

OSPF Convergence Details

(C) Herbert Haas 2009/05/02

Introduction



- In this section we analyze all the more or less "hidden" delays that are required by the OSPF RFCs
- Our results will be,
 - ♦ That standard OSPF converges only very slowly (worst case: minutes!)
 - ♦ And that configurable and dynamic timers are required to accelerate the convergence time

(C) Herbert Haas 2009/05/02 www.perihel.at

2

Recall: Network convergence



- 1. Detect the event (=failure)**
 - ◆ Failure may happen locally or remote
- 2. Propagate the event**
 - ◆ Communicate the failure event to other devices in the network
- 3. Process the event**
 - ◆ Repair delay
 - ◆ Notified devices must calculate an alternate path
- 4. Update the routing table/FIB**

Event Detection

The hello protocol

Basic Intervals



- **Standard hello interval**
 - ◆ 10 sec for Ethernet
 - ◆ 30 sec for non-broadcast links
 - ◆ Must match between peers
- **Dead time is *always* 4 times the hello interval**
 - ◆ 40 sec on Ethernet
 - ◆ 120 sec on WAN
 - ◆ Must match between peers

Basic Intervals



- **Hello packets include a list of all neighbors for which a hello packet has been received within the dead interval**
- **Additionally 5 second hold time before SPF is triggered**
 - ◆ On LAN up to 45 seconds until convergence!

OSPF Fast Hello packets



- Supported by several vendors
- Reduced intervals
 - ◆ 1 second dead time, <1 second hello interval
 - ◆ Hello message indicates dead=1s, hello=0s
- Detects lost neighbours within 1 second
- Minimum hello interval: 50 ms
- Scaling issue with many neighbors!
 - ◆ Router might process thousands of hellos per seconds

```
!!! Reduce the default dead interval (no fast hello feature, we
!!! examine this only for completeness)
(config-if)# ip ospf dead-interval <seconds>

!!! Alternatively, enable fast hello using the "minimal" keyword
!!! which means 1 s dead time. Send 3-20 hello packets per second.
(config-if)# ip ospf dead-interval minimal hello-multiplier <3-20>
```

Event Propagation

Update, collection, and
processing

When to emit LSAs?



- **Old: Event-collection interval**
 - ♦ Router and Network LSAs are not generated immediately
 - ♦ Always after 500 ms (OSPF_LSA_DELAY_INTERVAL)
 - ♦ Reason: Collect all events during that interval
- **Old: LSA generation interval**
 - ♦ Time between generation of LSAs
 - ♦ 5 seconds by default (MinLSInterval)
- **New: Exponential backoff**
 - ♦ Prolonging network instabilities slows down the LSA generation process

```
! Exponential backoff: start-interval, hold-interval, and  
! max-interval (all in msec). The defaults are shown: The first LSA  
! is delayed by 0 ms, subsequent LSAs are exponentially delayed by  
! 2^n * 5000 ms (n=0,1,2..), and same-LSAs are no longer delayed as  
! 5000 ms (i. e. same-LSAs experience a truncated exp. backoff)  
(config-router)# timers throttle lsa 0 5000 5000
```

When to accept LSAs?



- **Receiver's per-link minimum arrival interval**
 - ♦ 1 second by default (MinLSArrival)
 - ♦ LSAs arriving at shorter intervals are discarded !
- **Note**
 - ♦ This value should be less or equal to the hold-interval of the exponential backoff process

```
! Default is 1000 msec (RFC)  
(config-router)# timers lsa arrival <msec>
```

LSA processing time



- Upon receiving an LSA the router must check if the LSA is "new"
- If yes, compare the LSA with own database for topology changes
- Duration depends on number of links
 - ♦ Typically 0.5 - 1 ms (negligible)

SPF Computation



- Computation is scheduled = delayed!
 - ♦ Partial SPF is not delayed (triggered by LSA-3,4,5,7)
 - ♦ Complete SPF is always delayed to protect CPU resources (triggered only by LSA-1 and LSA-2)
- RFC conform scheduling delays
 - ♦ spf-delay interval 5 sec
 - ♦ spf-holdtime 10 sec (upon sustained changes)
- This dramatically slows down the convergence!
- New: dynamic timers (exponential backoff)

```
! Truncated Exponential Backoff: The initial schedule is delayed by
! spf-start, subsequent schedules are delayed by 2^n * spf-hold, and
! spf-max indicates the maximum delay (i.e. the truncation). All values
! must be provided in milliseconds.
! By default, no throttling is set (i. e. the RFC timers are used)
(config-router)# timers throttle spf <spf-start> <spf-hold> <spf-max>
```

SPF Computation (cont.)



- **SPF processing delay**
 - ◆ <1ms for partial SPF
 - ◆ Up to 50-100 ms for larger areas !
 - 10 ms for 50 nodes
 - Scales nearly linearly with the number of nodes (and logarithmic with the number of links)
- **Convergence can be significantly accelerated if areas are kept small**
 - ◆ Delay = $O(E + N \log N)$ assuming efficient implementation (E...number of links/edges, N...number of nodes)

SPF Computation (cont.)



- **SPF improvements**
 - ◆ A* algorithm (extends SPF with cost estimation function to optimize performance)
 - ◆ Incremental SPF (only calculate over affected parts of the tree)
- **ISPF is not enabled by default!**

```
(config-router)# ispf
! Verify if Incremental SPF (ISPF) is enabled:
# show ip ospf
...
Incremental-SPF enabled
...
```

Flooding Delay



- New LSAs are flooded to other neighbors – but not immediately!
 - ♦ RFC demands for 33 ms pacing timer with 10% jitter
 - ♦ That is, all LSAs in the flooding queue are paces at 33 ms intervals
 - ♦ Every node adds additional 33 ms !
 - ♦ Can be configured
- Warning
 - ♦ This effectively controls the interpacket spacing
 - ♦ Reducing this value may cause high CPU and buffer utilization upon significant topology changes

```
! Default is 33 msec (RFC) but configurable range is 5-100 ms  
(config-router)# timers pacing flood <msec>
```

```
! The retransmission pacing can be configured separately;  
! the default is 66 ms, the range is 5-200 ms  
(config-router)# timers pacing retransmission <msec>
```

RIB/FIB Update

The last step



- **Finally, the routing table must be updated**
- **Significant delay!**
 - ◆ Create routes from SPF-result (recursive algorithm)
 - ◆ Update forwarding structures (radix trie, CEF)
 - ◆ Download structures to line cards
- **Future optimizations needed!**

Non-Stop Forwarding (NSF)

Restarting Routers



- **Restarting a router means that all its neighbours lose their adjacency to that router**
 - ◆ Router-LSAs change and must be re-flooded
 - ◆ SPF is triggered everywhere
- **Solution: NSF (Internet draft)**
 - ◆ Before restarting, tell neighbours NOT to drop the adjacency
- **Communicated inside OSPF Hello packets**
 - ◆ Contains EO-TLV as part of the Link-Local Signalling (LLS) data block

Active/Standby Processors



- **Typical usage: Systems with an active and standby Route Processor**
 - ◆ Control Plane is forwarded
- **Peer system is notified "keep your FIB settings"**
 - ◆ NSF supports nearly instantaneous failover

Link-Local Signalling (LLS)



- LLS simply means "exchange arbitrary data on a link"
 - ♦ To support additional OSPF mechanisms
 - ♦ Typically for capabilities exchange
- RFC 4813 describes a method to provide a vendor-specific, backward-compatible technique to perform link-local signaling
 - ♦ Opaque data transfer
- Additional data block appended to standard OSPF Hello or DBD packet
 - ♦ After authentication data block (if any)

NSF Details



- The EO-TLV contains the LR and RS bits
 - ♦ LR = LSDB Resynchronization
 - ♦ RS = Restart
- LR indicates OOB resynchronization capability
 - ♦ Part of both hello and DBD packets
- RS bit tells neighbors "preserve your adjacencies"
 - ♦ Will be acknowledged with unicast hello from neighbors (but with RS=0)