ISSN:0975-9646

Idea of handling short-lived network failures using LOLS and loop free convergence using FCFR

Rama Gaikwad[1] and S. P. Pingat [2]

[1] PG Fellow, Department of Computer Engineering, S.K.N. College of Engineering , Pune,Maharashtra, India.

[2] Assistant Professor, Department of Computer Engineering, S.K.N. College of Engineering, Pune,Maharashtra, India.

Abstract. The short-lived failures are reasonably common in IP networks, there are many ways with which local rerouting can be provided for high availability but most of them are for single link failure. Here, we are suggesting a Localized On-demand Link State (LOLS) routing to safeguard the forwarding continuity even with multiple failures limiting the propagation of failure information to just a few hops. LOLS cannot guarantee loop-free forwarding during route convergence and this is the reason why we are working on integrating Fast Convergence Fast Reroute (FCFR) technique with LOLS to ensure loop free rerouting and convergence even with multiple failures. Fast Convergence with Fast Reroute (FCFR), which uses a fast reroute scheme such as Not-Via and desires just one additional bit in the packet header with much less per-packet overhead. Integrating LOLS with FCFR is going to hold the quality of LOLS of loop free forwarding and overcome drawback of LOLS by ensuring loop free convergence.

Keywords: **Fast Reroute, Failure Resilience, Local Rerouting.**

1. INTRODUCTION

The internet plays an important role in our lives these days. Providing nonstop service availability even with momentary failures is the foremost challenge for the service providers. Unfortunately, service disturbances occur even in well coped networks due to either link or node failure or both. To upkeep evolving time-sensitive requests in today’s Internet, these networks need to endure failures with minimal service disruption. For example, a disruption time of longer than 50 ms is considered intolerable for mission-critical applications [1].Hence, it is important to formulate schemes that protect the network against not only single failures but also *multiple independent failures*.

The essential concept behind LOLS(Localized On-demand Link State Routing) is to have packets transmit a blacklist of degraded links come across along the path that are to be avoided in order to ensure loop-free forwarding. The desirability of LOLS is that a packet’s blacklist is reset as soon as it makes forward progress towards the destination, limiting the spread of failure information to just a few hops. LOLS considers a link as degraded if its current state (say “down”) is worse than its globally advertised state (say “up”). Under LOLS, each packet carries a blacklist (a minimal set of degraded links come across along its path), and the next hop is determined by excluding the blacklisted links [2]. A packet’s blacklist is initially void and remains blank when there is no disagreement between the current and the advertised states of links along its path. But when a packet reaches at a node with a tainted link neighboring to its next hop, that link is added to the packet’s blacklist. The packet is then advanced to an alternate next hop.

The packet’s blacklist is retune to empty when the next hop makes forward progress, i.e., the next hop has a smaller path to the destination than any of the nodes navigated by the packet. With these simple steps, LOLS propagates the state of degraded links only when essential, and as far as necessary, and confirms loop-free delivery to all local destinations.

LOLS cannot promise loop-free forwarding during route convergence[2]. Traditional routing schemes such as OSPF activate link state advertisements in reaction to a modification in topology, and cause network-wide computation of routing tables. Such a global rerouting suffers some delay before traffic forwarding can resume on another paths. During this convergence delay, routers may have uneven sights of the network, resulting in forwarding or routing loops and dropping of packets [4].

In order to avoid routing loops during this intermediate period, other authors have suggested schemes such as ordered updates [5] and SafeGaurd technique [6].But ordered updates have extends the convergence period and SafeGaurd technique add multiple bytes in the header. Fast Convergence Fast Reroute (FCFR) is a technique which uses such as Not-Via to create alternate path during the convergence process. This is why we are working on integrating Fast Convergence Fast Reroute (FCFR) technique with LOLS to ensure loop free rerouting and convergence even with multiple failures. Fast Convergence with Fast Reroute (FCFR), which employs a fast reroute scheme such as Not-Via and needs just one additional bit in the packet header with much less per-packet overhead [3].

Integrating LOLS with FCFR is going to retain the quality of LOLS of loop free forwarding and overcome drawback by ensuring loop free convergence too. We provide the details of this integration in next sections. The rest of the paper is structured as follows. Section II presents LOLS approach for handling multilink failures and Section III throws light on how FCFR guarantees loop free convergence. In Section IV, we will brief on the idea of integrating LOLS with FCFR and finally in Section V, we conclude the paper.

2. RELATED WORK

Abundant methods have been projected in the past to make networks more robust to failures. We classify them into patterns that guard against single or linked failures and those that can deal with multiple independent letdowns. Also, some of them try to decrease the routing overhead by controlling the link state updates. We concisely describe a few systems fitting to each of these groups in the following section.

Single or Correlated Failures: The Not-via method[7]locally redirects a packet around a recognized failure by encapsulating the packet to an address that tacitly identifies the failed network factor to be avoided. The above systems offer flexibility against single or linked failures but are not intended to improve from multiple unconnected failures.

Multiple Independent Failures*:* Convergence-free routing using Failure Carrying Packets (FCP) [8] can recover from a random quantity of failures. FCP also carries data about the down links in the data packet and intermediary routers ignore those links while calculating the next hop. But, contrasting our system, the failure information under FCP is carried all the way to the endpoint, which is objectionable. Packet Re-cycling (PR) [9] is a method that also objects to decrease the number of bits needed to be carried in a packet header to guarantee effective rerouting. PR takes benefit of cellular graph embedding to redirect packets that would otherwise be dropped in circumstance of failures. It needs only in the order of $\log_2(D)$ bits in the header to shield all non- detaching failure arrangements, here D is the diameter of the network. While the small header overhead is noteworthy, packets under PR take longer way around than LOLS.

3. Proposed System

3.1 Problem Description:

Using LOLS, the multiple link failure can be handled. This makes sure that the data is delivered to destination even if the link is failed. Figure 1 shows the detailed architecture of our system.

Moreover, it does not transmit the facts of link failure to all the destinations if the link is failed for the duration less than the threshold.

The problem definition is to handling multiple link failures in IP network using Localized On-demand Link state routing while, to avoid loops at the time of convergence, integrate it with FCFR technique. The problem with the LOLS is that, it does not promise loop free forwarding at the time of convergence. The issue is handled using integrating the same with fast convergence and fast reroute technique, employs a fast reroute scheme such as Not-Via and desires just one extra bit in the packet header with far less per-packet overhead.

The flow of the activities to be performed is shown in the figure 2. Here we will be creating nodes using JAVA programming and enable user of the system to send and receive the packets. To perform the forwarding of the packets, we shall apply greedy forwarding algorithm. While forwarding the packets using greedy forwarding, if we will come across the down link, we will apply the blacklist based forwarding algorithm.

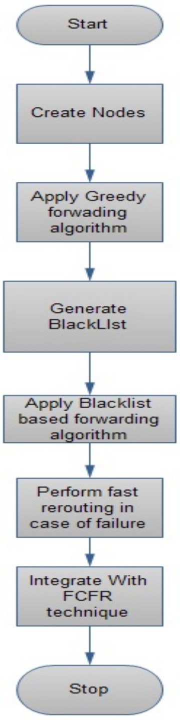
Start
Create Nodes
Apply Greedy forwading algorithm
Generate BlackLIst
Apply Blacklist based forwarding algorithm
Perform fast rerouting in case of failure
Integrate With FCFR technique
Stop

Figure 2: Flow of activities to be performed

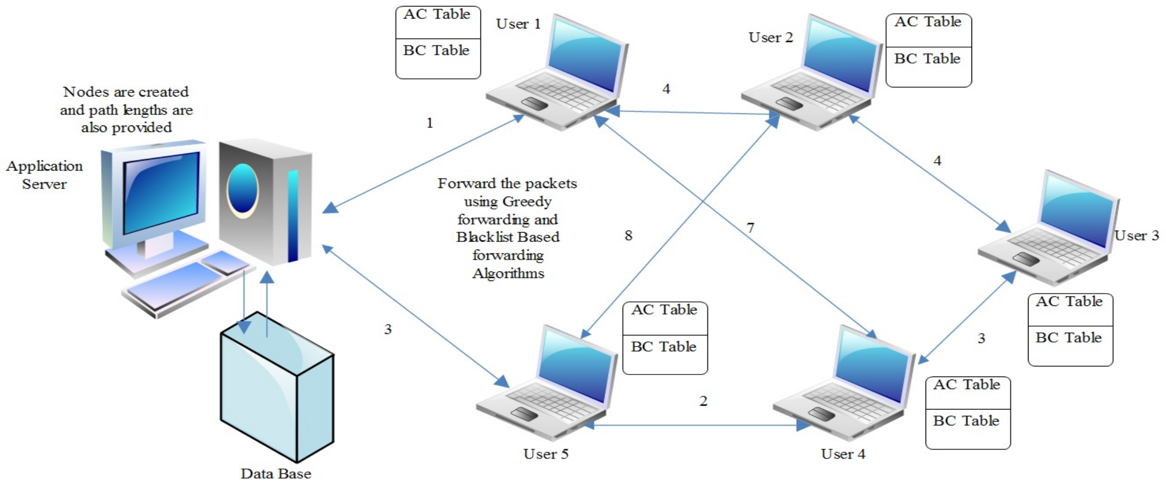

Figure 1: System Architecture

3.2 Greedy Forwarding algorithm:

We need to select a succeeding hop such thatthe packet does not get trapped in a forwarding loop. Anapproach toassure loop-freedom is to apply *greedy forwarding* that forwards the packet along a route with *reducing cost* to the destination, i.e., every hop makes *forward progress* in the direction ofthe destination. It is crucial that the path cost is determinedregularly at all nodes based on the broadcasted topology.

A packet is usually advanced in greedy mode to a succeeding hop along the path with reducing cost (w.r.t. the announced topology) to the endpoint. When a packet come across a dead-end(whose cost to the destination is lesser than any of the potential subsequent hops) in greedy mode, instead of dropping the packet, it is advanced in recovery mode.

In recovery mode, packets carry a blacklist, which is a set of degraded links come across the route. A packet's subsequent hop is selected along a path that does not contain blacklisted links. The forwarding of a packet is swapped back to greedy mode, i.e., the blacklist is retuned to empty, when it reaches at a node with lower cost (w.r.t. the announced topology) to the destination than the node at which it moved in the recovery mode. Thus, LOLS successfully transmits link state on demand, and only to as several nodes as essential.

Algorithm:

Let d be the destination of the packet, j is the adjacent next feasible hop, i is the source node from which the route is to be calculated.

1. If the cost of path from j to d is less than cost of path from i to d then, j is the next feasible hop for which node i has shortest path to destination d.
2. If no feasible hop is present then algorithm returns NULL and the packet is rejected.

We want to point out that this algorithm is a variation of standard greedy forwarding as it does not continuously select a next hop with maximum forward advancement. Instead, it chooses a next hop such that it aggregates to shortest path forwarding when there are no down links, which is surely a desired.

3.3 *Blacklist based forwarding algorithm:*

In Localized On-demand Link State routing, every packet p carries a blacklist p.blist with it while traveling through the network in its header, and packet is to destination or next hops based on both p.dest and p.blist. The blist that is blacklist is initialized to NULL at the source and it is increases or shrinks as and when required during the whole forwarding process.

Algorithm:

1. Find the next hop with smallest path cost and which does not have links present in packet's blacklist.
2. If the links to the neighbor are down or degraded, add these links in the packet's blacklist.
3. Repeat the steps 1 and 2 until either we find the next hop which forwards the packet to the next hop and resets the packet's blacklist, or there is no feasible next hop this means that the destination is unreachable and the packet must be dropped.

There are rules present for updating the packets blacklist p.blist at node i, the rules are briefed here.

1. The link from i to j is added to blacklist if
   a. Link is degraded
   b. No feasible next hop is present without the link i to j
   c. If the link i to j had not been down, then this link could have been the shortest path.
2. The blacklist is retuned to NULL when
   a. The feasible next of is present
   b. The cost from j to destination is less than that of any other node traversed by packet p.

3.4 Integrating with FCFR

We suggest fast convergence with fast reroute (FCFR),which uses an existing technique such as NotVia to generate alternate routing during the convergence procedure. Each router preserves two duplicates of their forwarding information table. The before change(bc) forwarding table relates to the fast reroute topology and the after convergence (ac) table is produced once the router has calculated the restructured topology. Each packet carries a bit that specifies its forwarding mode, i.e., which of these two tables are used for forwarding it that particular bit.

The outcome is that routers which have not yet calculated their altered table scan remain to use the bc tables in order to send packets during the convergence procedure. Routers that have a view of the new topology initiate to forward packets with the ac table. However, if the packets reach to a router with only the bc table, the router will revert to using that table. Once a packet has been sent using a bc forwarding table, the packet cannot return using a route from the ac topology.

This promises that packets which originate at an updated router will every time get transported, either along an ac route, or a grouping of an ac route and the bc route. Packets initiating at not yet updated routers follow the bc path all the way to the destination. Thus, FCFR guarantees loop-freedom while advancing packets along the ideal routes as soon as possible.

During the forwarding phase the above mentioned algorithms will be used, while during convergence period, the forwarding shall happen according to FCFR technique to ensure loop-free convergence too.

3.5 Mathematical Model

U is main set of users like u1, u2, u3….

U = {u1, u2, u3…….}

A is main set of Administrators like a1, a2, a3….

A = {a1, a2, a3…….}

P is main set of participating paths like p1, p2, p3…

P = {p1, p2, p3…….}

Identify the processes as P.

P = {Set of processes}

P = {P1, P2, P3……} & P1 = {e1, e2, e3, e4,e5}

Where

{e1=Find the nodes in the network}

{e2=Provide the weights to the each links in the network}

{e3=Perform Greedy forwarding algorithm}

{e4= generate blacklist for each packet and apply blacklist based forwarding algorithm}

{e5= Integrate system with FCFR}

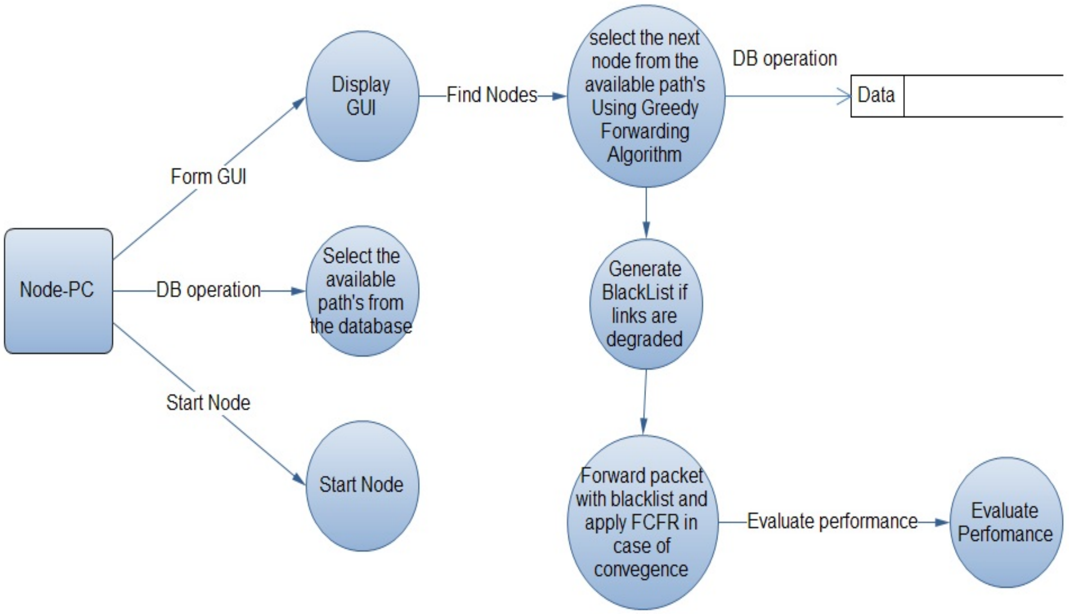
Node-PC
Form GUI
DB operation
Start Node
Display GUI
Select the available path's from the database
Start Node
Find Nodes
select the next node from the available path's Using Greedy Forwarding Algorithm
DB operation
Data
Generate BlackList if links are degraded
Forward packet with blacklist and apply FCFR in case of convegence
Evaluate performance
Evaluate Perfomance

Figure 3: Data Flow Diagram

3.6 *Data Independence and Data Flow Architecture*

A diagram showing the course of information through the function and the change undertakes is presented in figure 3

3.7 *Platform*

The project is to be developed in JAVA using eclipse IDE, RMI and JAVAFx shall be also used.

3.8 *Expected Result*

LOLS cannot promise loop-free forwarding during route convergence for this reason the use of FCFR is done. By integrating these two eminent techniques, the network will be sustainable for short-lived multiple failures and will be loop free in case of convergence. The total propagation distance of failures in case of LOLs is found less than that of the other multiple failure handling techniques like FCP [2].

4. Conclusion and Future Scope

In this paper, we presented an idea of LOLS integrating with FCFR, for controlling multiple failures in IP backbone networks and providing loop free convergence. The fundamental notion behind LOLS is to have packets carry a blacklist of degraded links came across the path that are to be escaped in order to guarantee loop-free forwarding. The significant feature of LOLS is that a packet's blacklist is reset to null as soon as it makes forward movement in the direction of the destination, restricting the propagation of failure information to limited hops. LOLS cannot promise loop-free forwarding during route convergence for this reason the use of FCFR is done. By integrating these two eminent techniques, the network will be sustainable for short-lived multiple failures and will be loop free in case of convergence. This technique can be enhanced for the MANET system.

References

[1] A. Gonzalez and B. Helvik, "Analysis of failures characteristics in the uni-net IP backbone network," in *Proc. 2011 IEEE Workshops of International Conference on Advanced Information Networking and Applications*, pp. 198–203.

[2] Glenn Robertson and Srihari Nelakuditi "Handling Multiple Failures in IP Networks through Localized On-Demand Link State Routing" in *IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT, VOL. 9, NO. 3, SEPTEMBER 2012*

[3]Glenn Robertson, James Bedenbaugh, SrihariNelakuditi "Fast Convergence with Fast Reroute in IP Networks" *in Proc. IEEE Infocom, Mar. 2007.*

[4] U. Hengartner, S. B. Moon, R. Mortier, and C. Diot, "Detection and analysis of routing loops in packet traces," *in IMW, Marseilles, France, Nov. 2002.*

[5]P. Francois and O. Bonaventure, "Avoiding Transient Loops during IGP Convergence in IP Networks," *ACM Transactions on Networking, vol. 15, no. 6, pp. 1280–1292, Dec. 2007.*

[6] A. Li, X. Yang, and D. Wetherall, "SafeGuard: Safe Forwarding during Routing Changes," *in CoNEXT, 2009.*

[7]S. Bryant, M. Shand, and S. Previdi, "IP fast reroute using not-via addresses," *Internet Draft (work in progress), July 2007, draft-ietf-rtgwgipfrr- notvia-addresses-01.txt.*

[8]K. Lakshminarayanan, M. Caesar, M. Rangan, T. Anderson, S. Shenker,and I. Stoica, "Achieving convergence-free routing using failure-carrying packets," in *Proc. 2007 SIGCOMM*, pp. 241–252.

[9] S. S. Lor, R. Landa, and M. Rio, "Packet re-cycling: eliminating packet losses due to network failures," in *Proc. 2010 HotNets*