Routing Area Working Group   Bin Liu
Internet-Draft   ZTE Inc., ZTE Plaza
Intended status: Informational   Yantao Sun
Expires: November 24, 2019   Jing Cheng
                                                       Yichen Zhang
                                         Beijing Jiaotong University
                                                  Bhumip Khasnabish
                                                       ZTE TX Inc.
                                                       May 23, 2019

        Generic Fault-avoidance Routing Protocol for Data Center Networks
                         draft-sl-rtgwg-far-dcn-12

Abstract

   This draft proposes a generic routing method and protocol for a
   regular data center network, named as fault-avoidance routing (FAR)
   protocol. FAR protocol provides a generic routing method for all
   types of regular topology network architectures that are proposed for
   large-scale cloud-based data centers over the past few years. FAR
   protocol is well designed to fully leverage the regularity in the
   topology and compute its routing table in a simplistic manner. Fat-
   tree is taken as an example architecture to illustrate how FAR
   protocol can be applied in real operational scenarios.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

   In recent years, with the rapid development of cloud computing
   technologies, the widely deployed cloud services, such as Amazon EC2
   and Google search, bring about huge challenges to data center
   networking (DCN). Today's cloud-based data centers (DCs) require
   large-scale networks with larger internal bandwidth and smaller
   transfer delay. However, conventional networks cannot meet such
   requirements due to limitations in their network architecture. In
   order to satisfy the requirements of cloud computing services, many
   new network architectures have been proposed for data centers, such
   as Fat-tree, MatrixDCN, and BCube. These new architectures can
   support non-blocking large-scale datacenter networks with more than
   tens of thousands of physical servers.

   All of these architectures have regular topologies, which are common
   features. The regular topology refers to the network topology
   structure with obvious regularity and symmetry, which is conducive to
   automatic configuration of the network, such as the Fattree network.
   In a regular topology, each network node such as a switch or router
   can be addressed by its location and through a node's address, the
   node's connections to its neighbors in a network can be determined,
   and furthermore, the route to the node from other nodes in the
   network can be determined. So nodes can compute route entries
   without learning topology.

   This draft proposes a generic routing method and protocol, fault-
   avoidance routing (FAR) protocol, for DCNs. This method leverages
   the regularity in the topologies of data center networks to simplify
   routing learning and accelerate the query of routing tables. This
   routing method has a better fault tolerance and can be applied to any
   DCN with a regular topology.

   FAR is not a routing protocol to replace generic routing protocols
   such as OSPF and IS-IS. It cannot be used in general local networks
   whose topological structures are arbitrary, and whose scales are also
   not very large. OSPF works very well in such a network. But in a
   large-scale network with regular topology, FAR has a better
   performance. Compared with OSPF and IS-IS, FAR has shorter time of
   network convergence and lower PDU overhead. Furthermore, FAR
   requires less computing and storage resources, which let FAR routers
   to run at a lower cost of production than the generic routers.

In addition, for each type of network architecture, researchers designed a routing algorithm according to the features of its topology. Because these routing algorithms are different and lack compatibility with each other, it is very difficult to develop a routing protocol for network routers supporting multiple routing algorithms. FAR has better adaptability than these specified routing methods.

FAR consists of three components, i.e., link state learning unit, routing table building unit and routing table querying unit. In the link state learning unit, FAR exchanges link failures among routers to establish a consistent knowledge of the entire network. In this stage, the regularity in topology is exploited to infer failed links and routers. In the routing table building unit, FAR builds up two routing tables, i.e., a basic routing table (BRT) and a negative routing table (NRT), for each router according to the network topology and link states. In the last component, routers forward incoming packets by looking up the two routing tables. The matched entries in BRT minus the matched entries in NRT are the final route entries to be used to forward an incoming packet.

The remainder of this draft is organized as follows. The problem to be addressed by FAR is described in Section 3. The framework of FAR routing protocol is described in Section 4. Section 5 and 6 introduce FAR's data format FAR and modules in detail. Section 7 describe how FAR works by finite state machine (FSM). In Section 8, we discussed how FAR works with variable network architectures. Section 9 takes Fat-tree network as an example to illuminate how FAR works.

1.1. Acronyms & Definitions

DCN - Data Center Network

FAR - Fault-Avoidance Routing

BRT - Basic Routing Table

NRT - Negative Routing Table

NDT - Neighbor Devices Table

ADT - All Devices Table

LFT - Link Failure Table

DA - Device Announcement

   LFA - Link Failure Announcement

   DLR - Device and Link Request

   IP - Internet Protocol

   UDP - User Datagram Protocol

   VM - Virtual Machine

2. Conventions used in this document

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL
   NOT","SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in
   this document are to be interpreted as described in RFC-2119
   [RFC2119]. In this document, these words will appear with that
   interpretation only when in ALL CAPS. Lower case uses of these words
   are not to be interpreted as carrying RFC-2119 significance.

3. Problem Statement

   The problem to be addressed by FAR as proposed in this draft is
   described in this section. The expansion of Cloud data center
   networks has brought significant challenges to the existing routing
   technologies. FAR mainly solves a series of routing problems faced
   by large-scale data center networks.

3.1. The Impact of Large-scale Networks on Route Calculation

   In a large-scale cloud data center network, there may be thousands of
   routers. Running OSPF and other routing protocols in such network
   will encounter these two challenges: a) Network convergence time
   would be too long, which will cause a longer time to elapse for
   creating and updating the routes. The response time to network
   failures may be excessively long; b) a large number of routing
   protocol packets need to be sent. The routing information consumes
   too much network bandwidth and CPU resources, which easily leads to
   packet loss and makes the problem (a) more prominent.

   In order to solve these problems, a common practice is to partition a
   large network into some small areas, where the route calculation runs
   independently within different areas. However, nowadays the cloud
   data centers typically require very large internal bandwidth. To
   meet this requirement, a large number of parallel equivalent links
   are deployed in a network, such as a Fat-tree network. Partitioning
   such a network will affect the utilization of routing algorithm on
   equivalent multi-path and reduce internal network bandwidth
   requirements.

In the FAR routing calculation process, a Basic Routing Table (BRT)
is built on local network topology leveraging the regularity of the
network topologies. In addition to BRT, FAR also builds a Negative
Routing Table (NRT). FAR gradually builds NRT in the process of
learning network link failure information, which does not require
learning a complete network fault information. FAR does not need to
wait for the completion of the network convergence in the process of
building these two tables. Therefore, it avoids the problem of
excessive network convergence overheads in the route calculation
process. In addition, FAR only needs to exchange a small amount of
link change information between routers, and hence consumes less
network bandwidth.

3.2. Dilemma of conventional routing methods in a large-scale network with giant number nodes of routers

There are many real world scenario where tens of thousands of
nodes(or much more nodes) need to be deployed in a flat area, such as
infiniband routing and switching system, high-performance computer
network, and many IDC networks in China. The similar problems have
been existed long ago. People have solved the problems through
similar solutions, such as the traditional regular topology-based
RFC3619 protocol, the routing protocols of infiniband routing and
switching system, and high-performance computer network routing
protocol.
Infiniband defines a switch-based network to interconnect processing
nodes and the I/O nodes. Infiniband can support very large scale
networks, use the regularity in topology to simplify its routing
algorithm, which is just the same to what we do in FAR.

Why OSPF and other conventional routing methods do not work well in a
large-scale network with giant number nodes of routers?

As everyone knows, the OSPF protocol uses multiple databases, more
topological exchange information (as seen in the following example)
and complicated algorithm. It requires routers to consume more
memory and CPU processing capability. But the processing rate of CPU
on the protocol message per second is very limited. When the network
expands, CPU will quickly approach its processing limits, and at this
time OSPF can not continue to expand the scale of the management.
The SPF algorithm itself does not thoroughly solve these problems.

On the contrary, FAR does not have the convergence time delay and the
additional CPU overheads, which SPF requires. Because in the initial
stage, FAR already knows the regular information of the whole network
topology and does not need to periodically do SPF operation.

One of the examples of "more topological exchange information":

In the OSPF protocol, LSA floods every 1800 seconds. Especially in
the larger network, the occupation of CPU and band bandwidth will
soon reach the router's performance bottleneck. In order to reduce
these adverse effects, OSPF introduced the concept of Area, which
still has not solved the problem thoroughly. By dividing the OSPF
Area into several areas, the routers in the same area do not need to
know the topological details outside their area. (In comparison with
FAR, after OSPF introducing the concept of Area, the equivalent paths
cannot be selected in the whole network scope)

OSPF can achieve the following results by Area : 1) Routers only need
to maintain the same link state databases as other routers within the
same Area, without the necessity of maintaining the same link state
database as all routers in the whole OSPF domain. 2) The reduction
of the link state databases means dealing with relatively fewer LSA,
which reduces the CPU consumption of routers; 3) The large number of
LSAs flood only within the same Area. But, its negative effect is
that the smaller number of routers which can be managed in each OSPF
area. On the contrary, because FAR does not have the above
disadvantages, FAR can also manage large-scale network even without
dividing Areas.

The aging time of OSPF is set in order to adapt to routing
transformation and protocol message exchange happened frequently in
the irregular topology. Its negative effect is: when the network
does not change, the LSA needs to be refreshed every 1800 seconds to
reset the aging time. In the regular topology, as the routings are
fixed, it does not need the complex protocol message exchange and
aging rules to reflect the routing changes, as long as LFA mechanism
in the FAR is enough.

Compared with the LSVR protocol, the LSVR protocol has no special
requirements for the network topology structure, however, the FAR
draft is applicable to the regular topology network architecture and
simplifies unnecessary processing. It is a solution proposed to
greatly improve the routing efficiency of the regular network
topology. The FAR solution is more efficient than the general
methods such as LSVR in regular topology.

Therefore, in FAR, we can omit many unnecessary processing and the
packet exchange. The benefits are fast convergence speed and much
larger network scale than other dynamic routing protocol. Now there
are some successful implementations of simplified routings in the
regular topology in the HPC environment. Conclusion: As FAR needs
few routing entries and the topology is regular, the database does
not need to be updated regularly. Without the need for aging, there
is no need for CPU and bandwidth overhead brought by LSA flood every

30 minutes, so the expansion of the network has no obvious effect on
the performance of FAR, which is contrary to OSPF.

Comparison of convergence time: The settings of OSPF spf_delay and
spf_hold_time can affect the change of convergence time. The
convergence time of the network with 2480 nodes is about 15-20
seconds; while the FAR does not need to calculate the SFP, so there
is no such convergence time.
These issues still exist in rapid convergence technology of OSPF
,ISIS (such as I-SPF) and LSVR. The convergence speed and network
scale constraint each other. FAR does not have the above problems,
and the convergence time is almost negligible.
Can FRR solve these problems? IP FRR has some limitations. The
establishment of IP FRR backup scheme will not affect the original
topology and traffic forwarding which are established by protocol,
however, we can not get the information of whereabouts and status
when the traffic is switched to an alternate next hop.

3.3. Network Addressing Issues

Routers are typically configured with multiple network interfaces,
each connected to a subnet. OSPF and other routing algorithms
require that each interface of a router must be configured with an IP
address. A large-scale data center network may contain thousands of
routers and each router has dozens of network interfaces, thus, there
are tens of thousands of IP addresses needed to be configured in a
data center. It will be very complex to configure and manage a large
number of network interfaces and will be difficult to troubleshoot
network problems, then network maintenance will be costly and error-
prone.

In FAR, the device position information is encoded in the IP address
of the router. Each router only needs to be assigned a unique IP
address according its location, which greatly solves complex network
addressing issues in large-scale networks.

3.4. Big Routing Table Issues

There are a large number of subnets in the large-scale data center
network. A router may build a routing entry for each subnet, and
therefore the size of routing tables on each router may be very
large. It will increase a router's cost and reduce the querying
speed of the routing table.

FAR uses two measures to reduce the size of its routing tables: a)It
builds a BRT on the regularity of the network topologies; b)It
introduces a new routing table, i.e., a NRT. In this way FAR can

reduce the size of routing tables to only a few dozen routing
entries.

[3.5](). Adaptivity Issues for Routing Algorithms

To implement efficient routing in large-scale datacenters, besides
FAR, some other routing methods are proposed for some specific
network architectures, such as Fat-tree and BCube. These routing
methods are different(from both design and implementation viewpoints)
and not compatible with the conventional routing methods, which
brings big troubles to network equipment providers to develop new
routers supporting various new routing methods.

FAR is a generic routing method. With slight modification, FAR
method can be applied to most of regular datacenter networks.
Furthermore, the structure of routing tables and querying a routing
table in FAR are the same as conventional routing method. If FAR is
adopted, the workload of developing a new type of router will be
significantly decreased.

[3.6](). Virtual Machine Migration Issues

Supporting VM migration is very important for cloud-based datacenter
networks. However, in order to support layer-3 routing, routing
methods including OSPF and FAR require limiting VM migration within a
subnet. For this paradox, the mainstream methods still utilize
layer-3 routing on routers or switches, transmit packets encapsulated
by IPinIP or MACinIP between hosts by tunnels passing through network
to the destination access switch, and then extract original packet
out and send it to the destination host.

By utilizing the aforementioned methods, FAR can be applied to Fat-
tree, MatrixDCN or BCube networks for supporting VM migration in
entire network.

[4](). The FAR Framework

FAR requires that a DCN has a regular topology, and network devices,
including routers, switches, and servers, are assigned IP addresses
according to their locations in the network. In other word, we can
locate a device in the network according to its IP address.

FAR is a distributed routing method. In order to support FAR, each
router needs to have a routing module that implements the FAR
algorithm. FAR algorithm is composed of three parts, i.e., link-
state learning, routing table building and routing table querying, as
shown in Fig. 1.

```
              Link-state Learning     |Routing Table  | Routing Table
                                       |  Building
                                       |               |   Querying
    +--------+   /---------------\     | +--------+    |    Packets
    |2 Device|<--| 1 Neighbor &  |----->| 5 BRT   \    |
    |Learning|   | Link Detection|     | |Building|\   |       |
    +--------+   \---------------/     | +--------+ \  |      \|/
         |            |                 |            \ |+--------------+
         |            |                 |            /||  7 Querying  |
        \|/          \|/                |           / || Routing Table|
         +-----------------------+|          /   |+--------------+
         |3 Invisible Neighbor & ||       /      |
         |Link Failure Inferring ||  +--------    |
         +-----------------------+|/| 6 NRT   |   |
                  |                / |Building|   |
                 \|/            /| +--------+    |
         +--------------+  / |              |
         |4 Link Failure| / |              |
         |  Learning    |   |              |
         +--------------+   |              |
                            |              |
```
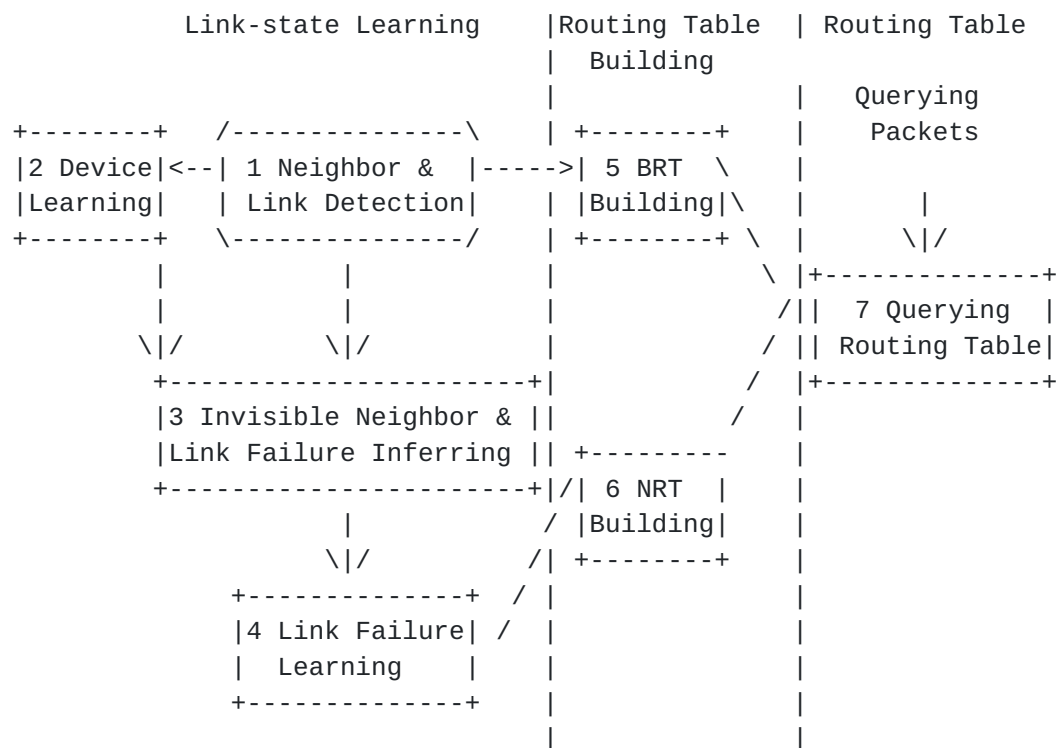
                     Figure 1: The FAR framework

   Link-state learning is responsible for a router to detect the states
   of its connected links and learn the states of all the other links in
   the entire network. The second part builds two routing tables, a
   basic routing table (BRT) and an negative routing table (NRT),
   according to the learned link states in the first part. The third
   part queries the BRT and the NRT to decide a next forwarding hop for
   the received (ingress) packets.

5. Data Format

5.1. Data Tables

   Some data tables are maintained on each router in FAR. They are:

   Neighbor Device Table (NDT): To store neighbor routers and related
   links.

   All Devices Table (ADT): To store all routers in the entire network.

   Link Failures Table (LFT): To store all link failures in the entire
   network.

   Basic Routing Table (BRT): To store the candidate routes.

Negative Routing Table(NRT): To store the avoiding routes.

The format of NDT

```
 -----------------------------------------------------------
 Device ID | Device IP | Port ID | Link State | Update Time
 -----------------------------------------------------------
```

Device ID: The ID of a neighbor router.

Device IP: The IP address of a neighbor router.

Port ID: The port ID that a neighbor router is attached to.

Link State: The state of the link between a router and its neighbor router. There are two states: Up and Down.

Update Time: The time of updating the entry.

The format of ADT

```
 ----------------------------------------------------
 Device ID | Device IP | Type | State | Update Time
 ----------------------------------------------------
```

Device ID: The ID of a neighbor router.

Device IP: The IP address of a neighbor router.

Type: The type of a neighbor router.

State: The state of a neighbor router. There are two states: Up and Down.

Update Time: The time of updating the entry.

The format of LFT

```
 ----------------------------------------------
 No | Router 1 IP | Router 2 IP | Timestamp
 ----------------------------------------------
```

No: The entry number.

Router 1 IP: The IP address of one router that a failed link connects to.

Router 2 IP: The IP address of another router that a failed link connects to.

Timestamp: It identifies when the entry is created.

The format of BRT

```
 ---------------------------------------------------------
 Destination | Mask | Next Hop | Interface | Update Time
 ---------------------------------------------------------
```

Destination: A destination network

Mask: The subnet mask of a destination network.

Next Hop: The IP address of a next hop for a destination.

Interface: The interface related to a next hop.

Update Time: The time of updating the entry.

The format of NRT

```
 --------------------------------------------------------------------
 Destination| Mask| Next Hop| Interface| Failed Link No| Timestamp
 --------------------------------------------------------------------
```

Destination: A destination network.

Mask: The subnet mask of a destination network.

Next Hop: The IP address of a next hop that should be avoided for a destination.

Interface: The interface related to a next hop that should be avoided.

Failed Link No: A group of failed link numbers divided by "/", for example 1/2/3.

Timestamp: The time of updating the entry.

5.2. Messages

Some protocol messages are exchanged between routers in FAR.

Hello Message: This message is exchanged between neighbor routers to learn adjacency.

Device Announcement (DA): Synchronize the knowledge of routers between routers.

Link Failure Announcement (LFA): Synchronize link failures between routers.

Device and Link Request (DLR): When a router starts, it requests the knowledge of routers and links from its neighbors by a DLR message.

A FAR Message is directly encapsulated in an IP packet. The protocol field of IP header indicates an IP packet is an FAR message. The protocol of IP for FAR should be assigned by IANA.

The four types of FAR messages have same format of packet header, called FAR header (as shown in Figure 2).

```
        |<--- 1 --->| <--- 1 --->|<--------- 2 ---------->|
        +-----------+------------+------------------------+
        | Version   |Message Type|    Message Length      |
        +-----------+------------+------------------------+
        |        Checksum        |        AuType          |
        +------------------------+------------------------+
        |                   Authentication                |
        +-------------------------------------------------+
        |                   Authentication                |
        +-------------------------------------------------+
        |                     Timestamp                   |
        +-------------------------------------------------+
```
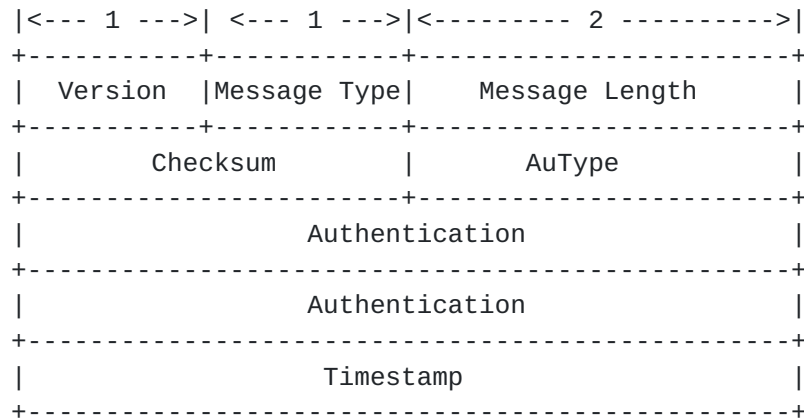
Figure 2: The format of FAR header

Version: FAR version

Message Type: The type of FAR message.

Packet Length: The packet length of the total FAR message.

Checksum: The checksum of an entire FAR message.

AuType:Authentication type. 0: no authentication, 1: Plaintext
Authentication, 2: MD5 Authentication.

Authentication: Authentication information. 0: undefined, 1: Key, 2:
key ID, MD5 data length and packet number. MD5 data is appended to
the backend of the packet.

AuType and Authentication can refer to the definition of OSPF packet.

```
     |<--- 1 --->| <--- 1 --->|<--------- 2 ---------->|
     +-----------+-----------+-----------------------+
     | Version   |Message Type|    Message Length     |
     +-----------+-----------+-----------------------+
     |      Checksum         |        AuType         |
     +-----------------------+-----------------------+
     |                  Authentication               |
     +-----------------------------------------------+
     |                  Authentication               |
     +-----------------------------------------------+
     |                   Timestamp                   |
     +-----------------------------------------------+
     |                   Router IP                   |
     +-----------------------+-----------------------+
     |     HelloInterval     |    HelloDeadInterval  |
     +-----------------------+-----------------------+
     |               Neighbor Router IP              |
     +-----------------------------------------------+
     |                      ...                      |
     +-----------------------------------------------+
```

Figure 3: The Format of Hello Messages

For Hello messages, the Message Type in FAR header is set to
1.Besides FAR header, a Hello message(Fig. 3) requires the following
fields:

Router IP: The router IP address.

HelloInterval: The interval of sending Hello messages to neighbor
routers.

RouterDeadInterval: The interval to set a neighbor router dead(out-
of-service). If in the interval time, a router doesn't receive a
Hello message from its neighbor router, the neighbor router is
treated as dead.

Neighbor Router IP: The IP address of a neighbor router. All the
neighbor router's addresses should be included in a Hello message.

```
     |<----1---->| <----1---->|<----------2----------->|
     +-----------+------------+------------------------+
     |  Version  |Message Type|    Message Length      |
     +-----------+------------+------------------------+
     |       Checksum         |        AuType          |
     +------------------------+------------------------+
     |                  Authentication                 |
     +-------------------------------------------------+
     |                  Authentication                 |
     +-------------------------------------------------+
     |                   Timestamp                     |
     +-----------------------+-------------------------+
     |                   Router1 IP                    |
     +-------------------------------------------------+
     |                      ...                        |
     +-------------------------------------------------+
```

Figure 4: The Format of DA Messages

For DA messages(Fig. 4), the Message Type in FAR header is set to 2.
Besides FAR header, a DA message includes IP addresses of all the
announced routers.

```
          |<----1---->| <----1---->|<----------2---------->|
          +-----------+------------+-----------------------+
          | Version   |Message Type|   Message Length      |
          +-----------+------------+-----------------------+
          |      Checksum          |        AuType         |
          +------------------------+-----------------------+
          |                Authentication                  |
          +------------------------------------------------+
          |                Authentication                  |
          +------------------------------------------------+
          |                  Timestamp                     |
          +------------------------+-----------------------+
          |                   Left IP                      |
          +------------------------------------------------+
          |                   Right IP                     |
          +------------------------+-----------------------+
          |                    State                       |
          +------------------------------------------------+
          |                     ...                        |
          +------------------------------------------------+
```
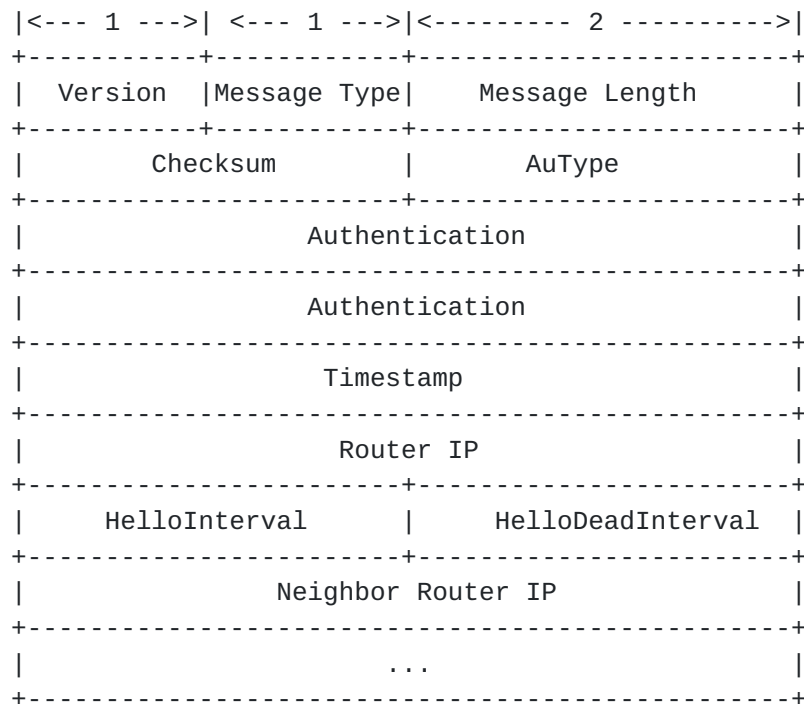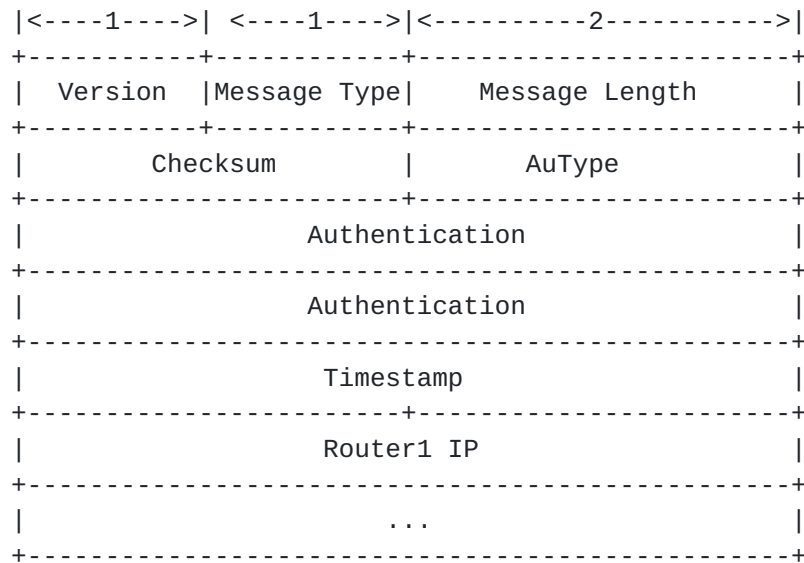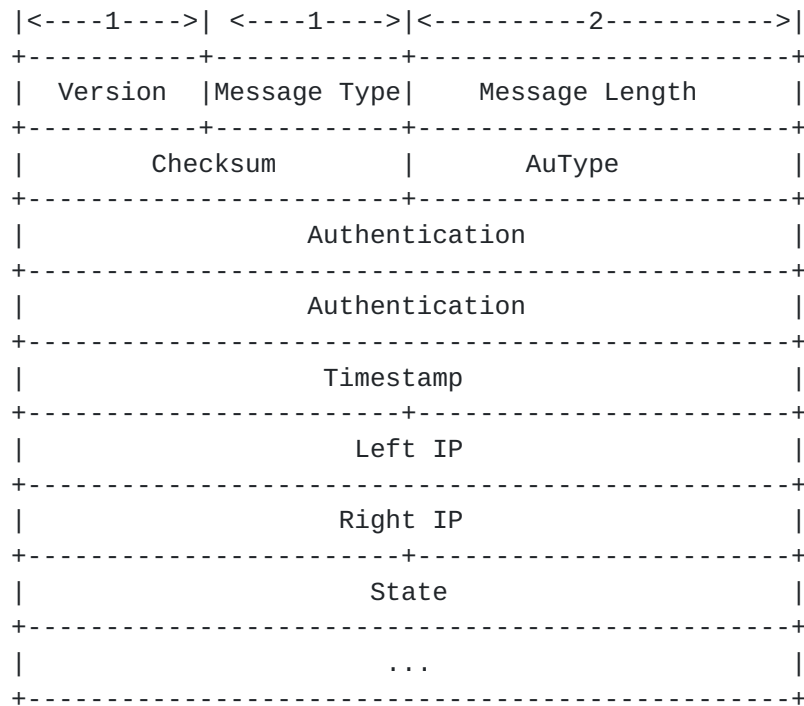
Figure 5: The Format of LFA Messages

For LFA messages(Fig. 5), the Message Type in FAR header is set to 3.
Besides FAR header, a LFA message includes all the announced link
failures.

Left IP: The IP address of the left endpoint router of a link.

Right IP: The IP address of the right endpoint router of a link.

State: Link state. 0: Up, 1: down

```
        |<----1---->| <----1---->|<----------2----------->|
        +-----------+------------+------------------------+
        |  Version  |Message Type|     Message Length     |
        +-----------+------------+------------------------+
        |        Checksum        |         AuType         |
        +------------------------+------------------------+
        |                   Authentication                |
        +-------------------------------------------------+
        |                   Authentication                |
        +-------------------------------------------------+
        |                     Timestamp                   |
        +-------------------------------------------------+
```
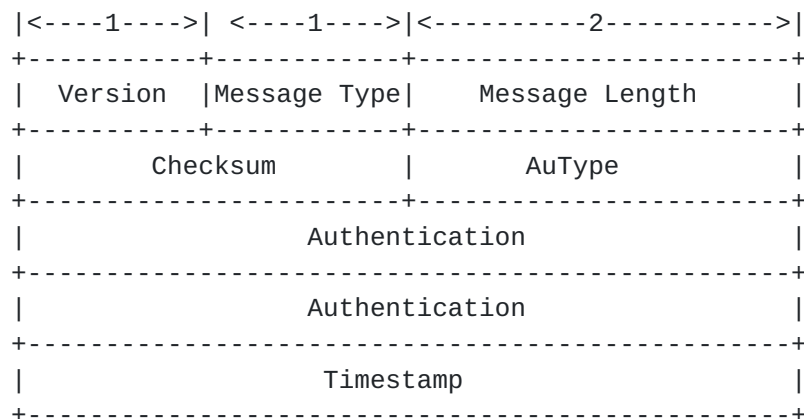
Figure 6: The Format of DLR Messages

For DLR messages(Fig. 6), the Message Type in FAR header is set to
1.Except for FAR header, DLR has no additional fields.

6. FAR Modules

6.1. Neighbor and Link Detection Module(M1)

M1 is responsible for sending and receiving Hello messages, and
detecting directly-connected links and neighbor routers. Each Hello
message is encapsulated in a UDP packet. M1 sends Hello messages
periodically to all the active router ports and receives Hello
messages from its neighbor routers. M1 detects neighbor routers and
directly-connected links according to received Hello Messages and
stores these neighbors and links into a Neighbor Devices Table (NDT).
Additionally, M1 also stores neighbor routers into an All Devices
Table (ADT).

6.2. Device Learning Module(M2)

M2 is responsible for sending, receiving, and forwarding device
announcement (DA) messages, learning all the routers in the whole
network, and deducing faulted routers. When a router starts, it
sends a DA message announcing itself to its neighbors and a DLR
message requesting the knowledge of routers and links from its
neighbors. If M2 module of a router receives a DA message, it checks
whether the router encapsulated in the message is in an ADT. If the
router is not in the ADT, M2 puts this router into the ADT and
forwards this DA message to all the active ports except for the
incoming one, otherwise, M2 discards this message directly. If M2
module of a router receives a DLR message, it replies a DA message
that encapsulates all of the learned routers.

6.3. Invisible Neighbor and Link Failure Inferring Module(M3)

   M3 is responsible for inferring invisible neighbors of the current
   router by means of the ADT. If the link between a router A and its
   neighbor B breaks, which results in that M1 module of A cannot detect
   the existence of B, then B is an invisible neighbor of A. Since a
   device's location is coded into its IP address, it can be judged
   whether two routers are adjacent, according to their IP addresses.
   Based on this idea, M3 infers all of the invisible neighbors of the
   current router and the related link failures. The results are stored
   into a NDT. Moreover, link failures also are added into a link-
   failure table (LFT). LFT stores all of the failed links in the
   entire network.

6.4. Link Failure Learning Module(M4)

   M4 is responsible for sending, receiving and forwarding link failure
   announcement (LFA) and learning all the link failures in the whole
   network. M4 broadcasts each newly inferred link failure to all the
   routers in the network. Each link failure is encapsulated in a LFA
   message and one link failure is broadcasted only once. If a router
   receives a DLR request from its neighbor, it will reply a LFA message
   that encapsulates all the learned link failures through M4 module.
   If M4 receives a LFA message, it checks whether the link failure
   encapsulated in the message is in a LFT by comparing two link ends
   and timestamp. If the link failure is not in the LFT or timestamp is
   different, M4 puts this link failure into the LFT (or update
   timestamp only) and forwards this LFA message to all the active ports
   except for the incoming one, otherwise, M4 discards this message
   directly.

   There is a special case a router will rebroadcast a link failure. If
   a router receives a data packet and must forward the packet going
   ahead to destination through a failed link, it means some previous
   router should avoid this failed link according to its NRT but it
   doesn't. In this case, maybe the previous router missed the LFA
   message of the link failure due to some uncertain reasons. So the
   forwarding router rebroadcasts the LFA message.

6.5. BRT Building Module(M5)

   M5 is responsible for building a BRT for the current router. By
   leveraging the regularity in topology, M5 can calculate the routing
   paths for any destination without the knowledge of the topology of
   whole network, and then build the BRT based on a NDT. Since the IP
   addresses of network devices are continuous, M5 only creates one
   route entry for a group of destination addresses that have the same
   network prefix by means of route aggregation technology. Usually,

the size of a BRT is very small. The detail of how to build a BRT is
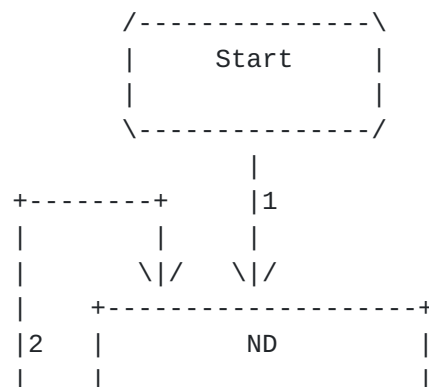described in section 5.

6.6. NRT Building Module(M6)

M6 is responsible for building a NRT for the current router. Because
M5 builds a BRT without considering link failures in network, the
routing paths calculated by the BRT cannot avoid failed links. To
solve this problem, a NRT is used to exclude the routing paths that
include some failed links from the paths calculated by a BRT. M6
calculate the routing paths that include failed links and stored them
into the NRT. The details of how to build a NRT is described in
section 5.

6.7. Routing Table Lookup(M7)

M7 is responsible for querying routing tables and selecting the next
hop for forwarding the packets. Firstly, M7 takes the destination
address of a forwarding packet as a criterion to look up route
entries in a BRT based on longest prefix match. All of the matched
entries are composed of a candidate hops list. Secondly, M7 look up
negative route entries in a NRT taking the destination address of the
forwarding packet as criteria. This lookup is not limited to the
longest prefix match, any entry that matches the criteria would be
selected and composed of an avoiding hops list. Thirdly, the
candidate hops minus avoiding hops are composed of an applicable hops
list. At last, M7 sends the forwarding packet to any one of the
applicable hops. If the applicable list is empty, the forwarding
packet will be dropped.

7. How a FAR Router Works

Figure 7 shows how a FAR router works by its FSM.

```
                    /---------------\
                    |     Start     |
                    |               |
                    \---------------/
                            |
         +--------+        |1
         |        |        |
         |       \|/     \|/
         |     +-------------------+
         |2    |         ND        |
         |     |                   |
```

```
             |    +------------------+
             |    |        |    |
             |    |        |    |3
             +--------+    |
                           \|/
                  +-------------+
                  |   ND-FIN    |
                  |             |
                  +-------------+
                         |
                         |4
                         |
                        \|/
     _____10_____\+-------------+/_____11_____
     |                  /|             |\                |
     |                   |   Listen     |                |
     |        ____9_____\|             |/_____12___    |
     |        |          /|             |\           |    |
     |        |           +-------------+            |    |
     |        |         5/ | |         \8            |    |
     |        |         |/_ | |        _\|           |    |
     |        | +------------+  6|  |7  +------------+ |   |
     |        --| HELLO-RECV |   |  |   | LFA-RECV   |--  |
     |          +------------+   |  |   +------------+    |
     |                 _____|   |_____                 |
     |                 |          |                       |
     |                \|/        \|/                      |
     |          +------------+  +------------+            |
     |_____| DLR-RECV   |  |  DA-RECV   |_____|
                +------------+  +------------+
```

```
                   _____
                   |        |
                   |        |
                  \|/       |
            +-------------+ |13
            | Hello Thread | |
            +-------------+ |
                   |        |
                   |_____|
```

```
                   _____
                   |        |
                   |        |
                  \|/       |
            +-------------+ |14
            |  LFD Thread | |
```

```
           +--------------+  |
                  |         |
                  |_____|


             _____
                  |         |
                  |         |
                 \|/        |
           +--------------+  |15
           |   DA Thread  |  |
           +--------------+  |
                  |         |
                  |_____|


             _____
                  |         |
                  |         |
                 \|/        |
           +--------------+  |16
           |   DFD Thread |  |
           +--------------+  |
                  |         |
                  |_____|
```
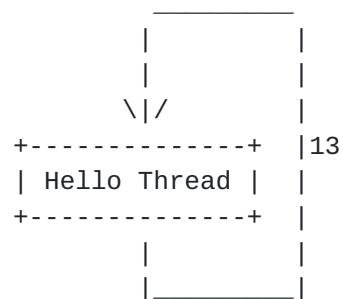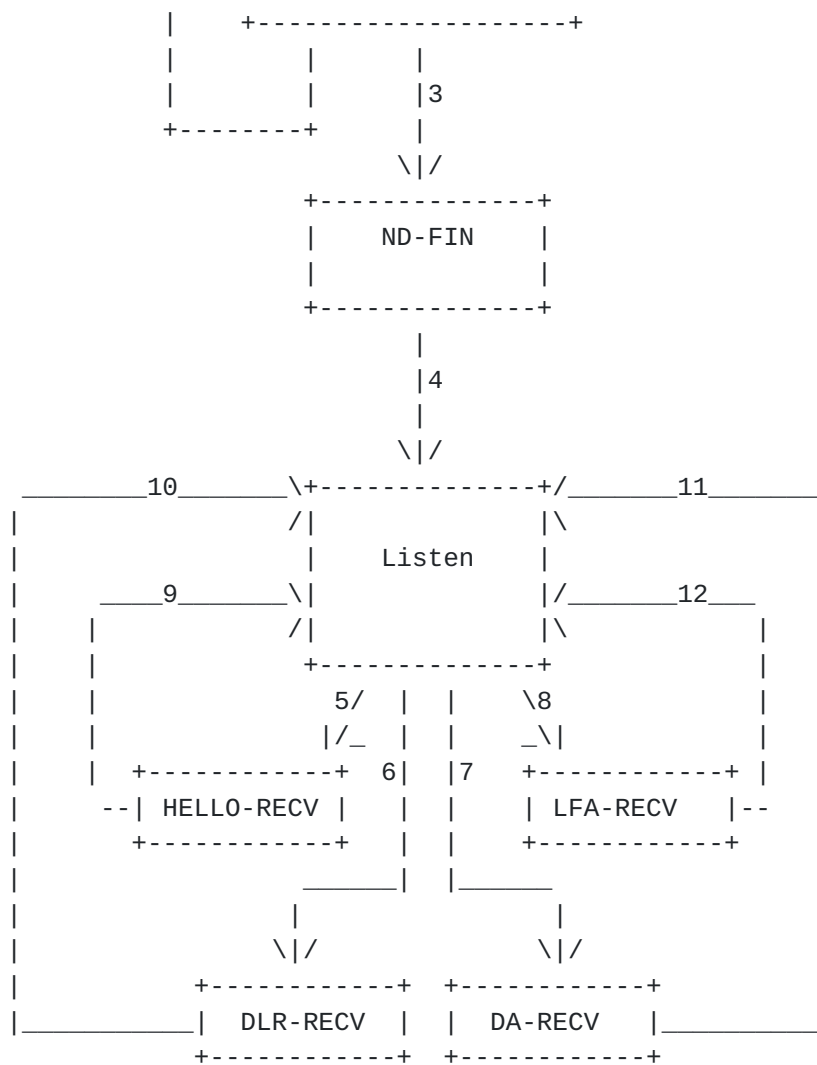
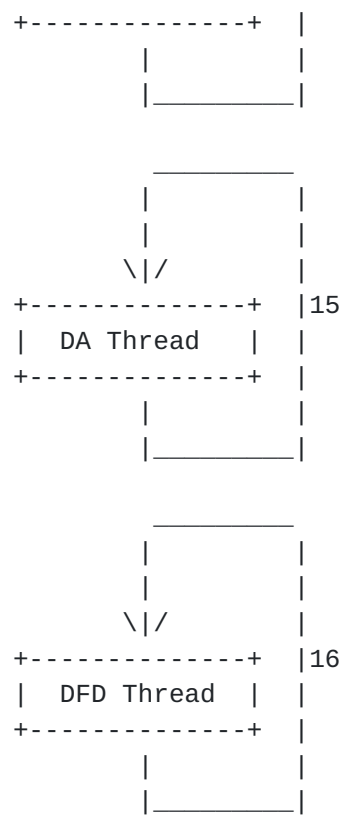   Figure 7: The Finite State Machine of FAR Router

   1)When a router starts up, it starts a Hello thread and then starts
   ND (neighbor detection) timer (3 seconds). Next the router goes into
   ND (neighbor detection) state.
   2)In the ND state, if a router received a Hello message, then it
   performs a Hello-message processing and goes back to the ND state.
   3)When the ND timer is over, a router goes into ND-FIN (neighbor
   detection finished) state.
   4)A router starts the LFD (link failure detection) thread and DFD
   (device failure detection) state, and sends DA message and DLR
   message to all of its active ports. Then the router goes into Listen
   state.
   5) If a router receives a Hello message, then goes into HELLO-RECV
   state.
   6) If a router receives a DLR message, then goes into DLR-RECV state.
   7) If a router receives a DA message, then goes into DA-RECV state.
   8) If a router receives a LFA message, then goes into LFA-RECV state.
   9) A router performs the Hello-message processing. After that, it
   goes back to Listen state.
   10) A router performs the DLR-message processing. After that, it
   goes back to Listen state.
   11) A router performs the DA-message processing. After that, it goes
   back to Listen state.

12) A router performs the LFA-message processing. After that, it
goes back to Listen state.
13) Hello thread produces and sends Hello messages to all its ports
periodically.
14) LFD thread calls link-failure-detection processing to check link
failures in all links periodically
15) DA thread produces and sends DA messages periodically (30
minutes).
16) When DFD thread starts up, it sleep a short time (30 seconds) to
wait for a router learning all the active routers in the network.
Then the thread calls the device-failure-detection processing to
check device failures periodically (30 minutes).

8. Compatible Architecture

As a generic routing protocol, FAR can be run in various DCNs with
regular topology. Up to now, we have implemented the FAR protocol
for 4 types of DCN, including Fat-tree, BCube, MatrixDCN and Diamond.

For different network architectures, most processing of FAR is same
besides calculation of routing tables. BRT routing tables are
calculated based on Hello messages and NRT routing tables are
calculated based on LFA messages in FAR. To extend FAR to support a
new network architecture, only processing of Hello and LFA messages
need providing to build BRT and NRT routing tables.

In this protocol, FAR can support maximally 12 network architectures
and at least support 1 built-in network architecture, such as Fat-
tree, BCube and MatrixDCN,etc. Each network architecture is assigned
a unique number from 1 to 12. For example, if the 1 built-in
architectures are assigned 1, and other customized architectures are
assigned 2 to 12.
1: Fat-tree
2: BCube
3: MatrixDCN.
4: xxx.
......
12: xxx.

9. Application Example

In this section, we take a Fat-tree network(Fig. 7) as an example to
describe how to apply FAR routing. Since M1 to M4 are very simple,
we only introduce how the modules M5, M6, and M7 work in a Fat-tree
network.

A Fat-tree network is composed of 4 layers. The top layer is core
layer, and the other layers are aggregation layer, edge layer and

server layer. There are k pods, each one containing two layers of
k/2 switches. Each k-port switch in the edge layer is directly
connected to k/2 hosts. The remaining k/2 ports are connected to k/2
of the k-port switches in the aggregation layer. There are (k/2)2
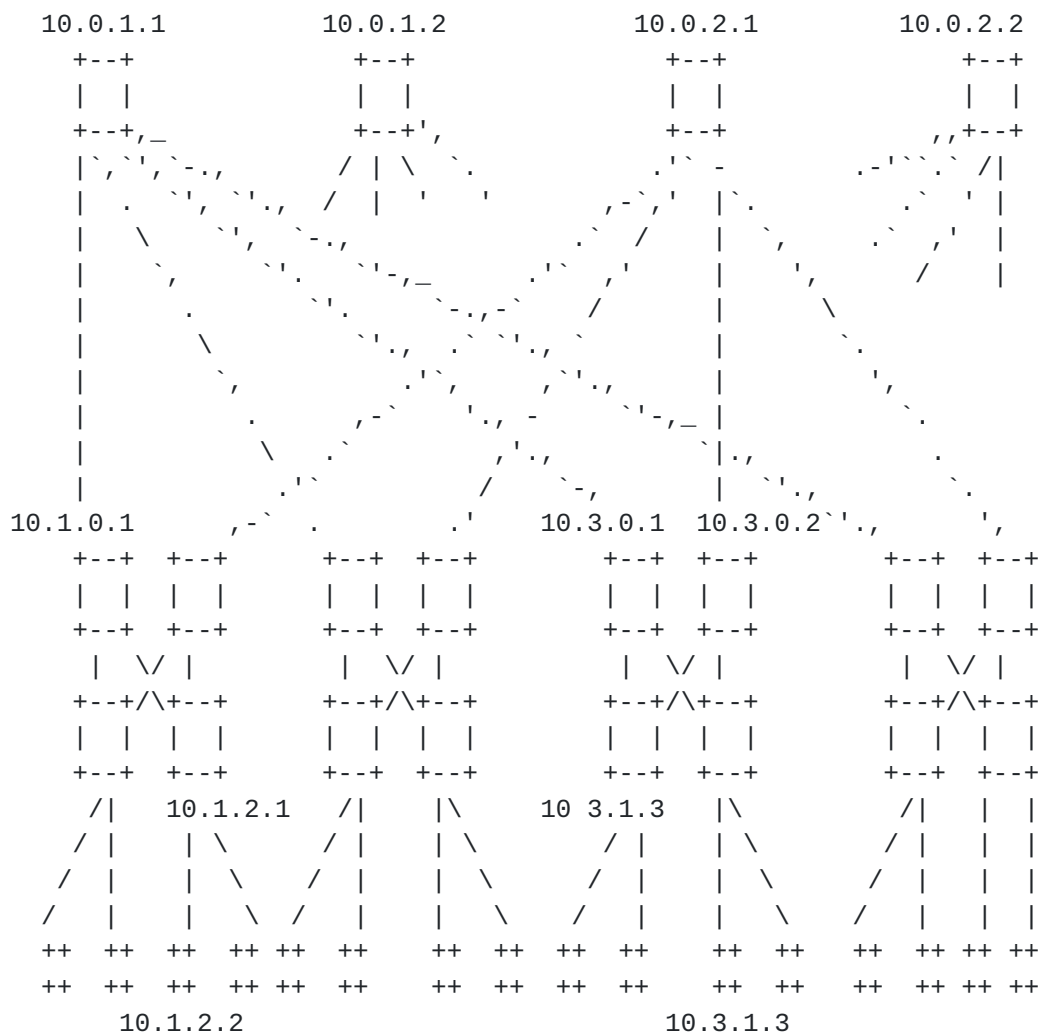k-port core switches. Each core switch has one port connected to
each of the k pods.

```
      10.0.1.1            10.0.1.2              10.0.2.1              10.0.2.2
       +--+                +--+                  +--+                  +--+
       |  |                |  |                  |  |                  |  |
       +--+,_              +--+',                +--+                ,,+--+
       |`,`|,`-.,        / | \  `.             .'` -            .-'``.` /|
       |  .  `|,  `|.,  /  |  '   '          ,-`,' |`.          .`  ' |
       |   \    `|,   `-.,/   |                . /   |   `,      .`  ,' |
       |     `    `|.   `|-,_     .'`  ,'      |   ',     /    |
       |      .      `|.      `-.,-`   /       |      \      `.
       |       \      `|.,  .`  `|.,  `       |        .
       |        `,     .'`,    ,`|.,        |          ',
       |          .   ,-`      '.,  -     `|-,_ |              .
       |           \ . `       ,'.,           `|.,           .
       |           .'`        /     `-,       |  `|.,         .
     10.1.0.1     ,-`  .        .'   10.3.0.1 10.3.0.2`|.,       ',
       +--+  +--+      +--+  +--+        +--+  +--+        +--+   +--+
       |  |  |  |      |  |  |  |        |  |  |  |        |  |   |  |
       +--+  +--+      +--+  +--+        +--+  +--+        +--+   +--+
        | \/ |          | \/ |            | \/ |            | \/ |
       +--+/\+--+      +--+/\+--+        +--+/\+--+        +--+/\+--+
       |  |  |  |      |  |  |  |        |  |  |  |        |  |  |  |
       +--+  +--+      +--+  +--+        +--+  +--+        +--+   +--+
        /|  10.1.2.1  /|     |\    10 3.1.3  |\         /|   |  |
       / |    | \    / |     | \      / |    | \       / |   |  |
      /  |    |  \  / |     |  \    / |    |  \     / |   |  |
     /   |    |   \/  |     |   \  /  |    |   \   /  |   |  |
     ++  ++  ++  ++ ++  ++      ++  ++  ++  ++    ++  ++   ++  ++ ++ ++
     ++  ++  ++  ++ ++  ++      ++  ++  ++  ++    ++  ++   ++  ++ ++ ++
         10.1.2.2                          10.3.1.3
```

                      Figure 8: Fat-tree Network

Aggregation switches are given addresses of the form 10.pod.0.switch,
where pod denotes the pod number, and switch denotes the position of
that switch in the upper pod (in [1, k/2]). Edge switches are given
addresses of the form 10.pod.switch.1, where pod denotes the pod
number, and switch denotes the position of that switch in the lower
pod (in [1, k/2]). The core switches are given addresses of the form
10.0.j.i, where j and i denote that switch's coordinates in the

(k/2)2 core switch grid (each in[1, (k/2)], starting from top-left).
The address of a host follows the pod switch to which it is connected
to; hosts have addresses of the form: 10.pod.switch.ID, where ID is
the host's position in that subnet (in [2, k/2+1], starting from left
to the right).

9.1. BRT Building Procedure

By leveraging the topology's regularity, every switch clearly knows
how it forwards a packet. When a packet arrives at an edge switch,
if the destination of the packet lies in the same subnet with the
switch, then the switch directly forwards the packet to the
destination server through layer-2 switching. Otherwise, the switch
forwards the packet to any of aggregation switches in the same pod.
When a packet arrives at an aggregation switch, if the destination of
the packet lies in the same pod, the switch forwards the packet to
the corresponding edge switch. Otherwise, the switch forwards the
packet to any of core switches that it is connected to. If a core
switch receives a packet, it forwards the packet to the corresponding
aggregation switch that lies in the destination pod.

The forwarding policy discussed above is easily expressed through a
BRT. The BRT of an edge switch, such as 10.1.1.1, is composed of the
following entries:


```
Destination/Mask        Next hop
10.0.0.0/255.0.0.0      10.1.0.1
10.0.0.0/255.0.0.0      10.1.0.2
```

The BRT of an aggregation switch, such as 10.1.0.1, is composed of
the following entries:


```
Destination/Mask          Next hop
10.1.1.0/255 255.255.0     10.1.1.1
10.1.2.0/255.255.255.0     10.1.2.1
10.0.0.0/255.0.0.0         10.0.1.1
10.0.0.0/255.0.0.0         10.0.1.2
```

The BRT of acore switch, such as 10.0.1.1, is composed of the
following entries:

```
Destination/Mask              Next hop
10.1.0.0/255 255.0.0          10.1.0.1
10.2.0.0/255.255.0.0          10.2.0.1
10.3.0.0/255.255.0.0          10.3.0.1
10.4.0.0/255.255.0.0          10.4.0.1
```

9.2. NRT Building Procedure

The route entries in an NRT are related with link and node failures.
We summarize all types of cases into three (3) catalogs.

9.2.1. Single Link Failure

In Fat-tree, Links can be classified as 3 types by their locations:
1) servers to edge switches; 2) edge to aggregation switches; 3)
aggregation to core switches. Link failures between servers to edge
switches only affect the communication of the corresponding servers
and don't affect the routing tables of any switch, so we only discuss
the second and third type of links failures.

Edge to Aggregation Switches

Suppose that the link between an edge switch, such as 10.1.2.1 (A),
and an aggregation switch, such as 10.1.0.1(B),fails. This link
failure may affect 3 types of communications.

o Sources lie in the same subnet with A, and destinations do not. In
this case, the link failure will only affect the routing tables of A.
As this link is attached to A directly, A only needs to delete the
route entries whose next hop is B in its BRT and add no entries to
its NRT when A's M6 module detect the link failure.

o Destinations lie in the same subnet with A, and sources lie in
another subnet of the same pod. In this case, the link failure will
affect the routing tables of all the edge switches in the same pod
except for A. When an edge switch, such as 10.1.1.1, learns the link
failure, it will add a route entry to its NRT:

```
Destination/Mask              Next hop
10.1.2.0/255.255.255.0        10.1.0.1
```

o Destinations lie in the same subnet with A, sources lie in another
pod. In this case, the link failure will affect the routing tables
of all the edge switches in the other pods. When an edge switch in
one other pod, such as 10.3.1.1, learns the link failure, because all
the routings that pass through 10.3.0.1 to A will certainly pass

through the link between A and B, 10.3.1.1 need add a route entry to
its NRT:

Destination/Mask   Next hop
10.1.2.0/255.255.255.0   10.3.0.1

Aggregation to Core Switches

Suppose that the link between an aggregation switch, such as 10.1.0.1
(A), and a core switch, such as 10.0.1.2(B), fails. This link
failure may affect 2 types of communications.

o Sources lie in the same pod (pod 1) with A, and destinations lie in
  the other pods. In this case, the link failure will only affect the
  routing tables of A. As this link is attached to A directly, A only
  need to delete the route entries whose next hop is B in its BRT and
  add no entries to its NRT when A's M6 module detect the link failure.

o Destinations lie in the same pod (pod 1) with A, and sources lie in
  another pod. In this case, the link failure will affect the routing
  tables of all the aggregation switches in other pods except for pod
  1. When an aggregation switch in one other pod, such as 10.3.0.1,
  learns the link failure, because all the routings that pass through
  10.0.1.2 to the pod 1 where A lies will certainly pass through the
  link between A and B, 10.3.0.1 need add a route entry to its NRT:

Destination/Mask   Next hop
10.1.0.0/255.255.0.0   10.0.1.2

9.2.2. A Group of Link Failures

If all the uplinks of an aggregation switch fail, then this switch
cannot forward packets, which will affect the routing of every edge
switches. Suppose that all the uplinks of the node A (10.1.0.1)
fail, it will affect two types of communications.

o Sources lie in the same pod (pod 1) with A, and destinations lie in
  the other pods. In this case, the link failures will affect the
  routing of the edge switches in the Pod of A. To avoid the node A,
  each edge switch should remove the route entry "10.0.0.0/255.0.0.0
  10.1.0.1" in which the next hop is the node A.

o Destinations lie in the same pod (pod 1) with A, and sources lie in
  other pods. In this case, the link failures will affect the routing
  of edge switches in other pods. For example, if the edge switch
  10.3.1.1 communicates with some node in the pod of A, it should avoid

the node 10.3.0.1, because any communication through 10.3.0.1 to the
pod of A will pass through the node A. So a route entry should be
added to 10.3.1.1:

Destination/Mask   Next hop
10.1.0.0/255.255.0.0   10.3.0.1

9.2.3. Node Failures

At last, we discuss the effect of node failures to a NRT. There are
3 types of node failures: the failure of edge, aggregation and core
switches.

o An edge switch fails. The failure doesn't affect the routing table
of any switch.

o A core switch fails. Only when all the core switches connected to
the same aggregation switch fail, they will affect the routing of
other switches. This case is equal to the case that all the uplinks
of an aggregation switch fail, so the process of link failures can
cover it.

o An aggregation switch fails. This case is similar to the case that
all the uplinks of an aggregation switch fail. It affects the
routing of edge switches in other pods, but doesn't affect the
routing of edge switches in pod of the failed switch. The process of
this failure is same to the second case in section 6.2.2.

9.3. Routing Procedure

FAR decides a routing by looking up its BRT and NRT. We illuminate
the routing procedure by an example. In this example, we suppose
that the link between 10.3.1.1 and 10.3.0.2 and the link between
10.1.2.1 and 10.1.0.2 have failed. Then we look into the routing
procedure of a communication from 10.3.1.3 (source) to 10.1.2.2
(destination).

Step 1: The source 10.3.1.3 sends packets to its default router
10.3.1.1

Step 2: The routing of 10.3.1.1.

1) Calculate candidate hops

10.3.1.1 looks up its BRT and gets the following matched entries:

```
   Destination/Mask             Next hop
   10.0.0.0/255.0.0.0           10.3.0.1
```

So the candidate hops = {10.3.0.1}

2) Calculate avoiding hops

Its NRT is empty, so the set of avoiding hop is empty too.

3) Calculate applicable hops

The applicable hops are candidate hops minus avoiding hops, so:

The applicable hops = {10.3.0.1}

4) Forward packets to 10.3.0.1

Step 3: The routing of 10.3.0.1

1) Calculate candidate hops.

10.3. 0.1 looks up its BRT and gets the following matched entries:

```
   Destination/Mask             Next hop
   10.1.0.0/255.255.0.0         10.0.1.1
   10.1.0.0/255.255.0.0         10.0.1.2
```

So the candidate hops = {10.0.1.1, 10.0.1.2}

2) Calculate avoiding hops

```
   Destination/Mask             Next hop
   10.1.0.0/255.255.0.0         10.0.1.2
```

So the avoiding hops = {10.0.1.2}

3) Calculate applicable hops

The applicable hops are candidate hops minus avoiding hops, so:

The applicable hops = {10.0.1.1}

4) Forward packets to 10.0.1.1

   Step 4: 10.0.1.1 forwards packets to 10.1.0.1 by looking up its
   routing tables.

   Step 5: 10.1.0.1 forwards packets to 10.1.2.1 by looking up its
   routing tables.

   Step 6:10.1.2.1 forwards packets to the destination 10.1.2.2 by
   layer-2 switching.

9.4. FAR's Performance in Large-scale Networks

   FAR has good performance to support large-scale networks. In this
   section, we take a Fat-tree network composed of 2,880 48-port
   switches and 27,648 servers as an example to show FAR's performance.

9.4.1. The number of control messages required by FAR

   FAR exchanges a few messages between routers and only consumes a
   little network bandwidth. Tab. 1 shows the required messages in the
   example Fat-tree network.
   Table 1:Required messages in a Fat-tree network.

Message Type	Scope	size(bytes)	Rate	Bandwidth
Hello	adjacent switches	less than 48	10 messages/sec	less than 4 kbps
DLR	adjacent switches	less than 48	(1)	48bytes
DA	entire network	less than 48	(2)	1.106M
LFA	entire network	less than 48	(3)	48 bytes

   (1)Produce one when a router starts
   (2)The number of switches(2,880) in a period
   (3)Produce one when a link fails or recovers

9.4.2. The Calculating Time of Routing Tables

   A BRT is calculated according to the states of its neighbor routers
   and attached links. An NRT is calculated according to device and
   link failures in the entire network. So FAR does not calculate
   network topology and has no problem of network convergence, which
   greatly reduces the calculating time of routing tables. The
   detection and spread time of link failures is very short in FAR.
   Detection time is up to the interval of sending Hello message. In
   FAR, the interval is set to 100ms, and a link failure will be
   detected in 200ms. The spread time between any pair of routers is

less than 200ms.If a link fails in a data center network, FAR can
detect it, spread it to all the routers, and calculate routing tables
in no more than 500ms.

9.4.3. The Size of Routing Tables

For the test Fat-tree network, the sizes of BRTs and NRTs are shown
in Tab. 2.

Table 2: The size of routing tables in FAR

Routing Table	Core Switch	Aggregation Switch	Edge Switch
BRT	48	48	24
NRT	0	14	333

The BRT's size at a switch is determined by the number of its
neighbor switches. In the example network, a core switch has 48
neighbor switches (aggregation switch), so it has 48 entries in its
BRT.Only aggregation and edge switches have NRTs. The NRT size at a
switch is related to the number of link failures in the network.
Suppose that there are 1000 link failures in the example network, the
number of failed links is 1.2% of total links, which is a very high
failure ratio. We suppose that link failures are uniformly
distributed in the entire network. The NRT size at an edge switch is
about 333 and the NRT size of an aggregation switch is about 14in
average.

10. Security Considerations

The security considerations will be discussed in a future version of
this document.

11. Conclusions

This draft introduces FAR protocol, a generic routing method and
protocol, for data centers that have a regular topology. It uses two
routing tables, a BRT and an NRT, to store the normal routing paths
and the forbidden (to-be-avoided) routing paths, respectively. This
makes the FAR protocol very simple and efficient. The sizes of these
two tables are very small. Usually, a BRT has only several tens of
entries and an NRT has only several or about a dozen entries.

12. **Acknowledgments**

13. **References**

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
   Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March
   1997.

   [FAT-TREE] M. Al-Fares, A. Loukissas, and A. Vahdat."A
   Scalable,Commodity, Data Center Network Architecture",In ACM SIGCOMM
   2008.

Authors' Addresses

   Bin Liu
   ZTE Inc., ZTE Plaza
   No.19 East Huayuan Road,Hai Dian District
   Beijing 100191
   China

   Phone: +86 -010-59932039
   Email: 13683610386@139.com

   Yantao Sun
   Beijing Jiaotong University
   No.3 Shang Yuan Cun, Hai Dian District
   Beijing 100044
   China

   Email: ytsun@bjtu.edu.cn

   Jing Cheng
   Beijing Jiaotong University
   No.3 Shang Yuan Cun, Hai Dian District
   Beijing 100044
   China

   Email: yourney.j@gmail.com

Yichen Zhang
Beijing Jiaotong University
No.3 Shang Yuan Cun, Hai Dian District
Beijing 100044
China

Email: snowfall_dan@sina.com

Bhumip Khasnabish
ZTE TX Inc.
55 Madison Avenue, Suite 160
Morristown, New Jersey   07960
USA

Phone: +001-781-752-8003
Email: vumip1@gmail.com
URI: http://tinyurl.com/bhumip/