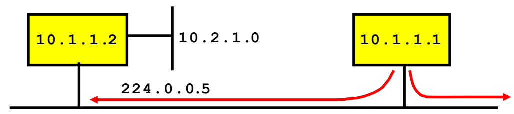



```
10.1.1.2: OSPFv2-ls_upd 76: area 0.0.0.1
{S 80000003 age 1 rtr 10.1.1.2
   {dr 10.1.1.1 if 10.1.1.2 tos 0 metric 1}
   {net 10.2.1.0 mask 255.255.255.0 tos 0 metric 1}}
```

- 10.1.1.2 sends an update
  - router-LSA (Type 1), L-Type 2 of Link Data - connection to transit network (Designated Router and IP address of the interface)

77

Example - LS Update

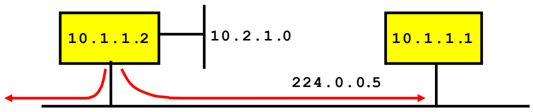



```
OSPFv2-ls_upd 64: area 0.0.0.1
{E S 80000003 age 1 rtr 10.1.1.1 B
  {dr 10.1.1.1 if 10.1.1.1 tos 0 metric 1}}
OSPFv2-ls_upd 60: area 0.0.0.1 TYPE 2
{E S 80000001 age 1 net dr 10.1.1.1 if 10.1.1.1 mask
  255.255.255.0 rtrs 10.1.1.1 10.1.1.2}
```

- 10.1.1.1 updates the information (incremented Seq. no)
- Sends a new network-LSA (Type 2)

78

Example - LS Update

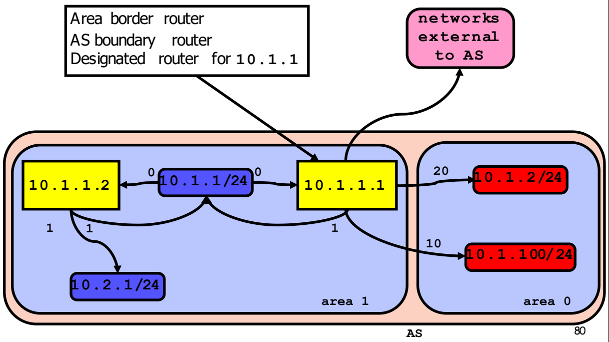



```
10.1.1.2: OSPFv2-hello 48:
area 0.0.0.1 E mask 255.255.255.0 int 10 pri 4 dead 40
dr 10.1.1.1 bdr 10.1.1.2 nbrs 10.1.1.1
```

- `10.1.1.2` sends a Hello message

79

Information gathered from OSPF



80

Convergence

- Route timeout after 1 hour
  - LS Update every 30 min.
- Detect a failure
  - 40 sec (dead interval)
- Smallest interval to recompute SPF
  - 30 sec (Dijkstra interval)
- Reconfiguration time
  - 70 sec.
- Proposals
  - Hello each 100 ms
  - SPF immediately

81

Conclusion

- OSPF vs. RIP
  - much more complex, but presents many advantages
    - no count to infinity
    - no limit on the number of hops (OSPF topologies limited by Network and Router LSA size (max 64KB) to O(5000) links)
    - less signaling traffic (LS Update every 30 min)
    - advanced metric
    - large networks - hierarchical routing
  - most of the traffic when change in topology
    - but periodic Hello messages
    - in RIP: periodic routing information traffic
  - drawback
    - difficult to configure

82