Analytical Modeling Approach of Routing Deflection for Intra-domain Networks

Rodolfo R. Gomes[1], Cristina K. Dominicini[1], Alextian B. Liberato[1]
Moisés R.N. Ribeiro[1], Magnos Martinello[1]

[1]Software Defined Networks Research Group (NERDS)
Federal University of Espírito Santo, Vitoria/ES, Brazil
rodolfo@ifes.edu.br, cristina.dominicini@ifes.edu.br
alextian@ifes.edu.br, moises@ele.ufes.br, magnos@inf.ufes.br

Abstract. *This paper presents an analytical modeling to* ***KAR (Key-for-Any-Route)****, an intra-domain resilient routing system in which edge-nodes set a route ID to select any existing route as an alternative to safely forward packets to their destination. KAR-enabled switches explore the existing routes by using special properties of Residue Number System as encoding and forwarding techniques. The encoding technique allows adding resilient forwarding paths to drive deflected packets (due to link failure) to their original destination so that loops are not formed. Three deflection methods are discussed along with their analytical models checked against numerical simulations. Results show that KAR efficiently allows deflected packets to automatically reach their destination.*

1. Introduction

Routing is an important feature in computer networking that selects a path between two nodes to enable their communication. The route can be determined within the network by each node or be entirely computed by ingress edge nodes, also known as source-routing (SR). Although non-SR protocols tend to be scalable, SR networks facilitate traffic engineering because they offer choice of routes for the sources to select the path in a network-wide view with undesirable characteristics [Yang and Wetherall 2006]. Thus, SR technique is the basis of many proposals to improve the reliability and performance of networks, as a promising approach to improve flexibility of the network layer in future Internet architectures.

Source routing has an important issue regarding resilience: its reaction speed to failures of a link or node that belongs to a path. As soon as a core node becomes aware of a link failure, it sends a failure notification to the edge nodes. The source node can, then, select another path that does not include the faulty link after the new topology converge. Even if an alternative path has been pre-selected as a protection (to avoid the delay imposed by convergence), the source node must wait to receive the notification message and the core node must wait the new path signaling. Until that, packets that had already left the source node are dropped.

In order to avoid this packet dropping, another failure reaction can happen within the network. Thus, a node can locally switch to the alternative path as soon as the node detects a failure on one of its directly connected links affecting that path. However, this approach requires that every node/switch should be able to compute and/or store the backup paths, so that there is a dependency between each switch forwarding table and the topology of the entire network. This is normally done by pre-signaling the protection paths to switches, as [Swallow et al. 2005] does, but it tends to cause an initial delay before packets start to flow. Another way is embedding extra paths in packet header, increasing its

length, as it is done in [Nguyen et al. 2011]. Moreover, this process often requires extra packet processing, such as changes on packet header, at each switch while the packet travels its route. Lastly, deflection routing is another approach for failure reaction. It may be done by a core node randomly selecting an outgoing port and relying on the adjacent node to deliver the packets to their destination. In this case, there is no protection signaling and no state requirement to core nodes. However, the packets would follow a random walk and create transient loops [Yang and Wetherall 2006].

We propose **KAR (Key-for-Any-Route)**, that is a new intra-domain routing system with a novel fast failure reaction mechanism which combines the benefits of source-controlled routing with driven deflections to provide network routing resiliency. A path in a KAR network is represented by a number, the Route ID, inserted into the packet header. The next-hop decision relies on the remainder of the division between the Route ID and the switch ID along the path(s). The core nodes are simple stateless forwarding nodes. As for the fast failure reaction mechanism, these core nodes randomly deflects packets from the faulty link instead of dropping them. Those packets then follow by a diverse set of switches by using driven deflection forwarding paths embedded in the route ID due to the encoding technique exploit properties from Residue Number System (RNS) [Garner 1959]. Thus, KAR approach addresses link failures keeping the network connectivity and allowing in-flight packets along the failed path to reach their destination with a small route stretch.

An analytic model is proposed to evaluate the effect of three routing deflection techniques on the number of hops taken to reach destination. The model is validated in a discrete event simulator.

2. KAR Resilient Routing System Design

As a source-routed networking, KAR has two main components: the edge nodes, which are responsible for routing, and the core nodes. In this work, the latter forward packets based on their unique identifier (Switch ID) and the Route ID, a packet header field. They do not need any route and protection signaling.

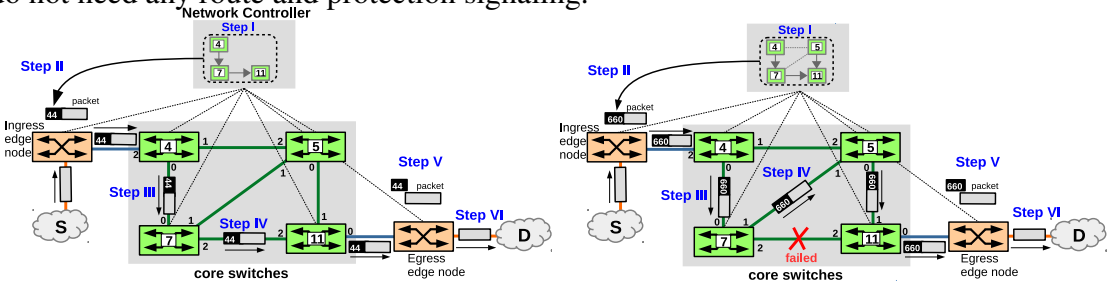


Figure 1. KAR design: (a) routing system based on shortest path, (b) fast failure reaction with driven deflection forwarding paths.

Fig. 1 illustrates an overview of the concept of *KAR* design with 6-node network. When the conventional node S in Fig. 1(a) wishes to communicate with another conventional node D, the ingress edge node selects an end-to-end path by using a routing algorithm (e.g. shortest path). Suppose it chooses the set of switches $S = \{4, 7, 11\}$ as the primary path. These IDs together with their respective output switch interfaces $P = \{0, 2, 0\}$ allows computing the *Route ID*, e.g. $R = 44$ (**Step I**) that should be assigned to the header of the incoming flow packets by the node itself (**Step II**). KAR *route ID* computation exploits RNS properties, and will be latter explained (section 2.2).

By computing the remainder of the division (denoted as $< a >_b \equiv a \; modulo \; b$) between the *route ID* ($R = 44$) and the *Switch ID*, the core switches know the output port to send packets to. Consequently, switch_ID 4 (SW4) forwards packets with route ID 44 to port $< 44 >_4 = 0$ (**Step III**). In its turn, SW7 forwards them to the port $< 44 >_7 = 2$ (**Step IV**). Finally, SW11 forwards them to port $< 44 >_{11} = 0$ (**Step V**). They then reach the egress edge node, that removes the *route ID* from the packet header (**Step VI**) and delivers them to node D.

A link failure makes the adjacent core switches notify the edge nodes, which recalculate the route ID ignoring the faulty link in their network view. In the meanwhile, all packets sent by the source before the route ID modification would be lost. In order to avoid packet loss (*Hitless* property), KAR uses a deflection routing approach. As for this fast failure reaction, it chooses randomly one of its healthy ports to forward the packets to. Although deflection routing may form transient loops [Yang and Wetherall 2006], KAR provides a guarantee of loop-free routing even in the event of a link failure based on its *Driven Deflection* property. To this end, it is necessary to compose protection paths that are responsible for driving deflected packets to the destination by adding new nodes in the computation of the route ID. Fig. 1(b) illustrates this concept, by including proactively SW5, in (Step I), in the route ID as a protection path that delivers the deflected packets to SW11 when a link failure happens (resulting in $R = 660$). Consider, also, that the selected deflection technique chooses randomly between the available ports when a link fails. Thus, when link SW7-SW11 fails, SW7 forwards packets randomly to port 0 (SW4) or 1 (SW5). Those packets sent to SW4 bounce back to SW7 and suffer another deflection. However, all the packets that reach SW5 by deflection in SW7 (Step IV) will be forwarded to SW11 ($< 660 >_5 = 0$) and, in this way, arrive at the destination.

2.1. Deflection Techniques

We propose the following three deflection routing techniques in order to make deflected packets reach the destination by the *Driven Deflections* property or even by chance.

Hot-Potato (HP): once a packet is deflected, its remaining route is completely random.

Any Valid Port (AVP): only when the output port computation does not represent a valid port ID (it does not exist or it is not available), the core node selects at random an active port and send packet to it.

Not the Input Port (NIP): the AVP method is changed by excluding the input port from the set of next-hop candidates even when the computation tells to send the packet back to it. This rule is broken only if the only available port is the ingress one. Besides generating less random paths, it avoids routing loops between two nodes.

There is a good reason to propose AVP or NIP: after deflection, a packet may arrive at a node that leads to the destination path. From there, it will follow the computed path once again. Note that in Fig. 1(a), without any *Driven Deflection* forwarding paths, a packet arriving at SW5 has 50% probability to go to SW11. In contrast, the addition of SW5 in the route ID and the use of NIP deflection technique cause all the packets to be driven through this forwarding path (SW5→SW11).

A final remark in terms of modeling is that an edge node can receive a packet not addressed to it. In this case, the node can choose between two approaches: it directly returns the packet to the network without any change or it queries the controller for a new route ID (path between the edge node and the destination) before returning the packet to the network. In our tests, we considered this second approach.

2.2. Encoding the Forwarding Paths

The generation of Route ID from Switch IDs and output port indexes is based on the Residue Number System, which is described in this section.

Let S be a set of modulo $S = \{s_1, s_2, \ldots, s_N\}$ of the N switch IDs on the desired path, in which all elements of the set are pairwise coprimes numbers. Let P be a set of outgoing ports $P = \{p_1, p_2, \ldots, p_N\}$, where p_i is the outgoing port index for the packet at the switch s_i.

Let M be

$$M = \prod_{i=1}^{S} s_i \tag{1}$$

A number $R \in \mathbb{N} | 0 \leq R < M$ can be represented by a residue set given a modulo set:

$$R \xrightarrow{RNS} \{p_1, p_2, \ldots, p_N\}_S \tag{2}$$

, where

$$p_i \ = \ R \, modulo \, s_i \tag{3}$$

The Chinese Remainder Theorem [Ding et al. 1996] states that is possible to reconstruct R through its residues in a RNS. This can be done as follows:

$$R =< \sum_{i=1}^{S} p_i \cdot M_i \cdot L_i >_M \tag{4}$$

where

$$< a >_b \ \equiv \ a \, modulo \, b \tag{5}$$

$$M_i \ = \ \frac{M}{s_i} \tag{6}$$

$$L_i \ = \ < M_i^{-1} >_{s_i} \tag{7}$$

Eq. (7) means that L_i is the modular multiplicative inverse of M_i. In other words, L_i is an integer number such that

$$< L_i \cdot M_i >_{s_i} = 1 \tag{8}$$

Returning to the example of this section, the computation of route ID from S to D is obtained as follows:

Single forwarding path	**Path protected by driven deflections**
$switches = \{s_1, s_2, s_3\} = \{4, 7, 11\}$	$switches = \{4, 7, 11, 5\}$
$ports = \{p_1, p_2, p_3\} = \{0, 2, 0\}$	$ports = \{0, 2, 0, 0\}$
$M = 4 \cdot 7 \cdot 11 = 308$	$M = 4 \cdot 7 \cdot 11 \cdot 5 = 1540$
$M_1 = 77, M_2 = 44, M_3 = 28$	$M_1 = 385, M_2 = 220, M_3 = 140, M_4 = 308$
$L_1 =< M_1^{-1} >_{s_1} =< 77^{-1} >_4 = 1$	$L_1 =< 385^{-1} >_4 = 1$
$L_2 =< 44^{-1} >_7 = 4$	$L_2 =< 220^{-1} >_7 = 5$
$L_3 =< 28^{-1} >_{11} = 2$	$L_3 =< 140^{-1} >_{11} = 7$
$R =< L_1 \cdot M_1 \cdot p_1 + L_2 \cdot M_2 \cdot p_2 + L_3 \cdot M_3 \cdot p_3 >_M$	$L_4 =< 308^{-1} >_5 = 2$
$R =< 0 + 352 + 0 >_{308} = 44$	$R =< 0 + 2200 + 0 + 0 >_{1540} = 660$

It can be noticed in Eq. (4) that the Route ID does not store or keep the information about the switch sequence the packet would travel along. Data from each switch (switch

ID and port index) belong to its own addend of the summation and does not influence the other summation addends. As the finite summation is commutative, the switch order is irrelevant to derive the route ID. This property allows embedding, in the route ID, extra switches that are disjoint of the desired route. This is the fundamental concept of the Driven Deflection Forwarding Paths, and it is useful to protect a desired route when a packet is deflected due to a faulty link since it is possible to guide deflected packets to destination via path segments.

3. Analytic Modeling of Deflection Routing

The model presented in this paper is designed to understand the random effect of the deflection techniques HP, AVP and NIP. The basis of the analytic model is Markov chain with absorbing states, as it can be used to trace the random walk of a test packet through a network under deflection routing [Forghieri et al. 1995].

Let N be number of nodes of a given network and Π be the NxN transition matrix whose elements π_{ij} represent the probability of a *packet* move to node *i* at its *(k+1)*th hop, being at node *j* at its *k*th hop. This is a static information for a given (static) topology. The vector **p**(k) represents the probability distribution of the test packet at *k*th hop. That is, its $p_i(k)$ elements represent the probability of the packet be at *i*th node at its *k*th hop. Therefore, it yields

$$\mathbf{p}(k+1) = \Pi \cdot \mathbf{p}(k) \tag{9}$$

and

$$p_i(0) = \begin{cases} 1 & \text{, if node } i \text{ is source node} \\ 0 & \text{, otherwise} \end{cases} \tag{10}$$

If node d is the destination node, it retains the received packets. As packets cannot go anywhere else, their next hop is their current node.

$$\pi_{id} = \begin{cases} 1 & \text{, if node } i \text{ is the destination node d} \\ 0 & \text{, otherwise} \end{cases} \tag{11}$$

Finally, let n_j be the number of active ports of *j*th switch.

Each iteration described by Eq. (9) represents a hop and the interations continue until the destination node has 100% of packages. In this way, it is possible to find the cumulative distribution function of the number of hops needed to reach the destination node from source node, by recording the state vector at each step.

The rest of this section describes how to fill the transition matrix and the state vector for each deflection technique.

Under **HP deflection**, the probability of a packet going to node i from node j is $\pi_{ij} = 1/n_j$, if *j* has a direct active link with *i*, or zero otherwise. For example, for the network of Figure 1, the resulting matrix Π is

$$\Pi_{HP} = \begin{bmatrix} 0 & 1/3 & 1/3 & 0 & 1 & 0 \\ 1/3 & 0 & 1/3 & 1/3 & 0 & 0 \\ 1/3 & 1/3 & 0 & 1/3 & 0 & 0 \\ 0 & 1/3 & 1/3 & 0 & 0 & 0 \\ 1/3 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1/3 & 0 & 1 \end{bmatrix} \tag{12}$$

The first column represents the next hop probability for packets at first node, SW4. As it has three ports, a packet has 33.33% chance to be sent to SW5, to SW7 or back to

S. The last column represents the destination D, and that is why it has 100% of packets going to itself.

The source node for the analysis of HP, unlike the further discussed techniques, cannot be the real one but the node attached to the faulty link. Otherwise, the test packet would not follow the right path until the faulty link, as the probabilities of the traversed path previous to the faulty link were 100% until failure point: it was not marked as a diverted packet yet. So, the computed **p**(k) actually represents the state after *k+h* hops, where *h* is the number of hops a packet does to reach the failing link on the path from the ingress node.

As for the **AVP**, the state transition matrix Π depends also on the route ID and the switch IDs. When $< routeID >_{switchID}$ matches an active port index, this should become the only possible outgoing port. As a result, the element in the row representing the output port is 1.0; others are 0. If $< routeID >_{switchID}$ does not match an available port, the switch *j* randomly forwards it to any of its active ports. So, $\pi_{ij} = 1/n_j$ if *j* has a direct active link with *i*, and zero otherwise.

Adopting the route ID 44 computed in section 2.2 for the route S-SW4-SW7-SW11-D, and performing the modulo operation $< routeID >_{switchID}$, we find out the output port for each node.

Switches SW4 and SW11 have valid computed ports. Switch 5 has no port numbered as 4 ($< 44 >_5 = 4$). In such case, the packet destination is random. Hence, the second column of Π is equal to the HP matrix transition. As D is the destination node, its column keeps unchanged compared to Π_{HP}. When analyzing the failure in link SW7-SW11, SW7 has only two available ports: back to SW4 or deflect to SW5. This can be seen in Eq. (13) in the third column of the following matrix.

$$\Pi_{AVP} = \begin{bmatrix} 0 & 1/3 & 1/2 & 0 & 1 & 0 \\ 1 & 0 & 1/2 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 1/3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix} \tag{13}$$

For NIP, the state vector **p**(k) now has to account for the fact that a given switch bearing the *test packet* at *k*th hop must not receive it back at *k+2*th hop. This requires the **p**(k) to encompass N^2 states, instead of just N, in order to trace the previous node the test packet has visited. Consequently, the transition matrix Π grows now to $N^2 \times N^2$. The first element of the state vector, $p_1(k)$, represents the probability of the test packet being in the first node coming from itself at *k*th hop. The second element, in its turn, represents a packet in node 1 coming from node 2, and so on until the *N*th element. The value of $p_{N+1}(k)$ ($p_5(k)$ in our example) is the probability, at *k*th hop, of the packet being in second node coming from the first one. Figure 2 illustrates a hypothetical $p(k)$ value for our network example. At a kth hop, node 1 (SW4) has 20% of packets, and they were all at node 4 (D) in previous hop. In its turn, node 2 (SW5) has 40% of packets at this state: 30% came from node 1 (SW4) and 10% from node 3 (SW7). Hence, the matrix Π must be created avoiding the chance of the traveling packet going back to its previous state when the hot-potato is performed only.

The state transition probability matrix is now expanded to 36x36 in order to sup-

port this previous-hop information. The matrix is not shown due to space limitation.

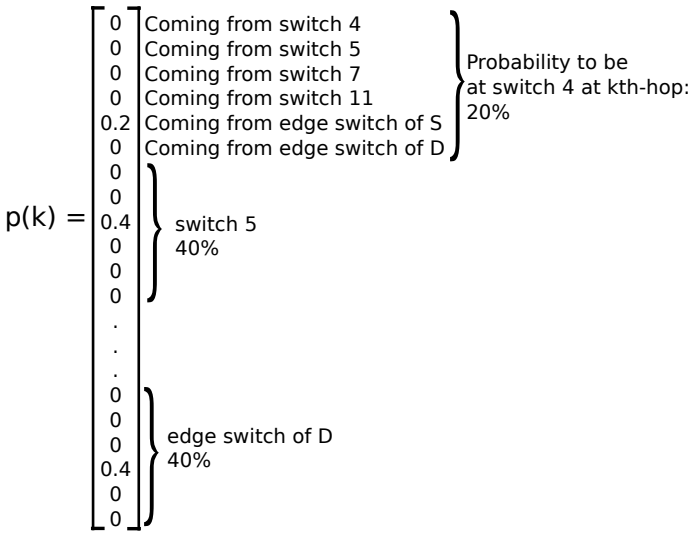
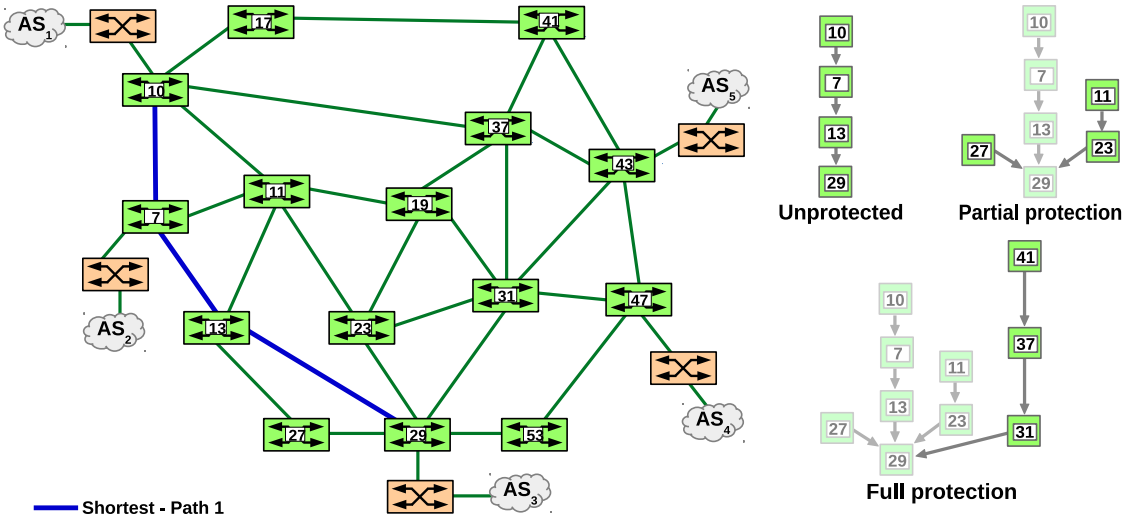


Figure 2. Example of state vector content at hypothetical kth hop for NIP

4. Evaluation

The KAR investigation considers one general topology[1] (Fig. 3(a)) with a 15-node network to represent an experimental scenario to illustrate the main insights of KAR routing deflection approach.



(a) (b)

Figure 3. KAR Analysis: (a) topologies with 15-node network, (b) protection mechanism for resilient routing.

Our evaluation methodology is structured in two parts. The first part is related to the validation of our deflection routing model. The second analysis considers the number of hops taken by packets to reach their destination as the performance metric. This study investigates the impact of driven deflection forwarding paths as the network protection mechanism to ensure loop-free (safety) condition.

[1] A common range for network topologies http://www.topology-zoo.org/

4.1. Deflection Routing Model Validation

In order to validate the analytic model, the network scenario of Fig. 3(a) has been simulated using OMNeT++ V3.0, a discrete event simulator. Simple custom modules were created to simulate KAR nodes using the proposed deflection techniques (HP, AVP and NIP). After each single link failure, 1,000,000 packets are sent by the same route without protection. The number of hops taken by each packet is evaluated.

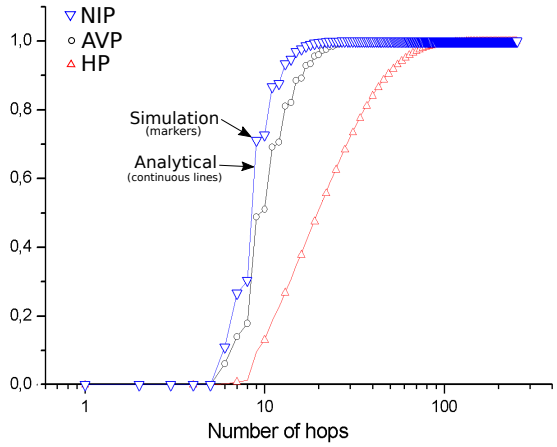


Figure 4. Cumulative Distribution Function (CDF) of the number of hops (Model versus Simulation).

The diameter of the topology Fig. 3(a) was selected (from AS_2 to AS_4 via $7, 11, 19, 31, 47$) as the route under analysis. Fig. 4 shows a comparison between analytical and simulation approaches. It presents Cumulative Distribution Functions for both of them, considering the number of hops to reach the destination as a performance metric.

More specifically, Fig. 4 is related to the failure at link (SW11-SW19) where the continuous lines represent the analytical values and the markers (triangles and circles) identify the results obtained by simulation. As it can be seen, the analytical model approach matches precisely the simulation results for the three deflections techniques. The differences between them were lower than 10^{-3} for all cases. It is worth to mention that all other failures at the selected path (diameter) had accuracy at the same order of magnitude.

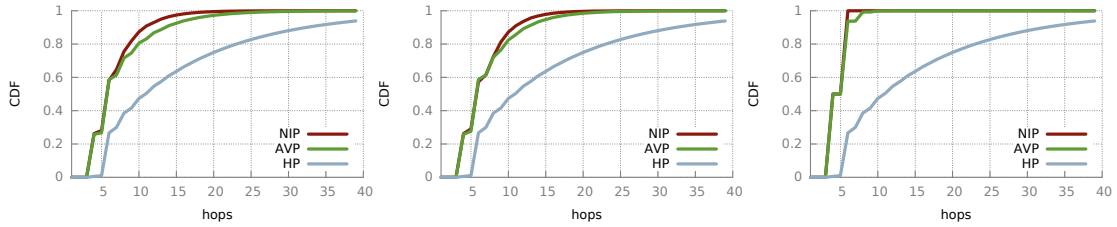
4.2. Analysis of KAR Resilient Routing System

Let us assume that the shortest path of Fig. 3(a) (Path 1) is selected to allow a communication between AS_1 and AS_3. If a failure happens at the link (SW10-SW7), then the NIP deflection technique chooses one link uniformly among 3 links to forward packets to switches SW17, SW37 and SW11. Although it implies that packets are not lost, they are likely to be disordered at destination or even to enter in a loop. Thus, a partial protection for KAR resilient routing can be built adding switches SW11 and SW23 to the route ID (Fig. 3(b)), so that packets deflected to SW11 with $1/3$ of probability will be all driven to their destination. In the case of failures at links (SW7-SW13) or (SW13-SW29), they are protected by this tree branch (SW11-SW13). Adding SW27 allows to fully protect the failure near the destination SW29. Therefore, in case of failure at link (SW13-SW29), half of packets is forwarded to SW11 with one more hop than the other half sent to SW27.

Besides, there is still $2/3$ of packets that will be sent to switches SW17 or SW37 in a failure of link (SW10-SW7) close to the source. In order to support the highest resilience for such a route (from AS_1 to AS_3), a full protection route can be ensured by including a branch coming from SW41, e.g. SW41-SW37-SW31, see Fig. 3(b).

In order to evaluate the performance of KAR driven deflections and understand the contribution of the different protection mechanisms and deflection techniques proposed, we considered each individual possible link failure in the shortest path 1 from AS_1 to AS_3 and generated the CDF curves that outline the probability that packets will arrive in the destination within a value less than or equal to a given number of hops. Fig. 5, 6 and 7 consider the selection of shortest **Path 1** (10; 7; 13; 29).

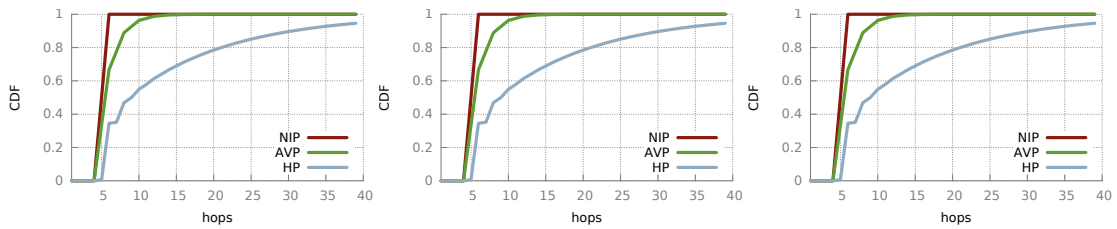
Fig. 5 presents **Path 1** results for a failure in SW10-SW7 link considering Unprotected (Fig. 5(a)), Partial Protection (Fig. 5(b)) and Full Protection (Fig. 5(c)) mechanisms. Regarding protection mechanisms using AVP or NIP deflection techniques, Full Protection curve is much steeper when compared to Unprotected and Partial Protection mechanisms. For instance, considering 99th percentile, Full Protection takes 7 hops for packet delivery while Unprotected and Partial Protection mechanisms take 19 hops. To understand NIP behaviour in this case, consider that when the failure in SW10-SW7 happens there are three possible next hops (SW11, SW17, SW37) with equal probability ($1/3$). However, for Partial Protection, only SW11 hop is included in the defined alternative paths, while for Full Protection all three possible hops lead to protected paths.



(a) (b) (c)

Figure 5. CDF of number of hops to reach AS_3 from AS_1 using shortest Path 1 - varying deflections techniques (failure at link SW10-SW7): (a) Unprotected, (b) Partial Protection, (c) Full Protection.

Fig. 6 presents **Path 1** results for a failure in SW7-SW13 link considering Unprotected (Fig. 6(a)), Partial Protection (Fig. 6(b)) and Full Protection (Fig. 6(c)) mechanisms. This figure depicts no difference in the application of the different protection mechanisms, even in the case of Unprotected mechanism. Regarding deflection techniques, it is possible to note that NIP took about half the number of nodes to reach probability 1 when compared to AVP. Partial and Full Protection have similar behavior, because when the failure in SW7-SW13 happens there are two possible next hops (SW11, AS_2) with equal probability ($1/2$) and, in both cases, SW11 hop is included in the defined alternative paths while AS_2 returns the packet to SW7.



(a) (b) (c)

Figure 6. CDF of number of hops to reach AS_3 from AS_1 using shortest Path 1 - varying deflections techniques (failure at link SW7-SW13): (a) Unprotected, (b) Partial Protection, (c) Full Protection.

Fig. 7 presents **Path 1** results for a failure in SW13-SW29 link considering Unprotected (Fig. 7(a)), Partial Protection (Fig. 7(b)) and Full Protection (Fig. 7(c)) mechanisms. This figure depicts almost no difference in the application of the Full and Partial Protection mechanisms, but shows a much slower convergence for the Unprotected case. This performance can be explained because the point of failure is fully surrounded by nodes that are part of the protection path. Regarding deflection techniques, it is possible to note that NIP took about half the number of nodes to reach probability 1 when compared to AVP for Full and Partial Protection, but this difference between NIP and AVP is smaller when using Unprotected Protection. This difference is justified by the fact that AVP allows the packet to return to the previous hop (SW13) while all NIP options guide the packet in the direction of the destination.

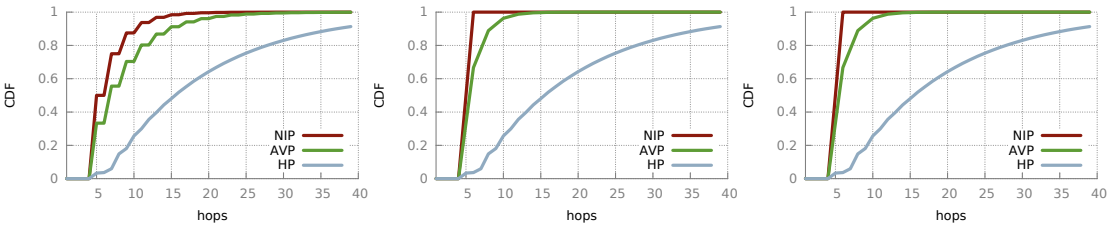


Figure 7. CDF of number of hops to reach node AS_1 from AS_3 using shortest Path 1 - varying deflections techniques (failure at link SW13-SW29): (a) Unprotected, (b) Partial Protection, (c) Full Protection.

For all the results presented in this section, it is important to observe that HP deflection technique increases substantially the number of hops, when compared to AVP and NIP (more than 40 hops). Therefore, the results from the application of HP technique serves as baseline to outline the gains from NIP and AVP techniques in each scenario.

5. Related Work

There has been much work on failure reaction within a network. The most closely related works include MPLS Fast Reroute [Swallow et al. 2005], SafeGuard [Li et al. 2009], and OpenFlow 1.3 Fast Failover [Sharma et al. 2012]. The common part among these proposals is the precomputation of alternative paths to each destination for intra-domain routing, so a router can locally switch to the alternative path without waiting for a topology convergence process. However, these approaches require network states stored at the switches tables (statefull) and lack the flexibility of source-controlled routing. In the case of MPLS Fast Reroute, it still requires the support of a signaling protocol such as Label Distribution Protocol (LDP) for MPLS-enabled switches.

Although source-controlled routing is not in mainstream use of the Internet today, perhaps because source routes do not fit the Internet model in which ISPs set routing policy based on destination addresses, this approach has inspired many innovative proposals for future Internet architectures [Yang and Wetherall 2006, Motiwala et al. 2008, Nguyen et al. 2011, Ramos et al. 2013, Martinello et al. 2014]. Among the main reasons to revisit this approach are [Lee et al. 2015]: i) The data plane becomes simpler because core nodes perform very simple forwarding operations; ii) Traffic engineering is more flexible, allowing application-optimized path selection at the source; iii) Routing stability is improved (e.g. no transient loops) since the path computation is centralized at the source.

Slick Packets [Nguyen et al. 2011] and SlickFlow [Ramos et al. 2013] were proposed to achieve fast data plane failure reaction by embedding alternative routes within the packet headers at the source. The idea is to represent the paths as a sequence of segments that will be used by each switch (or router) to perform the forwarding operation. Also, [Yang and Wetherall 2006] and [Motiwala et al. 2008], both use path label bits set by the source to pseudo-randomly select a next hop at each router or AS. In [Yang and Wetherall 2006], pseudo-random forwarding can lead to forwarding loops. In [Motiwala et al. 2008] routers follow certain rules that ensure loop-free, but reduce path diversity. In contrast to previous works, KAR network core is stateless and its forwarding strategy does not depend on the network topology. The second important difference is related to the driven deflection forwarding paths as the resilient routing mechanism for network protection. Rather than define the complete protection path, only small parts of a path can be included or even a unique node can be added to the route ID. This gives more flexibility, keeps the network core fast and simple and does not increase the route ID size.

RNS-based forwarding is used by [Martinello et al. 2014]. However, KAR routing system has focused on resilient routing that is not taken into account in KeyFlow proposal. In contrast, KAR advances the state of art by dealing with failed links by using routing deflections. In addition, one of the main contributions of KAR is our analytical model approach for evaluating deflection routing. This model was validated against simulation results and allowed us to obtain probabilistic insights on different methods of deflection.

6. Conclusion and Future Work

This paper proposed the analytical modeling of Key-for-Any-Route (KAR), that is a novel fast failure reaction scheme to avoid packet loss and improve resiliency for intra-domain routing systems. Our proposal combines a source routing technique based on the *Residue Number System* (RNS) with *Driven Deflections* property in order to enable efficient routing even in the event of a link failure. This network routing strategy uses stateless core switches, which provide high forwarding performance with the use of simple, low-cost switches. It is a resilient routing scheme because the protection paths enable the packet delivery after a failure through loop-free alternative paths without any reconfiguration on the network nodes.

Analytical models were proposed for three deflection techniques (HP, AVP, NIP) and checked against numerical simulations for different protection mechanisms (Unprotected, Partial Protection, Full Protection). Results show that KAR efficiently allows deflected packets to automatically reach their destination and that NIP and AVP techniques presented substantial performance improvements when compared with a lower bound classical HP technique. NIP deflection with fully protected path presents itself as the best combination in terms of failure reaction.

Finally, mainly when using NIP deflection, KAR poses as a fast failure reaction scheme with small stretch, which is the ratio of the number of hops to achieve the destination after a failure to the original shortest path. In the analyzed scenarios, the addition of protection paths reduces the stretch by at least half.

As future work, we plan to explore the use of multiple paths in the case of redundant links and to investigate the application of KAR in the service chaining of virtualized network functions. Furthermore, we intend to extend the conclusions of the analytical model to enable the prediction of traffic engineering information that could be used to

define more efficient routes.

Acknowledgments

This work has received funding from CNPq, CAPES and FAPES. In addition, it is part of the FUTEBOL project, which has received funding from the European Union's Horizon 2020 for research, technological development, and demonstration under grant agreement no. 688941 (FUTEBOL), as well from the Brazilian Ministry of Science, Technology and Innovation (MCTI) through RNP and CTIC.

References

Ding, C., Pei, D., and Salomaa, A. (1996). *Chinese Remainder Theorem: Applications in Computing, Coding, Cryptography*. World Scientific Publishing Co., Inc., River Edge, NJ, USA.

Forghieri, F., Bononi, A., and R.Prucnal, P. (1995). Analysis and comparison of hot-potato and single-buffer deflection routing in very high bit rate optical mesh networks. *IEEE Transactions on Communications*.

Garner, H. L. (1959). The residue number system. *Transactions on Electronic Computers*, pages 140 – 147.

Lee, T., Pappas, C., Basescu, C., Han, J., Hoefler, T., and Perrig, A. (2015). Source-based path selection: The data plane perspective. In *The 10th International Conference on Future Internet*, CFI '15, pages 41–45, New York, NY, USA. ACM.

Li, A., Yang, X., and Wetherall, D. (2009). Safeguard: safe forwarding during route changes. In *Proceedings of the 5th international conference on Emerging networking experiments and technologies*, CoNEXT '09, pages 301–312, New York, NY, USA. ACM.

Martinello, M., Ribeiro, M., de Oliveira, R., and de Angelis Vitoi, R. (2014). Keyflow: a prototype for evolving sdn toward core network fabrics. *Network, IEEE*, 28(2):12–19.

Motiwala, M., Elmore, M., Feamster, N., and Vempala, S. (2008). Path splicing. *SIGCOMM Comput. Commun. Rev.*, 38(4):27–38.

Nguyen, G. T., Agarwal, R., Liu, J., Caesar, M., Godfrey, P. B., and Shenker, S. (2011). Slick packets. *SIGMETRICS Perform. Eval. Rev.*, 39(1):205–216.

Ramos, R. M., Martinello, M., and Rothenberg, C. E. (2013). Slickflow: Resilient source routing in data center networks unlocked by openflow. *IEEE Conference on Local Computer Networks*, pages 01–08.

Sharma, S., Staessens, D., Colle, D., Pickavet, M., and Demeester, P. (2012). A Demonstration of Fast Failure Recovery in Software Defined Networking. *TRIDENTCOM 2012*, pages 411–414.

Swallow, G., Pan, P., and Atlas, A. (2005). RSVP-TE fast reroute. RFC 4090, http://www.ietf.org/rfc/rfc4090.txt.

Yang, X. and Wetherall, D. (2006). Source selectable path diversity via routing deflections. *SIGCOMM Comput. Commun. Rev.*, 36(4):159–170.