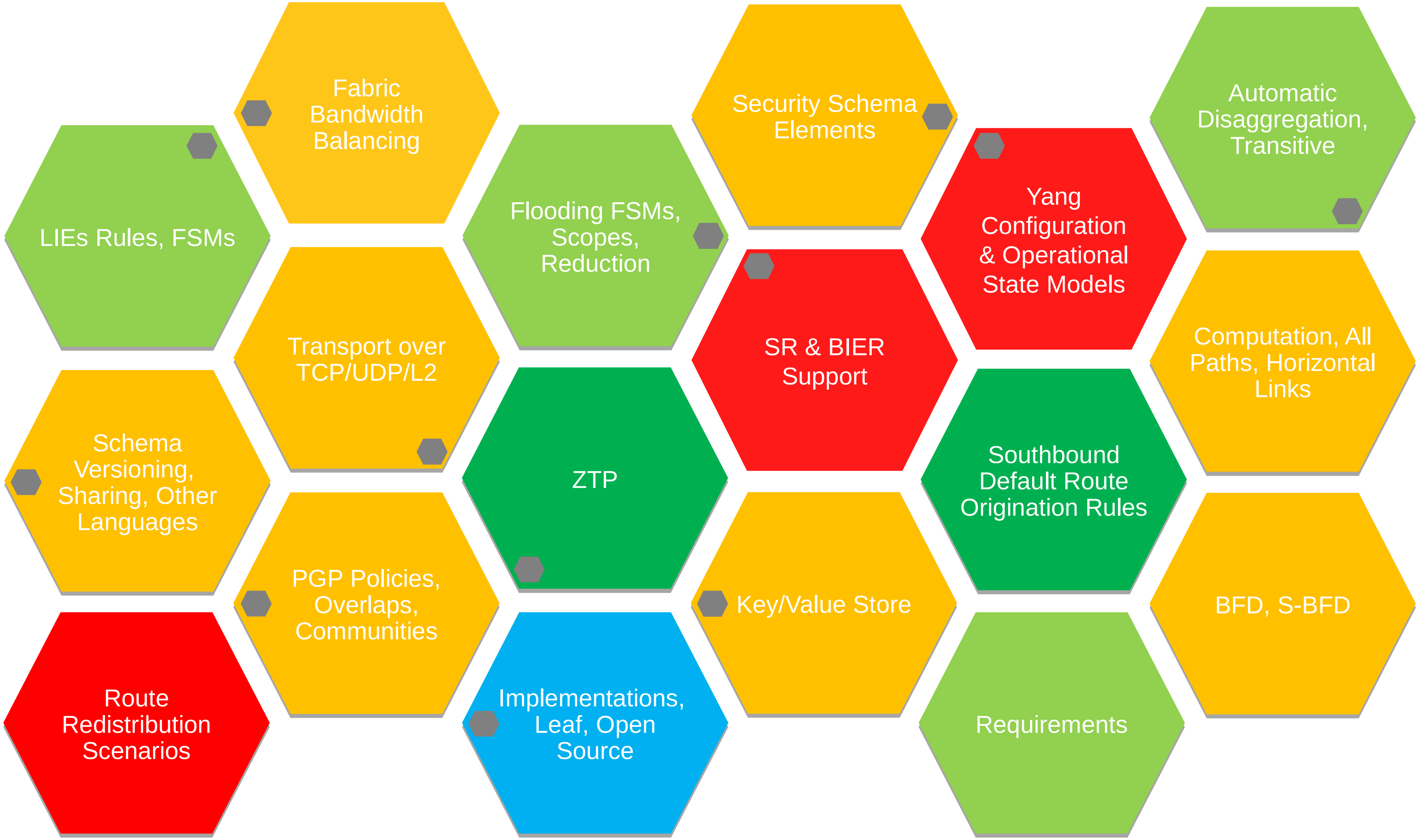
RIFT Specification

Problems Understood, Spec’ed & Needing Bricking Up vs. Problems in Fluid State vs. Open Parts and Problems

Tony Przygienda (Juniper)
Alankar Sharma (ComCast)
Alia Atlas (Juniper)
John Drake (Juniper)

Parts and Problems: Traffic Light Colored Mind Map



LIEs Rules, FSMs

Schema Versioning, Sharing, Other Languages

Route Redistribution Scenarios

Fabric Bandwidth Balancing

Transport over TCP/UDP/L2

PGP Policies, Overlaps, Communities

Flooding FSMs, Scopes, Reduction

ZTP

Implementations, Leaf, Open Source

Security Schema Elements

SR & BIER Support

Key/Value Store

Yang Configuration & Operational State Models

Southbound Default Route Origination Rules

Requirements

Automatic Disaggregation, Transitive

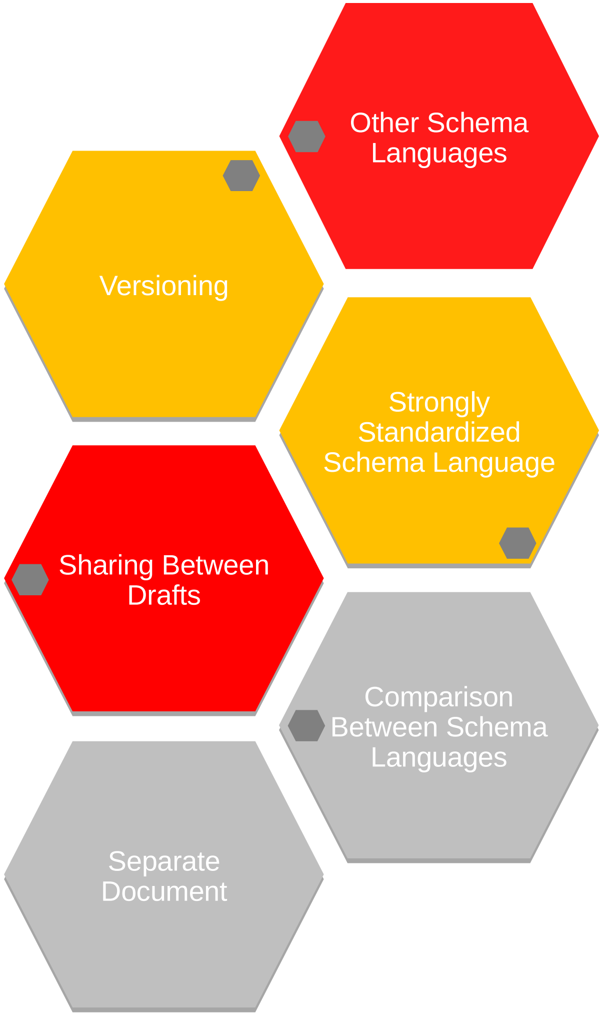
Computation, All Paths, Horizontal Links

BFD, S-BFD

Requirements

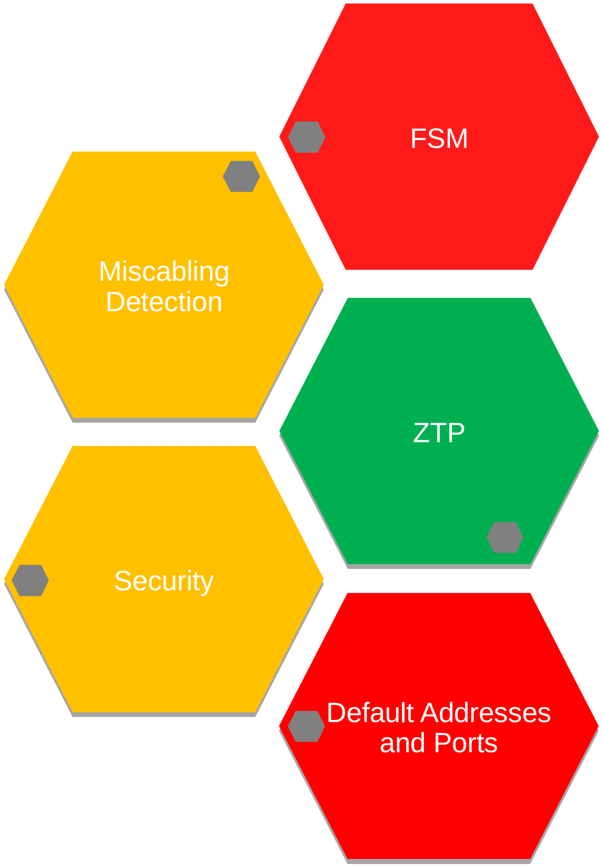
- Should Be Likely Split into Separate Document

Schema


Other Schema Languages
Versioning
Strongly Standardized Schema Language
Sharing Between Drafts
Comparison Between Schema Languages
Separate Document

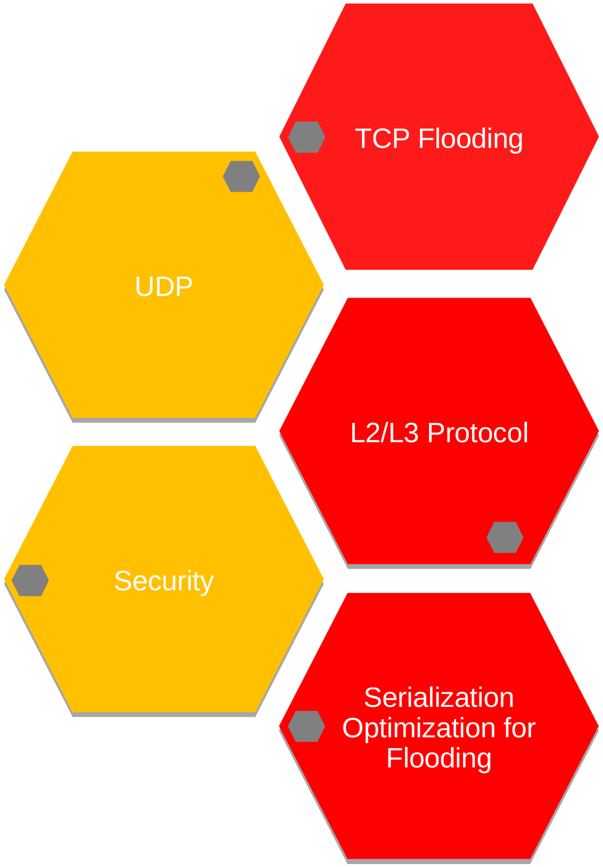
- Versioning
- Sharing Between Drafts
  - Optional Elements Without Defaults
  - Capabilities with “Mount-points”
  - Periodic Schema Reconciliation
  - Schema as Separate Document (?)
- Other Schema Languages
  - Encoding Variants (binary, compressed)
- Strongly Standardized IDL
  - IETF Reference to Thrift

LIEs



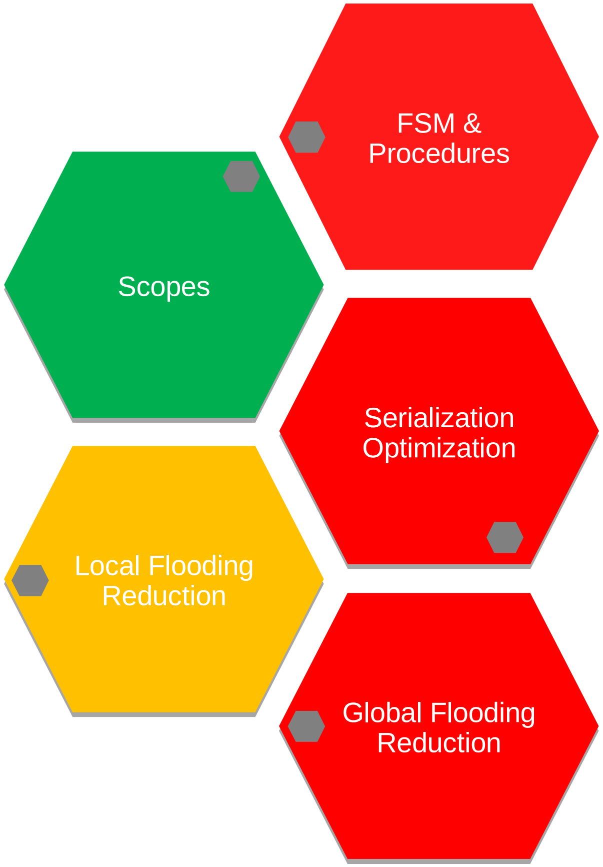
- FSM Needed
- ZTP Negotiation
- Security Hooks
- Default Addresses and Ports

Transport



- UDP
- No L2 SNAP, no L3 Protocol (?)
- No TCP Support (?)
- Transport Security (?)

TIEs & Flooding

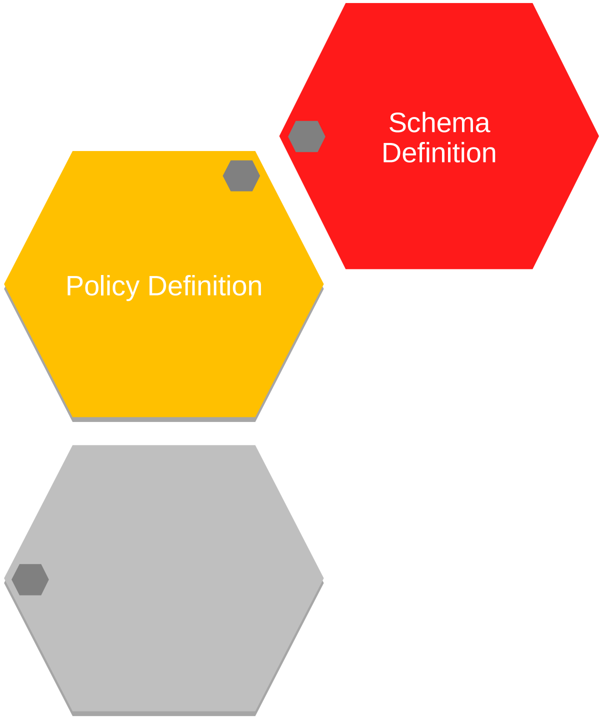


- Scopes
- FSM & Procedures
- Serialization Optimizations

```
+--------+-------------+----------+-------------+--------+-----------------+
| UDP    | Security    | TIE      | Security    | LIE    | Serialized      |
| Header | Fingerprint | Lifetime | Fingerprint | Nonces | RIFT       .... |
|        | Length      |          | (e.g. SHA)  |        | Object          |
+--------+-------------+----------+-------------+--------+-----------------+
```

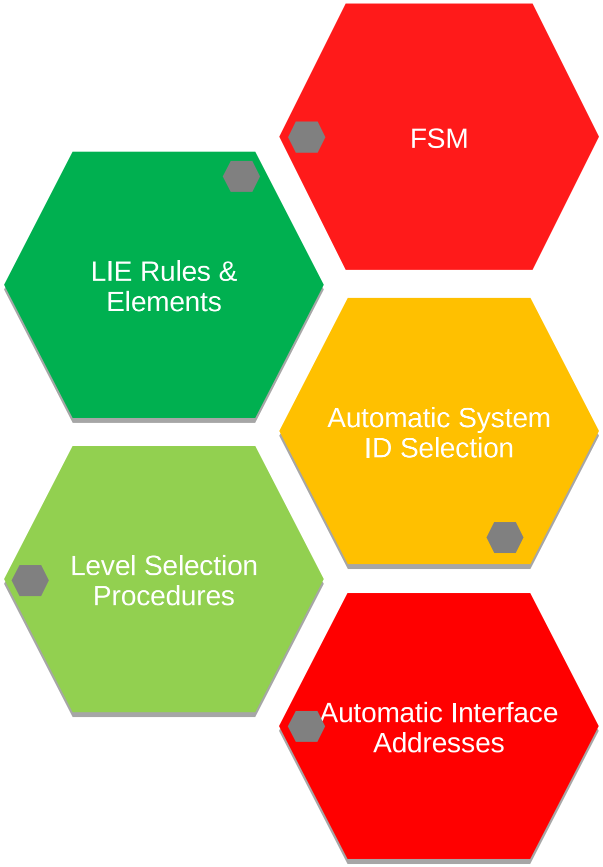
- Local Flooding Reduction
- Global Flooding Reduction

PGP



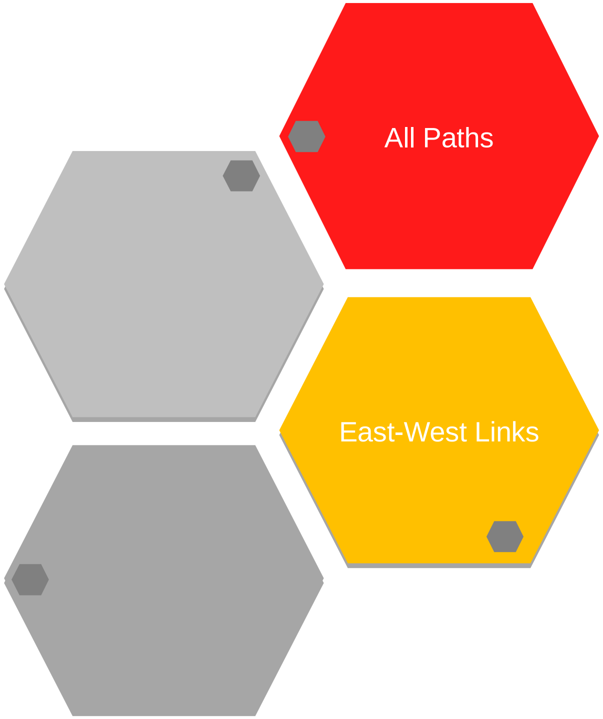
- Schema Definition
- Policy Definition

ZTP


FSM
LIE Rules & Elements
Automatic System ID Selection
Level Selection Procedures
Automatic Interface Addresses

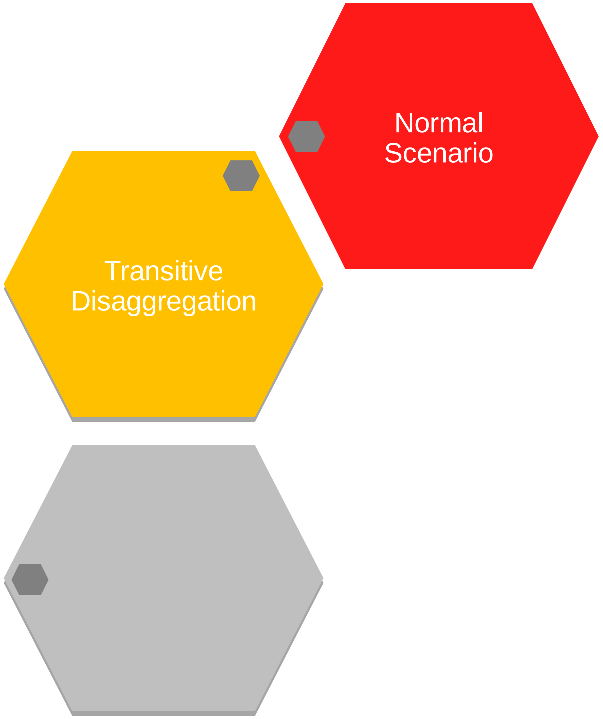
- FSM
- LIE Extensions
- Level Selection Procedures
- Automatic System ID Selection
- Automatic Interface Addresses (?)

Reachability Computation



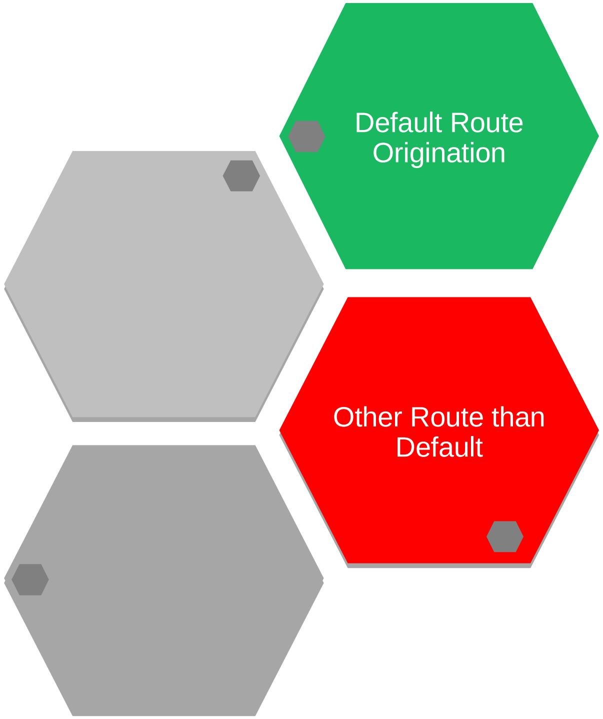
- All Paths Computation
- East-West Link

Automatic Disaggregation



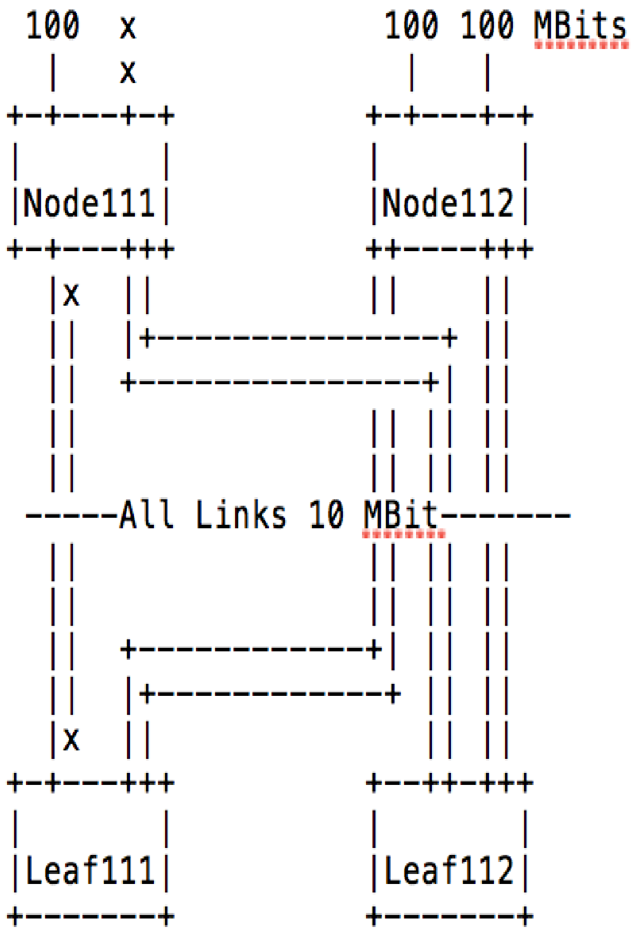
- Normal Scenario
- Transitive Disaggregation on Super-Spine Partitioning

Southbound Default Route Origination



- Default Route Origination Southbound
- Using Non-Default Route

Update: Northbound Bandwidth Balancing



RIFT calculates the amount of northbound bandwidth available towards a node compared to other nodes at the same level and adjusts the default route distance accordingly to allow for the lower level to have different weights on load balancing.

BAD_N: Bandwidth Adjusted Metric to N

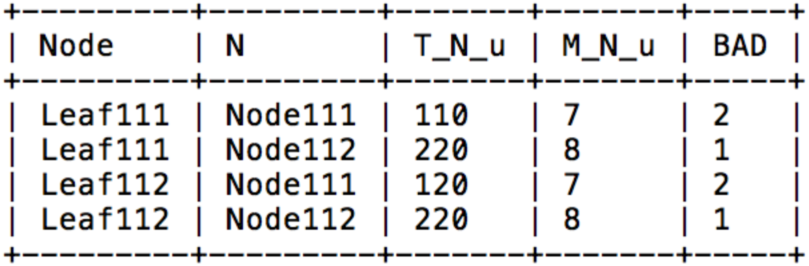
L_N_u: as sum of the bandwidth available from L to N

N_u: as sum of the uplink bandwidth available on N

T_N_u: L_N_u + N_u

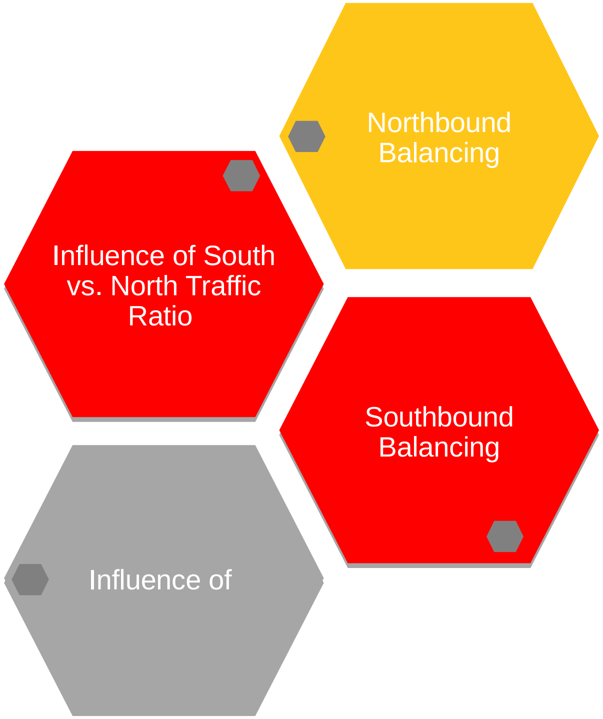
M_N_u: log_2(next_power_2(T_N_u))

BAD_N: D * (1 + maximum_of_all(M_N_u) - M_N_u)



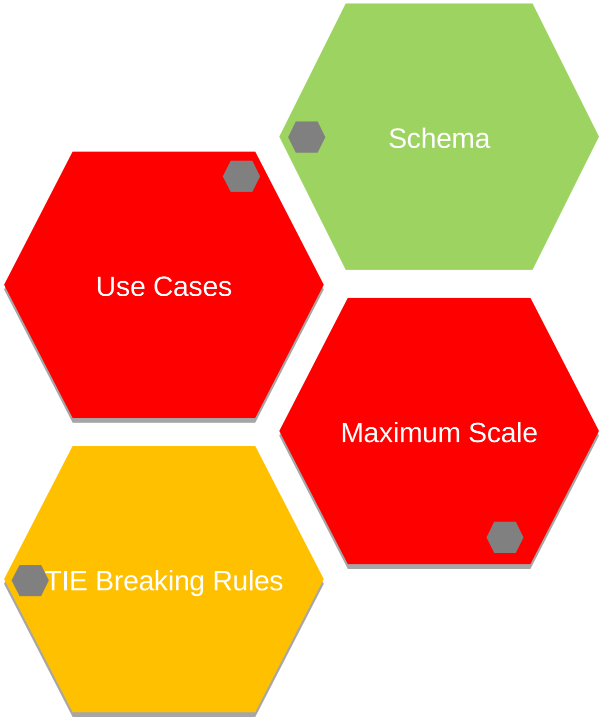
Node	N	T_N_u	M_N_u	BAD
Leaf111	Node111	110	7	2
Leaf111	Node112	220	8	1
Leaf112	Node111	120	7	2
Leaf112	Node112	220	8	1

Fabric Bandwidth Balancing



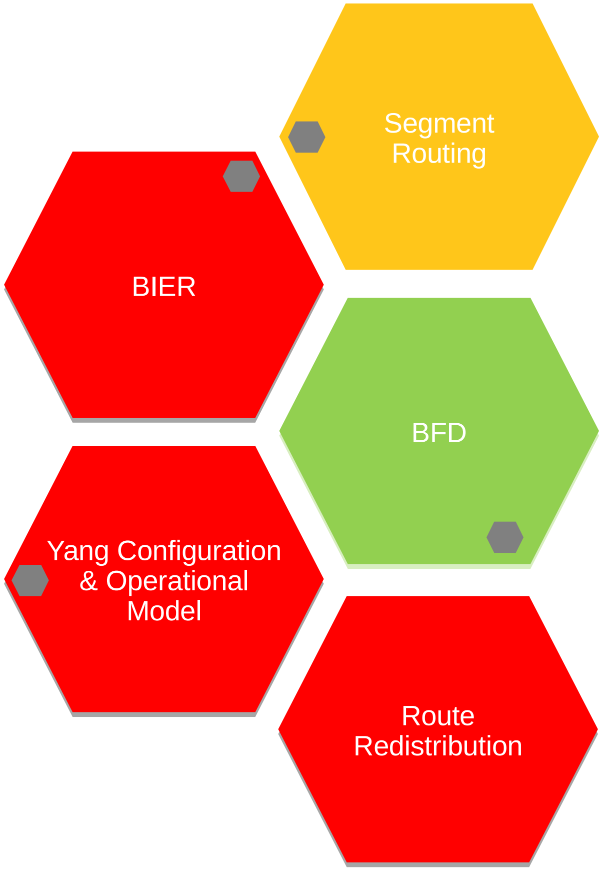
- Northbound Balancing
- Southbound Balancing
- Influence of N/S Traffic Ratio on BAD calculation

Key/Value Store



- Schema
- Use Cases
- Maximum Scale
- TIE-Breaking Rules

Other Technologies & Loose Ends



- Segment Routing
  - Node SIDs from Controller via KV or
  - Flat Flooding
- BIER
- BFD
  - S-BFD (?)
- Yang Models
- Route Redistribution

Implementations

- Public Binary Package Release in Immediate Preparation:
  - Announcement Will Be Posted to List
  - Standalone Binary Package Available in Next Few Weeks from Juniper for OSX & Linux
  - Large Parts of RIFT Implemented
  - Contains Environment to Build Fabrics Easily and Experiment With RIFT
  - Can Be Used to Test Interoperability Against It
- Open Source Implementation Encouraged
  - Leaf Version Much Simpler to Build