

OSPF

Ein Internet Routing Protokoll
(von Dirk Jacob)

Inhalt

- ↳ OSPF Überblick
- ↳ Link State Advertisements
- ↳ Link State Datenbank
- ↳ OSPF Pakete
- ↳ Synchronisation von Link State Datenbanken
- ↳ Routing Tabelle
- ↳ Netzwerk-Typen
- ↳ Hierarchisches Routing
- ↳ Externe Routing Informationen
- ↳ OSPF Erweiterungen
- ↳ OSPF Paket-Typen

Dirk Jacob, Seite 2

OSPF Überblick

- ↳ OSPF = Open Shortest Path First
 - ↳ benutzt Dijkstra's SPF Algorithmus
 - ↳ offener Standard, entwickelt von der IETF
 - ↳ aktuell: Version 2, RFC 2328
- ↳ Interior Gateway Protocol zum Routing innerhalb eines AS
- ↳ Link State Protokoll
- ↳ Vorteile:
 - ↳ Schneller Datenbankgleich bei Topologie-Änderungen
 - ↳ unterstützt große Netzwerke
 - ↳ geringe Anfälligkeit gegenüber fehlerhaften Routing-Informationen

Dirk Jacob, Seite 3

OSPF Features

- ☞ Konzept der Areas für hierarchische Topologien und Reduzierung der CPU- und Speicherlast auf den Routern
- ☞ unabhängig von IP-Subnetz-Klassen
- ☞ beliebige, dimensionslose Metrik
- ☞ Load Balancing bei Pfaden mit gleichen Kosten
- ☞ spezielle reservierte Multicast-Adressen reduzieren Auswirkungen auf Nicht-OSPF Geräte
- ☞ Authentifizierung
- ☞ External Route Tags
- ☞ Möglichkeit des TOS-Routing (aus RFC 2328 gestrichen, da nicht genutzt)

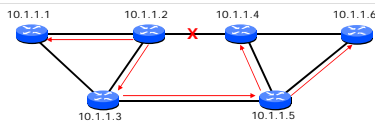
Dirk Jacob, Seite 4

OSPF Funktionsweise

- ☞ Router identifizieren beim Start ihre Nachbarn
- ☞ nicht alle angrenzenden Router werden auch zu Nachbarn (adjacents)
- ☞ Abgleich der Link State Datenbanken mit den Nachbarn
- ☞ periodische Keepalives zur Aufrechterhaltung der Nachbarschaft
- ☞ periodische Link State Updates, um die Datenbanken konsistent zu halten
- ☞ Flooding von LSA's bei Topologieänderungen

Dirk Jacob, Seite 5

Einfaches Beispiel

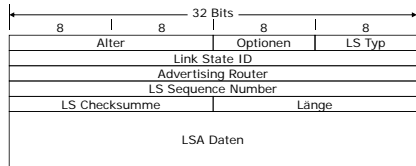


- ☞ Point-To-Point Verbindungen
- ☞ Kosten für jede Verbindung 1
- ☞ Datenbanken synchronisiert
- ☞ jeder Router kennt kürzesten Pfad zu jedem anderen Router
- ☞ 10.1.1.1 hat zwei Routen mit gleichen Kosten nach 10.1.1.6
- ☞ Verbindung zwischen 10.1.1.2 und 10.1.1.4 fällt aus
- ☞ LSA's werden über das ganze Netz verteilt
- ☞ nachdem DB Synchronisiert nur noch eine kürzeste Route

Dirk Jacob, Seite 6

Link State Advertisements

- ☛ Router verschicken LSA für jeden ihrer Links
- ☛ beschreiben damit den für sie sichtbaren Teil der Topologie
- ☛ alle LSA's zusammen ergeben die Link State Datenbank



- ☛ verschiedene Arten von LSA's zur Implementierung der einzelnen Features

Dirk Jacob, Seite 7

Identifikation von LSA's

- ☛ eindeutige Identifikation durch
 - ☛ LS Typ
 - ☛ Link State ID
 - ☛ Advertising Router
- ☛ LSA Instanzen werden identifiziert durch LS Sequence Number
- ☛ LSA Inhalt verifiziert durch LS Checksumme
 - ☛ Fletcher Checksumme über komplettes LSA, außer Alter-Feld
 - ☛ Schutz gegen „Corruption“
 - ☛ Identität von Paketen gleichen Alters mit gleicher Sequence Number

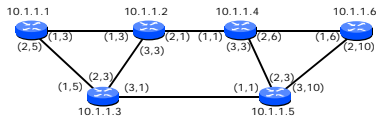
Dirk Jacob, Seite 8

LSA Header Felder

- ☛ Alter
 - ☛ Alter eines LSA in Sekunden
 - ☛ erneutes Flooding nach 30 Minuten
 - ☛ Maximales Alter 60 Minuten
- ☛ Optionen
 - ☛ Flags, die eine besondere Handhabung von LSA's anzeigen
- ☛ Länge

Dirk Jacob, Seite 9

Beispiel: Router LSA



- ☛ den Seriellen Router Interfaces wurden keine IP-Adressen zugewiesen
- ☛ keine IP Subnetze für serielle Verbindungen
- ☛ Interfaces haben Nummer und Kosten

Dirk Jacob, Seite 10

Beispiel: Router LSA (2)

32 Bits			
8	8	8	8
Alter = 0	Optionen	Typ = 1	
Link State ID = 10.1.1.1			
Advertising Router = 10.1.1.1			
Sequence Number = 0x80000006			
Checksumme = 0x9b47		Länge = 60	
00000	0 0 0 0	0x00	Anzahl Links = 3
Link ID = 10.1.1.2			
Link Daten = Interf. Index 1			
Link Typ = 1	# TOS = 0	Link-Kosten = 3	
Link ID = 10.1.1.3			
Link Daten = Interf. Index 2			
Link Typ = 1	# TOS = 0	Link-Kosten = 5	
Link ID = 10.1.1.1			
Link Daten = 255.255.255.255			
Link Typ = 3	# TOS = 0	Link-Kosten = 0	

Link Typ 1: Peer-to-peer
Link Typ 3: Stub Network

Dirk Jacob, Seite 11

Link State Datenbank

- ☛ LSA's zusammen bilden Link State Datenbank
- ☛ nach entdecken eines benachbarten Routers werden Datenbanken untereinander ausgetauscht
- ☛ danach Synchronisation durch Reliable Flooding
- ☛ LS Datenbank gibt komplette Beschreibung des Netzwerks und des Zustands aller Router
- ☛ Quelle zur Berechnung der Routing Tabelle

Dirk Jacob, Seite 12

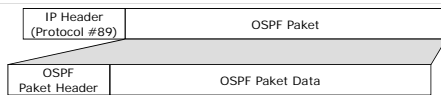
Beispiel

Link State DB des Netzwerks aus dem vorigen Beispiel:

LS-Type	Link State ID	Adv. Router	Checksum	Seq. No.	Age
Router-LSA	10.1.1.1	10.1.1.1	0x9b47	0x80000006	0
Router-LSA	10.1.1.2	10.1.1.2	0x219e	0x80000007	1618
Router-LSA	10.1.1.3	10.1.1.3	0x6b53	0x80000003	1712
Router-LSA	10.1.1.4	10.1.1.4	0xe39a	0x8000003a	20
Router-LSA	10.1.1.5	10.1.1.5	0xd2a6	0x80000038	18
Router-LSA	10.1.1.6	10.1.1.6	0x05c3	0x80000005	1680

Dirk Jacob, Seite 13

OSPF Pakete

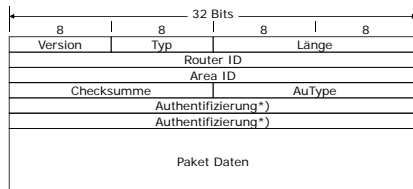


5 Pakettypen:

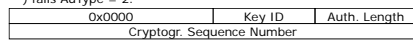
- ☛ Hello
- ☛ Database Description
- ☛ Link State Request
- ☛ Link State Update
- ☛ Link State Acknowledgement
- ☛ Übertragung über IP, Protokoll #89
- ☛ Übertragung direkt an Nachbarn oder über Multicast Adressen
- ☛ OSPF Pakete werden nur zwischen Nachbarn im Netz ausgetauscht, niemals außerhalb ihres Ursprungs-Netztes weiter gerouted (TTL=1)

Dirk Jacob, Seite 14

OSPF Header



*) falls AuType = 2:



Dirk Jacob, Seite 15

Identifizieren von Nachbarn

- ☛ Nachbarn sind die Router, mit denen ein Router direkt Informationen austauscht
- ☛ periodisches Senden von Hello-Paketen über alle Interfaces
 - ☛ Hello Protokoll
- ☛ auch zum erkennen fehlerhafter Links
- ☛ sicherstellen mehrerer Voraussetzungen
 - ☛ bidirektionale Kommunikation
 - ☛ Übereinstimmung verschiedener Parameter
 - ☛ erkennen und aushandeln verschiedener Erweiterungen
 - ☛ in bestimmten Netzwerktypen zusätzliche Funktion

Dirk Jacob, Seite 16

Datenbank-Synchronisation

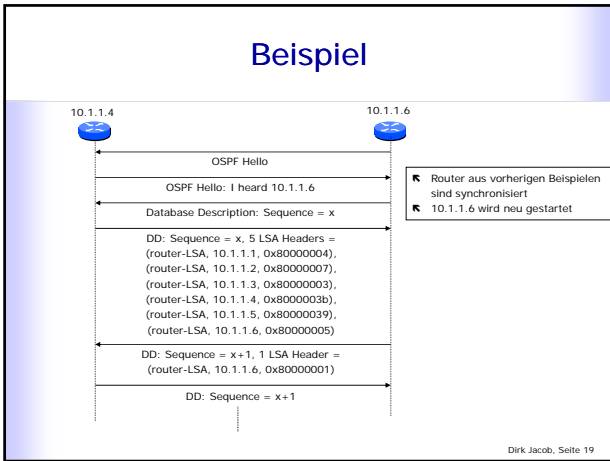
- ☛ zentraler Punkt, da alle Router identische Datenbanken haben müssen
- ☛ 2 Arten von Synchronisation:
 - ☛ Initiale Synchronisation, wenn das Nachbarschafts-Verhältnis aufgebaut wird
 - ☛ Kontinuierliche Synchronisation, um die Konsistenz der Datenbanken zu gewährleisten (flooding)

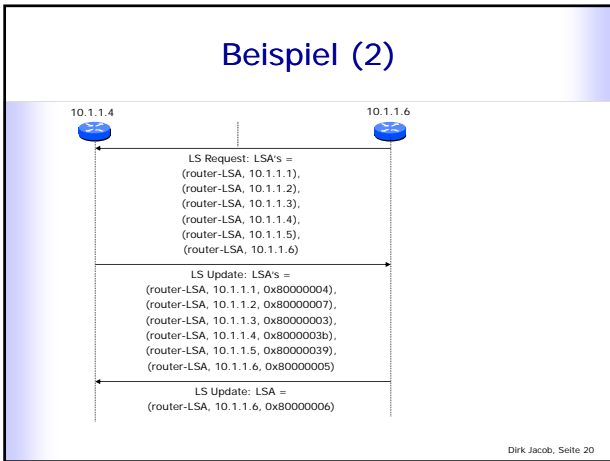
Dirk Jacob, Seite 17

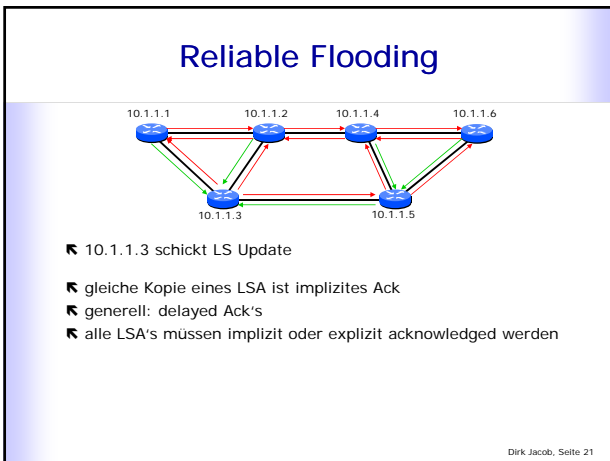
Initiale Synchronisation

- ☛ Explizite Übertragung der gesamten Datenbank, wenn Nachbarschaftsverhältnis aufgebaut wird
- ☛ abhängig vom Netzwerk-Typ (später)
- ☛ Synchronisation, wenn bi-direktionale Kommunikation besteht
- ☛ Senden aller LS Header aus der eigenen LSDB an den Nachbarn
 - ☛ OSPF Database Description Pakete
- ☛ Flooding aller zukünftigen LSA's über die Verbindung
- ☛ immer nur ein DD Paket zur selben Zeit, Senden des nächsten Pakets erst nach Acknowledge durch Senden eines entsprechenden DD Pakets des Nachbarn
- ☛ bestimmen, welche LSA's in der eigenen DB fehlen und Anfordern dieser Pakete mit Link State Request Paketen
- ☛ Nachbar „flooded“ diese in Link State Update Paketen
- ☛ danach bereit für Daten-Verkehr über die Verbindung (fully adjacent)

Dirk Jacob, Seite 18







Robustheit des Flooding

- ☞ Flooding über alle Links nicht nur über Spanning Tree
 - ☞ einzelne gestörte Links stören nicht die Synchronisation
- ☞ LSA-Refreshes nach 30 Minuten, um Fehler in Router-Datenbanken zu korrigieren
- ☞ Übertragungsfehler werden durch Checksumme erkannt
 - ☞ solche LSA's nicht acknowledge
- ☞ LS Alterung zwingt zum regelmäßigen Austausch der LSA's in den Datenbanken
 - ☞ nach max. 1h ist auf jeden Fall aktuelle LSA in der Datenbank
- ☞ LSA's können höchstens alle 5s aktualisiert werden
- ☞ keine Annahme von LSA's die weniger als 1s vorher bereits angenommen wurden

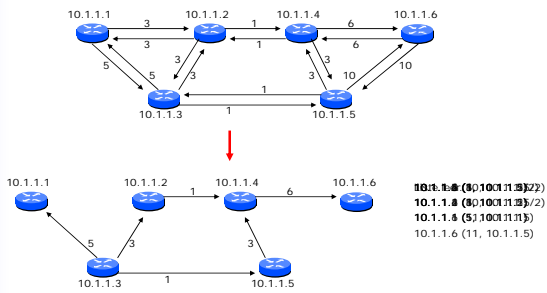
Dirk Jacob, Seite 22

Berechnung der Routing Tabelle

- ☞ Link State Datenbank ist ein gerichteter Graph mit Kosten für jeden Link
- ☞ Dijkstra's SPF Algorithmus zur Berechnung der kürzesten Pfade zu allen Zielen gleichzeitig:
 - ☞ addiere Router zum Shortest-Path-Tree
 - ☞ füge alle Nachbarn zur Candidate List hinzu
 - ☞ addiere den Router mit den kleinsten Kosten zum Tree
 - ☞ füge die Nachbarn dieses Routers an die Candidate List an
 - ☞ wenn noch nicht vorhanden
 - ☞ wenn Kosten niedriger als im bisherigen Listeneintrag
 - ☞ wiederhole, bis die Candidate List leer ist
- ☞ Laufzeit $O(l * \log(n))$

Dirk Jacob, Seite 23

Beispiel



Dirk Jacob, Seite 24

Netzwerk-Typen

- ☛ bisher nur einfache Point-to-Point Verbindungen
- ☛ viele andere Netzwerk-Technologien (Ethernet, Token Ring,...)
- ☛ spezifische Anforderungen an OSPF

- ☛ Unterschiede in:
 - ☛ Aufbau und Aufrechterhaltung von Nachbarschafts-Verhältnissen
 - ☛ Datenbank Synchronisierung
 - ☛ Repräsentation in der Link State Datenbank

- ☛ Point-To-Point
- ☛ Broadcast
- ☛ Nonbroadcast Multiaccess
- ☛ Point-to-Multipoint

Dirk Jacob, Seite 25

IP Subnetze

- ☛ IP Subnetze bestehen aus Netzwerk-Adresse und Subnetz Maske
- ☛ IP-Routing auf Subnetz Ebene, nicht auf Host Ebene

- ☛ Verkehr zwischen IP Subnetzen geht über Router
- ☛ Pakete können zwischen Hosts/Routern im gleichen Subnetz verschickt werden
- ☛ Datenaustausch zwischen Routern nur, wenn sie an ein gemeinsames Subnetz angeschlossen sind

- ☛ OSPF Hello's werden nur akzeptiert, wenn:
 - ☛ gleiche Subnet-Mask
 - ☛ beide Interfaces gehören zum gleichen IP Subnetz

Dirk Jacob, Seite 26

Broadcast Subnets

- ☛ Netzwerk, in dem ein Paket von allen anderen im Netz empfangen wird (z.B. Ethernet)

- Nachbarschaftsverhältnisse:
 - ☛ OSPF Router gehören der Multicast Gruppe AllSPFRouters (224.0.0.5) an
 - ☛ Hello Pakete werden an diese Adresse geschickt
 - ☛ ermöglicht automatisches Erkennen von Nachbarn, ohne deren IP Adressen kennen zu müssen
 - ☛ Reduzieren von Hello-Paketen, da alle angrenzenden Router mit einem Hello Paket erreicht werden
 - ☛ wenn Multicast nicht unterstützt wird, werden Hello's als Broadcast verschickt (also auch an Nicht-OSPF Knoten)

Dirk Jacob, Seite 27

Broadcast Subnets (2)

Datenbank-Synchronisation:

- ☛ viel Last durch unnötige Last, wenn jeder Router LSA's an jeden anderen Router verschickt
- ☛ Konzept des Designated Routers, der das Netzwerk repräsentiert
- ☛ Router bilden nur Nachbarschaftsverhältnis mit dem DR
- ☛ Backup DR steht bereit, falls der DR ausfällt
- ☛ Designated Routers haben Multicast Adresse 224.0.0.6 (AllDRouters)

Dirk Jacob, Seite 28

Wahl des DR und BDR

- ☛ Router wird aktiv und bestimmt alle Nachbarn
- ☛ wenn aktiver DR und BDR, diese übernehmen
- ☛ kein BDR, dann wird der Router mit der höchsten Priorität BDR
- ☛ bei gleicher Priorität, Router mit der größten ID
- ☛ kein DR, dann BDR zum DR wählen und Prozedur für BDR wiederholen

- ☛ erste beide zur Wahl stehende Router werden DR und BDR
- ☛ nur Router mit Priorität > 0 mit bidirektionaler Kommunikation stehen zur Wahl
- ☛ steht kein Router zur Wahl, dann kein DR und BDR
 - ☛ keine Adjacencies
- ☛ nur ein Router, dann DR und kein BDR

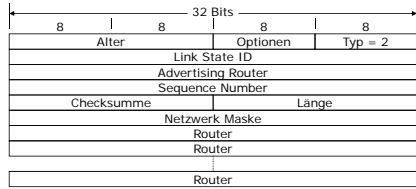
Dirk Jacob, Seite 29

Network LSA's

- ☛ Network LSA repräsentiert das gesamte Broadcast Subnetz
- ☛ Router LSA's haben einen Link zur Network LSA
- ☛ Reduktion der Links von $n*(n-1)$ auf $n*2$
- ☛ DR verbreitet Network LSA
- ☛ Link State ID = IP-Adresse des DR

Dirk Jacob, Seite 30

Network LSA Pakete



Dirk Jacob, Seite 31

NBMA Subnets

- ☞ NBMA Subnetze erlauben beliebigen Routern direkt miteinander zu kommunizieren, bieten aber kein Broadcast (z.B. ATM,...)
- ☞ NBMA Netze können als Point-to-Multipoint Netze modelliert werden
- ☞ aber auch ähnliche Mechanismen wie bei Broadcast Netzen
- ☞ kein Broadcast möglich, also müssen benachbarte Router „von Hand“ konfiguriert werden
- ☞ nur in Routern mit Priorität >0, da nur diese DR werden können und alle anderen kennen müssen
- ☞ diese wählen dann DR und BDR
- ☞ DR schickt Hello's an die übrigen Router, die dann per Unicast antworten können
- ☞ Datenbank Synchronisation wie bei Broadcast Netzen
- ☞ Repräsentation in der Datenbank auch hier mit Network LSA

Dirk Jacob, Seite 32

Point-to-Multipoint Subnets

- ☞ überall Anwendbar, wo auch NBMA Subnetze anwendbar sind (ATM,...)
- ☞ normalerweise verbindungs-gebundene Netzwerke
- ☞ nicht alle Router müssen paarweise kommunizieren können
- ☞ einzelne Verbindungen werden dabei wie Point-to-Point Links behandelt
- ☞ Aufbau von Nachbarschaft wie bei Point-to-Point Links
- ☞ Nachbarschaft mit allen Routern zu denen bidirektionale Kommunikation besteht
- ☞ Datenbank-Synchronisation wie bei Point-to-Point Links
- ☞ in der Router LSA:
 - ☞ eine point-to-point Verbindung zu jedem Nachbarn
 - ☞ ein Stub Network für seine eigene IP-Adresse

Dirk Jacob, Seite 33

Hierarchisches Routing

- ☛ große AS führen zu großen Link State Datenbanken
 - ☛ hohe Anforderungen an Speicher und Prozessoren der Router
 - ☛ hohe Bandbreiten beim Austausch von Routing Informationen
- ☛ Partitionieren des Netzes in einzelne Teile
- ☛ Hierarchie
- ☛ oberste Ebene: Routing ist flach, alle Router kennen alle Netzwerksegmente
- ☛ aber: nur rudimentäre Kenntnisse über andere Partitionen
- ☛ weiterreichen von Paketen an die jeweils höhere Stufe, wenn Zieladresse nicht bekannt
- ☛ möglicherweise wird der Weg von Quelle zu Ziel länger

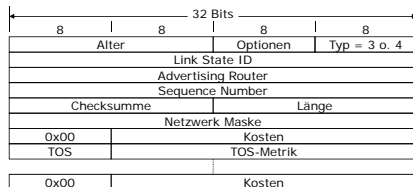
Dirk Jacob, Seite 34

OSPF Areas

- ☛ in OSPF zwei Hierarchiestufen
 - ☛ Areas
- ☛ jede Area hat eigene Link State Datenbank
- ☛ Routing innerhalb der Area ist flach
- ☛ Details über die Topologie der Area sind außerhalb der Area nicht bekannt
- ☛ Router-LSA's und Network-LSA's werden nicht über Area Grenzen hinweg geflooded
- ☛ Router, die zu mehreren Areas gehören heißen Area Border Routers
- ☛ ABR's tauschen Informationen über Areas in Network Summary LSA's aus
- ☛ in Summary LSA's werden Adressen aus der Area zu einem „Longest Prefix“ zusammengefaßt
- ☛ Link State ID ist die Prefix-Adresse (z.B. 10.2.0.0)
- ☛ Kosten = Kosten vom ABR zum „teuersten“ Ziel-Netz

Dirk Jacob, Seite 35

Network Summary LSA



Dirk Jacob, Seite 36

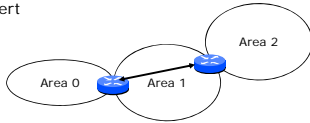
Area Organisation

- ☛ alle Areas müssen mit der Backbone Area (Area ID 0) verbunden sein
- ☛ ABR's schicken Summary LSA's ihrer Areas in den Backbone
- ☛ für jedes Ziel wird die beste Summary LSA in die eigene Area geschickt
- ☛ nicht unbedingt physikalische Links an Backbone erforderlich
 - ☛ virtuelle Links

Dirk Jacob, Seite 37

Virtuelle Links

- ☛ logischer Anschluß von Areas an den Backbone
- ☛ Summary LSA's werden durch einen Tunnel durch eine Area transportiert



- ☛ Datenpakete werden nicht über Backbone gerouted, falls dieser nicht auf dem kürzesten Weg liegt
- ☛ virtuelle Links vergleichbar mit Punkt-zu-Punkt Links

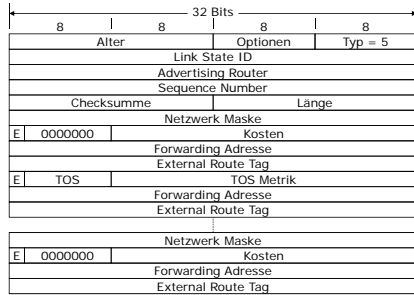
Dirk Jacob, Seite 38

Externe Routing Information

- ☛ Internet besteht aus Autonomen Systemen
 - ☛ OSPF innerhalb von AS
- ☛ Grenze einer OSPF Domain zur Außenwelt ist AS Boundary Router
- ☛ Routing Information aus anderen Routing Protokollen kann von ASBR's mit eingebracht werden um den besten Weg aus dem eigenen Netz heraus zu finden
- ☛ AS External LSA's
- ☛ zwei Arten von „Externem Traffic“
 - ☛ Intra-Area
 - ☛ Inter-Area
 - ☛ Type 1 External (Kosten bis zum ASBR + Externe Kosten)
 - ☛ Type 2 External (nur externe Kosten)

Dirk Jacob, Seite 39

AS External LSA



Dirk Jacob, Seite 40

Areas und Externe Routen

- ☛ AS External LSA's werden über Area Grenzen geflooded
- ☛ zusätzlich werden ASBR-Summary LSA's von den ABR's in ihren Areas verteilt

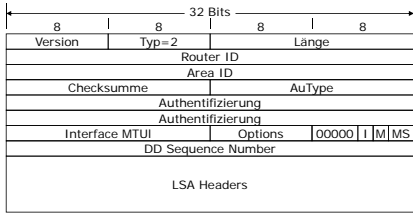
Dirk Jacob, Seite 41

Area Typen

- ☛ Verschiedene Area Typen, um Link State Datenbanken weiter verkleinern zu können
- ☛ Stub Areas
 - ☛ AS External LSA's werden dorthin nicht übertragen
 - ☛ Routing zu externen Zielen über Default Routen
 - ☛ keine ASBR's
 - ☛ keine virtuellen Links
 - ☛ Summary LSA's optional
- ☛ NSSA's
 - ☛ Erweiterung zu Stub Areas
 - ☛ kleine Anzahl externer Rouen erlaubt
 - ☛ NSSA External LSA's
 - ☛ werden an der NSSA-Grenze in AS-External LSA's übersetzt
 - ☛ NSSA-Grenze ist „Einbahnstraße“ für Externe Routing-Information

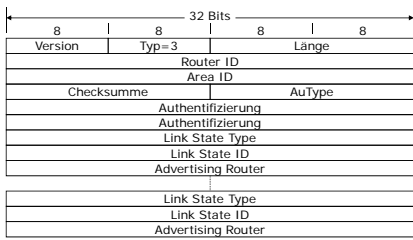
Dirk Jacob, Seite 42

Database Description Paket



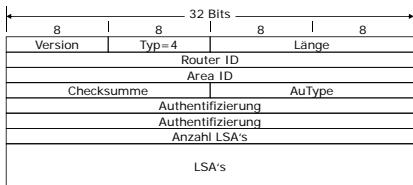
Dirk Jacob, Seite 46

Link State Request



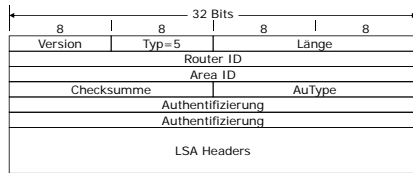
Dirk Jacob, Seite 47

Link State Update



Dirk Jacob, Seite 48

Link State Acknowledgement



Dirk Jacob, Seite 49

ENDE.

Dirk Jacob, Seite 50
