Loop Resolution Mechanism for Flow-Aware Multi-Topology Adaptive Routing

Robert Wójcik, Jerzy Domżał, Zbigniew Duliński, Piotr Gawłowicz, and Piotr Jurkiewicz

***Abstract*—Flow-Aware Multi-Topology Adaptive Routing is a new proposal which provides multipath transmissions in the networks based on any IP routing protocol. However, due to utilization of flow tables which store flow forwarding information, it suffers from routing instability and the occurrence of failures. This problem is solved by the mechanism proposed in this letter. The mechanism is based on a Time-To-Live field from the IP protocol header. The mechanism is simple, yet very effective. The evaluation shows that at the cost of additional operations performed by routers the problem is completely eliminated.**

***Index Terms*—Multipath, routing, flow, SDN.**

I. Introduction

EVER since the problem of congestions started to appear, an interest in adaptive, i.e., load-sensitive routing mechanisms emerged. One may come up with an idea of changing routing protocol link costs dynamically, according to the current link load. For example, an algorithm may set the cost of a link to a predefined very high value when its load exceeds a defined threshold, and set it back to the default value when it falls below it. The information about the updated cost propagates to all routers which is a standard action performed by each routing protocol. In effect, routing tables change and this affects routing decisions.

A variation of such an idea was used in the first computer networks, for example in ARPANET [1]. The IGRP protocol (EIGRP predecessor) also used the path load information to construct the resulting routing table [2]. However, such an approach was abandoned, mostly due to the stability issues. Suppose that there are two paths, A and B. Path A has lower cost and, therefore, all the traffic is forwarded through it. When its load exceeds a defined threshold, its cost rises and, henceforth, path B becomes the cheaper one. All the traffic from path A is immediately switched to path B. Path A becomes unused. This means that after the next routing update, path A will be preferred again as it has lower cost because it is uncongested. The oscillations will continue and no equilibrium will be reached. Such oscillations between alternative paths lead to instability, which was shown in [3] and [4]. The performance of a network which undergoes oscillations can be even worse than the performance of a network which utilizes a simple fixed routing scheme [4].

Multipath transmission is possible in, e.g., MPLS. This letter [5] reviews various multipath routing possibilities. However, they are mostly based on manual configurations and are very complex. To our knowledge, many major operators have their MPLS nodes configured with such complexity that every change is problematic.

Flow-Aware Multi-Topology Adaptive Routing (FAMTAR) is a routing mechanism which does not introduce route flapping, is conceptually simple, and provides multipath transmissions. It is introduced in Section II. However, FAMTAR is susceptible to failures and this problem is not yet resolved, as is presented in Section III. In Section IV we present a method to solve the occurring problem of failures and loops. The method is then evaluated in Sections V and VI, after which this letter is summarized.

II. Flow-Aware Multi-Topology Adaptive Routing

FAMTAR is a new multipath adaptive routing mechanism, introduced in [6]. Traffic engineering in FAMTAR is based on currently popular concept of flows. Software Defined Networking with its OpenFlow protocol is a similar example. Also, there are many approaches to quality of service assurance which are based on flows [7].

The key idea of FAMTAR, which distinguishes it from the other approaches, is to employ alternative paths only if necessary and only to new transmissions. Upon congestion, new flows use alternative routes, whereas the old ones remain on their primary paths. Effectively, this results not only in adaptive routing, but also in multipath routing as many possible paths can be utilized simultaneously. This has been shown in [8]. To achieve that, the FAMTAR router caches once obtained route for each flow in a so-called Flow Forwarding Table (FFT), which is an entity distinct from the current routing table. FFT is an associative array in which the keys are flow identifying fields, such as, e.g., for the IPv4 protocol: IP addresses, port numbers and the protocol field, for the IPv6 protocol: IP addresses and the flow label field. For each flow, the FFT indicates the interface to which the packets are forwarded. This information is taken from the current routing table when the flow is added to the FFT, i.e., when its first packet appears. Entries in the FFT expire after some inactivity time.

When routing tables change, e.g., due to a link cost change or a failure somewhere in the network, only new flows are going to be routed according to the updated tables. Old flows, which were active before that event, are routed on their existing paths stored in the FFT. Since the original, optimal path does not take new flows, its load gradually declines as existing transmissions naturally end. Therefore, after some time, the cost of links on this path are set back to their default values, and the path starts to accept new traffic again. The described procedure ensures smooth transition of traffic as there are no rapid movements. All changes in the network affect only those flows which start their transmission after the event. To avoid frequent link cost changes, FAMTAR uses a hysteresis-based approach for link congestion notification as described in [6].

Manuscript received January 16, 2015; revised May 26, 2015; accepted May 28, 2015. Date of publication June 1, 2015; date of current version August 10, 2015. The associate editor coordinating the review of this paper and approving it for publication was V. Eramo.

R. Wójcik, J. Domżał, P. Gawłowicz, and P. Jurkiewicz are with Department of Telecommunications, AGH University of Science and Technology, 30-059 Krakow, Poland (e-mail: robert.wojcik@kt.agh.edu.pl).

Z. Duliński is with Faculty of Physics, Astronomy and Applied Computer Science, Jagiellonian University, 30-059 Krakow, Poland.

Digital Object Identifier 10.1109/LCOMM.2015.2439679

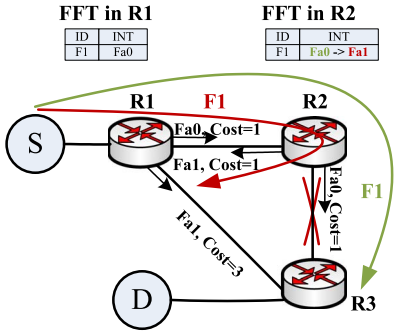

Fig. 1. FAMTAR failure recovery.

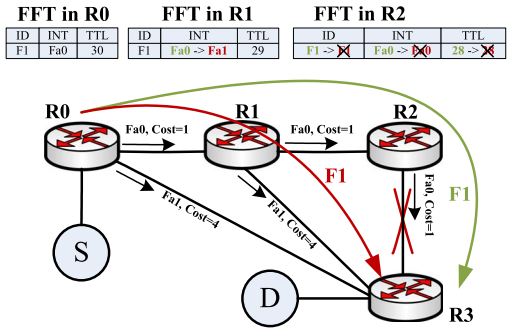

Fig. 2. FAMTAR TTL-based loop resolution mechanism.

More information about the FAMTAR mechanism can be found in [6] and [8]. However, these publications did not analyze the problem of failures and loops in the FAMTAR networks which is explained in the following section.

III. Failure-Based Loop Problem in FAMTAR

The FAMTAR mechanism ensures efficient and loop-free transmission in a stable network. One of the most important challenges of the FAMTAR approach is how to serve traffic when a link in the network fails.

In IP networks operated based on a dynamic routing protocol, such as Open Shortest Path First (OSPF) or Intermediate System to Intermediate System (IS-IS), the recovery from a failure is performed implicitly. After a link or node fails, the routers attached to this link inform all the other routers in the network about the noticed failure. The routing protocol is used to propagate the information about the topology change. As a result, routers change their routing tables to exclude paths with the failed link. The routing protocol selects new paths which are optimal at that moment and loop-free. Temporal loops may appear when the routing protocol is not in the converged state, for example, during updates. However, these usually last just for a few milliseconds and are immediately resolved when the routing protocol reaches convergence.

Unfortunately, this is problematic for FAMTAR. Here, routing table changes affect only new flows. Active flows, with identifiers registered to the FFT, are still routed according to their old routes stored in the FFT. As a result, in case of a failure, active flows can still be routed to the failed links, even after a routing protocol reaches convergence.

Therefore, when a FAMTAR router detects that it lost one of its neighbours, all flows forwarded to that neighbour must be deleted from the FFT. When new packets of these deleted flows arrive, they will be treated as belonging to new flows, and therefore, new FFT entries with the current routes will be created for them. However, with the mechanism described above, it is possible that paths from source (S) to destination (D) nodes will become permanently looped. Such a situation is presented in Fig. 1.

Suppose that traffic (flow F1) is sent from S to D. There are two possible paths: R1-R2-R3 (optimal with the cost of 2) and R1–R3 (with the cost of 3). As we can see, before the failure, packets of flow F1 are sent through interfaces Fa0 in R1 and Fa0 in R2 (green path). When the link between R2 and R3 goes down, R2 detects it because of lost carrier or neighbour adjacency. As a result, R2 changes the flow entry in the FFT. From that moment, packets of flow F1 will be sent through the interface Fa1 in R2 (red path). This means that packets of flow F1 will be sent back to R1. Unfortunately, based on the FFT in R1, they will be sent again to R2 through interface Fa0. The permanent loop which is created will last for as long as new packets of that flow will appear. It will be resolved only when the FFT entries for that flow will expire because of flow inactivity.

IV. TTL-Based Loop Resolution Mechanism

In this section, we propose a TTL-based mechanism to detect and resolve routing loops in FAMTAR networks. The key concept of this mechanism is to store the TTL value from a flow's first packet in the FFT. When a packet of the active flow arrives, its TTL value is compared to the one stored in the FFT. If they match, everything works as originally proposed in [6] and summarized in Section II.

After the first packet appears, the path for the corresponding flow is fixed and does not change over time. Therefore, unequal TTL values may indicate that a loop appeared in the network. If a router receives the packet with the TTL lower than the one stored in the FFT, it may suggest that this packet arrives at the router for the second time. In such a case, the FFT entry for the particular flow should be removed from the list and the packet should be routed as if there were no entry for its flow in the FFT, i.e., it should be treated as a new flow and forwarded according to the current routing table.

A. Advantages, Disadvantages and Costs

The main advantage of using the TTL-based mechanism is the fact that it protects the FAMTAR networks against the occurrence of failures. The mechanism also protects the network against loops caused by the lack of routing protocol convergence. Such loops are transient in standard IP networks, but they may become permanent in FAMTAR. This happens when a new flow is added to the FFTs during temporal lack of routing protocol convergence in the network. However, the TTL comparison resolves this problem as well.

One drawback of the proposed mechanism is a fact that paths created after the loop resolution may be suboptimal from the global point of view. Such an example is presented in Fig. 2. We can see that normally (without failure), packets of flow F1 are forwarded through path composed of R0, R1, R2, and R3

(green path). When the link between R2 and R3 fails, R1 has to find the path to R3 (red path). The direct link between R1 and R3 is chosen as from the R1's point of view, this is the optimal path to D. However, globally, the optimal path goes through R0 and R3, omits R1, and is one hop shorter. Although the creation of suboptimal paths is possible, it is not problematic. This is because even if the path is not optimal, it still avoids congested links which is guaranteed by FAMTAR. We reckon that it is better to use longer, but uncongested path, rather than a shorter one with congestion problems.

We assume that the TTL values for a particular flow do not vary on entering the domain, i.e., previous operator does not practise intra-flow multipath routing. To ensure that all incoming packets of a particular flow have the same TTL, an operator can optionally change the TTL values in a border router to the value registered in the FFT with the first packet.

The loop-related problem of FAMTAR is solved by the TTL-based mechanism at the cost of additional operations that need to be performed by the routers. Therefore, here we discuss its costs of operation. Because the TTL field is already present in every packet and processing it is a standard router's duty, only little new functionality is necessary. The only required action is to store the TTL value in the FFT and compare it for all packets. The amount of memory required is negligible (only 8-bit-long TTL value per flow) and the comparison processing is very quick (bit-by-bit XOR operation). Compared to other functions, such as, e.g., path computations, table entry lookups, etc., the addition of the TTL-mechanism is in whole different order of resource utilization.

V. Evaluation

To evaluate the proposal, the ns-2 simulator was used. We implemented a real network topology: the Polish national research and education network PIONIER [9]. Three cases were compared: the standard OSPF routing, FAMTAR without loop resolution mechanism and FAMTAR with the TTL-based mechanism. In case of FAMTAR without loop resolution mechanism, each router removes flow entries from the FFT after it discovers link failure (as described in Section III), but no loop resolution mechanism is used. In the experiments, we simulated random link failures with excessive quantity.

In each scenario, 1500 TCP or 1500 UDP (128 kbit/s) flows were generated. The volume was modelled using the Pareto distribution with the the mean flow size of 1 MB and the shape factor of 1.5. The packet size was set to 1000 bytes and the flows were started from the beginning of the simulation with the mean interval equal to 0.1 second. All links had the same capacity of 200 Mbit/s and propagation delay of 1 ms. Such volume of traffic did not saturate the existing links. It was done deliberately, in order to not trigger FAMTAR multipath transmission. Such an approach gives the possibility to compare the number of lost packets between legacy routing and FAMTAR scenarios.

In each interface, a FIFO queue with the buffer size of 500 packets was installed. The mean time between failures (MTBF) and mean time to repair (MTTR) parameters were used to generate up and down times independently for each link. The average for MTBF varied through simulations, whereas MTTR was fixed to 1 second. The simulation time was set to 250 seconds and the warm-up time was 30 seconds. Each simulation was repeated at least 10 times to get credible results. 95% confidence intervals were calculated based on the Student's t-distribution.

TABLE I
Mechanisms Efficiency Comparison. MTBF = 10 s

Scenario	Received Packets	Lost Packets
legacy routing	UDP: 817483 ± 104	29259 ± 77
	TCP: 2537712 ± 155949	5693 ± 565
FAMTAR	UDP: 152410 ± 115	694332 ± 130
	TCP: 2260067 ± 63781	15732 ± 557
FAMTAR with TTL	UDP: 817687 ± 105	29055 ± 79
	TCP: 2536693 ± 154349	5898 ± 601

Table I presents the number of received and lost packets for MTBF equal to 10 s. The number of lost packets corresponds to the amount of time the network was unable to provide proper communication. Given the size of the simulated network, such a low MTBF means that at any given time, there are failures somewhere in the network. This is unrealistic scenario, however, it shows the mechanisms performance well.

When legacy routing is used, there are no permanent loops in the network and the non-zero value of lost packets is caused by already queued packets in outgoing queues which are lost when link goes down. When FAMTAR is enabled, much more packets are lost. This is due to the occurrence of permanent loops. Adjacent routers remove flows from their FFTs when they detect a link failure, however other routers do not do that (Fig. 1). When the TTL-based loop resolution mechanism is used, the amount of lost packets is roughly the same as with legacy routing. That means that packets are not lost due to loops. Therefore, it shows that the proposed loop resolution mechanism works as expected.

UDP traffic is particularly good to demonstrate the mechanism because the flow rate of packets does not change after the failure has occurred. This means that the number of lost packets directly relates to the amount of time the connection was unavailable. TCP traffic behaves differently. Firstly, TCP does not have a controlled rate, which results in much greater (compared to UDP) number of packets generated and sent. Secondly, TCP stops its transmission when acknowledgments do not arrive. This is the reason for such a low number of packets lost. Because of that, the amount of lost packets for TCP is not a good indicator of the problem. Therefore, in further simulations, we narrowed our experiments to UDP only.

Fig. 3 compares the amount of retransmitted data and the mean number of hops that each packet must follow with respect to the varying MTBF. It can be observed that the amount of retransmitted data for legacy routing and FAMTAR with TTL-based mechanism is negligible. The amount of lost packets for classic FAMTAR is quite high and it grows with decreased MTBF. From the bottom plot, we can see that the mean number of hops that each packet follows slightly decreases when there are less failures in the network. Such an observation is natural. We can also observe that FAMTAR provides a bit higher average hop count, however, this was already documented in [6]. The TTL-based mechanism increases the hop-count even more, yet, still the absolute number is only marginally higher than for legacy routing. Therefore, this is a very small price to pay for the loop-free possibility of multipath transmissions.

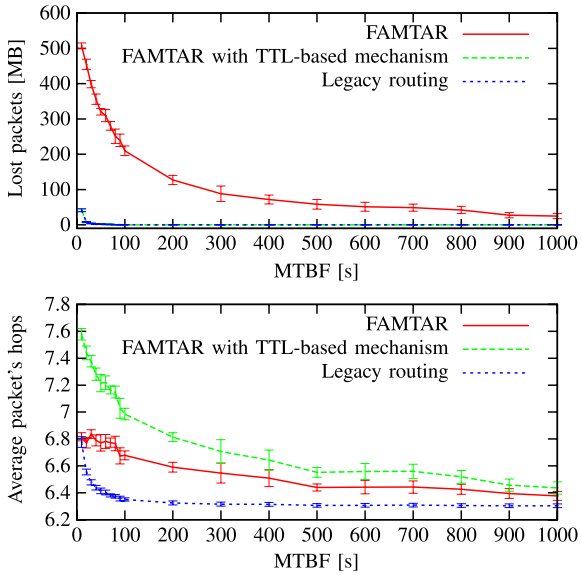

Fig. 3. Loop resolution efficiency with respect to network failure quantity.

TABLE II
MECHANISMS EFFECTIVENESS COMPARISON

Scenario	Packets Dropped
legacy routing	3841 ± 631
FAMTAR with TTL	4656 ± 48

VI. LABORATORY EXPERIMENTS

In this set of tests, we examined the proposal in a real network environment. We created a simple network with three possible paths between S and D. Each router in this network was a separate PC machine with multiple network interface cards, running our FAMTAR router software implementation. The test implementation is based on the Click Modular Router [10] suite. The OSPF routing is provided by XORP [11].

Two scenarios were examined. Each scenario was repeated five times to obtain credible results. In each, one UDP flow from host S to D was generated, using the D-ITG [12] traffic generator application. The packet size was set to a constant of 64 bytes and the flow generated 5555 packets per second which produced roughly 3 Mbit/s. The first scenario is a network with legacy OSPF routing. In the second, FAMTAR with TTL-based loop resolution mechanism was enabled. In each run, a single link failure was emulated. It was performed by physically unplugging a network cable from the currently used Ethernet interface. Using D-ITG, we measured the number of packets dropped during the failure recovery time. Results are shown in Table II.

In FAMTAR with the TTL-based mechanism we observe only slightly greater number of lost packets compared to legacy routing which does not result in any significant difference considering loop resolution. In case of legacy routing, our routers switched paths as soon as the XORP routing demon discovered the failure. The amount of time in which this action happened varied from experiment to experiment, hence large confidence interval. In FAMTAR, also the Click application must discover the failure and erase the corresponding entries from the FFT list. In our implementation, due to Linux kernel limitations, both actions, i.e. failure discovery by XORP and Click, use different event discovery mechanisms and therefore, happen independently. The latter takes a bit more time, but its time variations are smaller. We envision that in the hardware implementation both actions would be performed at the same time and there would be no difference between the legacy routing and FAMTAR's TTL-based mechanism.

VII. CONCLUSION

FAMTAR's major problem with rooting loops and failures is resolved by a TTL-based mechanism presented in this letter. The analysis shows that the mechanism is very efficient, does not interfere with any advantages provided by FAMTAR and completely eliminates the problem. On top of that, it requires little new functionality, the amount of memory required is negligible and the required processing power is low when related to other packet processing functions induced by FAMTAR.

ACKNOWLEDGMENT

The research was carried out with the support of the project "Flow-Aware Multi-Topology Adaptive Routing" founded by the National Centre for Research and Development in Poland.

REFERENCES

[1] J. Mcquillan, I. Richer, and E. Rosen, "The new routing algorithm for the ARPANET," *IEEE Trans. Commun.*, vol. 28, no. 5, pp. 711–719, May 1980.
[2] S. Low and P. Varaiya, "Stability of a class of dynamic routing protocols(IGRP)," in *Proc. IEEE Conf. Comput. Commun. INFOCOM*, 1993, vol. 2, pp. 610–616.
[3] D. Bertsekas, "Dynamic behavior of shortest path routing algorithms for communication networks," *IEEE Trans. Automat. Control*, vol. 27, no. 1, pp. 60–74, Feb. 1982.
[4] Z. Wang and J. Crowcroft, "Analysis of shortest-path routing algorithms in a dynamic network environment," *ACM Comput. Commun. Rev.*, vol. 22, no. 2, pp. 63–71, Apr. 1992.
[5] J. Domzal *et al.*, "A survey on methods to provide multipath transmission in wired packet networks," *Comput. Netw.*, vol. 77, pp. 18–41, Feb. 2015.
[6] R. Wojcik, J. Domzal, and Z. Dulinski, "Flow-aware multi-topology adaptive routing," *IEEE Commun. Lett.*, vol. 18, no. 9, pp. 1539–1542, Sep. 2014.
[7] R. Wójcik and A. Jajszczyk, "Flow oriented approaches to QoS assurance," *ACM Comput. Surv.*, vol. 44, no. 1, pp. 5:1–5:37, Jan. 2012.
[8] R. Wojcik, J. Domzal, Z. Dulinski, P. Gawlowicz, and D. Kowalczyk, "Performance evaluation of flow-aware multi-topology adaptive routing," in *Proc. IEEE CQR Int. Workshop*, Tucson, AZ, USA, May 2014, pp. 1–5.
[9] PIONIER Network. [Online]. Available: http://www.pionier.net.pl/online/en/projects/
[10] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The click modular router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, 2000.
[11] M. Handley, E. Kohler, A. Ghosh, O. Hodson, and P. Radoslavov, "Designing extensible IP router software," in *Proc. Symp. Netw. Syst. Des. Implement.*, 2005, pp. 189–202.
[12] A. Botta, A. Dainotti, and A. Pescapè, "A tool for the generation of realistic network workload for emerging networking scenarios," *Comput. Netw.*, vol. 56, no. 15, pp. 3531–3547, Oct. 2012.