Towards Disruption-Free Intra-Domain Routing

Ang Li   Xiaowei Yang
University of California, Irvine
{angl, xwy}@uci.edu

David Wetherall
Intel Research Seattle
and University of Washington
djw@cs.washington.edu

ABSTRACT

We present the basic design of Cost-Carrying Packets (CCP), a novel routing framework to achieve disruption-free intra-domain routing. Under CCP, each packet carries a short label to indicate the remaining path cost to reach its destination. Downstream routers use path costs carried by packets and their locally computed path costs to detect inconsistent forwarding paths and repair the inconsistencies using pre-computed paths. The preliminary simulation results suggest that CCP can shorten the service disruption time to be commensurate with the failure detection time without delaying routing convergence.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing Protocols*

General Terms

Algorithms, Design, Reliability

Keywords

Network failures, routing loops, fast recovery

1. INTRODUCTION

Emerging real-time applications demand negligible service disruption times on IP networks. Even sub-second periods of loss or delay can adversely impact the users of these applications. Yet IP forwarding combined with the OSPF or IS-IS routing protocols in use today can easily produce noticeable periods of disruption following the failure or restoration of equipment: routing convergence may take several seconds, and during this period, forwarding may be disrupted because of invalid routes or temporary micro-loops.

The demand for near-zero disruption times has driven much work on IP fast failure recovery in the past few years. One approach is to speed up the routing convergence process and reduce the transition time. Unfortunately, recent work shows that the routing transition time can be shortened to sub-second periods, but no less due to fundamental limits such as speed of light and memory update latency. This intrinsic limit has motivated another area of work: MPLS fast-reroute and IP fast-reroute (IPFRR). Work in this area aims to reduce packet disruption times during routing transitions by rapidly repairing failures at routers adjacent to them
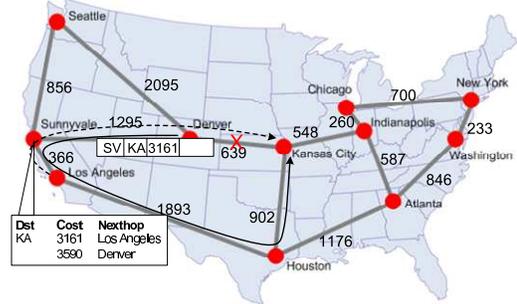
Figure 1: Forwarding using cost-carrying packets in the Abilene network.

and by preventing micro-loops during routing transitions. Unfortunately, local repair mechanisms do not prevent micro-loops during routing transitions [1, 6]. Similarly, micro-loop mitigation mechanisms extend the duration of the routing convergence process [2, 4], during which traffic may still be disrupted. Recently a mechanism called Failure Carrying Packets (FCP) [5] has been proposed to provide disruption-free forwarding. FCP completely eliminates convergence and requires a novel centralized scheme to distribute topology maps. However, it is unlikely that routing convergence could be replaced in near future, while service disruption is a pressing issue.

This paper presents Cost-Carrying Packets (CCP), a routing framework that minimizes forwarding disruption time while retains efficient and loop-less convergence. Communication is immediately restored once the network failure is detected by its adjacent routers. Moreover, CCP neither changes the current convergence paradigm nor extends the convergence duration, and it can be combined with any existing and future link-state routing protocol.

2. DESIGN OVERVIEW

To achieve disruption-free forwarding without modifying convergence, packets need to carry some information to help routers detect routing anomaly. In our design, a packet carries a label $cost$ that specifies the remaining path cost from its current router n_i to its destination d perceived by n_i's upstream router n_{i-1}. The current router n_i compares this cost value with its locally computed path cost $n_i.cost(d)$[1], and takes different forwarding actions dependent on the comparison result.

Figure 1 shows an example of how CCP works in the Abilene network topology. Suppose the link between Denver and Kansas City fails. Duration convergence, it is possible that the Denver

---

[1]If the context is clear, we omit the destination in a path cost $n_i.cost$ for clarity.

node has updated its forwarding tables to use the Sunnyvale node to reach Kansas City, while the Sunnyvale node has not updated its forwarding tables. Without carrying costs, packets will loop between Sunnyvale and Denver, until the Sunnyvale node updates its forwarding tables.

With CCP, the Sunnyvale node stamps the remaining path cost (639) from its next hop to Kansas City into the packets it forwards to Denver. As the Denver node has updated its forwarding tables to bypass the failed Denver to Kansas City link, its local path cost (4456) is higher. The Denver node will detect a cost inconsistency. Since Denver's local cost is higher than the packet cost, the Denver node can infer its topology is smaller than other routers in the network, and its shortest path must be failure-free in case of single failure. Thus Denver updates the packet cost to be its own remaining path cost (3161, as shown in the figure), and forwards the packet back to the Sunnyvale node.

When the Sunnyvale node receives this packet, it will also detect a cost inconsistency, because Sunnyvale's local cost (1934) is smaller than the packet cost (3161). The Sunnyvale node can infer its topology is larger than other routers in the network, and its shortest path must pass the failure. Therefore Sunnyvale should not use its shortest path next hop to forward. Here the packet cost (3161) identifies a failure-free repair path: Sunnyvale → Los Angeles → Houston → Kansas City. Sunnyvale can choose the repair path identified by the packet cost, and forward the packet to Los Angeles instead of its default next hop Denver. The micro-loop between Sunnyvale and Denver is thus prevented, and eventually the packets will reach their destination Kansas City.

2.1 Prepare the Repair Path Database

A router needs to pre-compute repair paths that bypass certain network elements (links, nodes, or SRLGs) in case they fail in future. The repair paths' next hops and costs are stored in a repair path database (RPD). During a routing transition, if a router's local path cost differs from a packet cost, the router may attempt to find a repair path that matches the cost on the packet, and forward the packet along that path. A router may re-compute this RPD whenever it receives a routing update that changes its network map. The computation could be done in low priority to avoid slowing down convergence related tasks.

## 2.2 Forwarding Using Cost Carrying Packets

When a packet that carries its remaining path cost $pkt.cost$ arrives at a router n_i, the router compares its local cost $n_i.cost$ to the packet's destination with $pkt.cost$. Dependent on the comparison result, the router n_i takes different forwarding actions. The comparison has three outcomes:

Normal ($n_i.cost \equiv pkt.cost$): This indicates that the router n_i and its upstream router have consistent forwarding paths. The router updates the packet's cost $pkt.cost$ by subtracting its link cost to the next hop n_{i+1}: $pkt.cost \leftarrow pkt.cost - cost_{n_i \rightarrow n_{i+1}}$, and forwards the packet to its next hop n_{i+1}.

Cost Increase Inconsistency ($n_i.cost > pkt.cost$): This inconsistency shows that a router's local path cost is higher than its upstream router's cost. The network must be in routing transition, as the router n_i has computed different paths from other routers.

In the cost increase case, the router n_i repairs the inconsistency by updating the packet cost using its local cost: $pkt.cost \leftarrow n_i.cost - cost_{n_i \rightarrow n_{i+1}}$ and forwards the packet to its next hop n_{i+1}. This repair action is safe because n_i must have a smaller topology than other routers, as it has a higher path cost, and thus n_i's topology must not contain the changed equipment.

Cost Decrease Inconsistency ($n_i.cost < pkt.cost$): This incon-
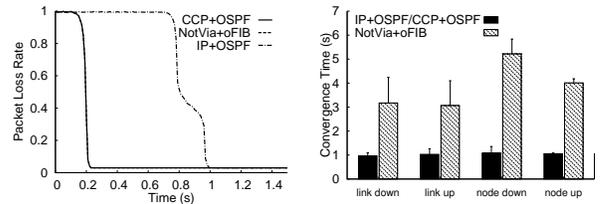
(a) Average packet loss rate   (b) Average convergence time

Figure 2: Preliminary results based on the Sprint topology after a single link failure

sistency shows that a router's local cost is lower than its upstream router. Again, the network must be in a transition, and the router will attempt to resolve the path inconsistency. However, as the router has a lower cost, it must have a larger topology. It is no longer safe to forward along the router's default next hop, because it may lead to a failed component.

To resolve a cost decrease inconsistency, a router uses the packet cost $pkt.cost$ to look up a repair path in its RPD. Suppose this lookup returns a next hop n'_{i+1}. The router n_i updates the packet cost using the link cost to reach n'_{i+1}: $pkt.cost \leftarrow pkt.cost - cost_{n_i \rightarrow n'_{i+1}}$, and forwards the packet to n'_{i+1}.

3. PRELIMINARY RESULTS

To learn how CCP may perform under real network settings, we did a simulation study of CCP using SSFnet, a widely-used packet-level simulator. In the simulation we compare CCP with both vanilla OSPF and the current IETF proposal under standardization, a combination of a route repair technique called NotVia [3] and a loop-free convergence scheme called oFIB [4].

Figure 2 shows some preliminary results of our evaluation. The results are generated based on the Sprint topology (315 nodes) inferred by the Rocketfuel project. In Figure 2(a) we show the average packet loss rate after a single link failure. From the figure we can see both NotVia+oFIB and CCP eliminate packet loss immediately after the failure is detected (after $\sim$ 200ms), while under vanilla OSPF the packet disruption times last about 1 second. In Figure 2(b) we show the average convergence time after various failure cases. As CCP does not modify the convergence protocol, it converges as fast as vanilla OSPF with around 1 second. oFIB on the other hand takes much longer time to converge ($3 \sim 5$ seconds) because it requires routers to update their routing information in a strict order rather than simultaneously. The results suggest that CCP achieves its goal: minimizing forwarding disruption without modifying convergence.

Details of the design and more results could be found at:
`http://www.ics.uci.edu/~angl/papers/ccp.pdf`

4. REFERENCES

[1] A. Atlas and A. Zinin. Basic Specification for IP Fast-Reroute: Loop-free Alternates. Internet draft, draft-ietf-rtgwg-ipfrr-spec-base-10, Nov 2007.

[2] S. Bryant and M. Shand. A Framework for Loop-free Convergence. Internet draft, draft-bryant-shand-lf-conv-frmwk-03.txt, Oct 2006.

[3] S. Bryant, M. Shand, and S. Previdi. IP Fast Reroute Using Notvia Addresses. Internet draft, draft-ietf-rtgwg-ipfrr-notvia-addresses-00.txt, Dec 2006.

[4] P. Francois and O. Bonaventure. Avoiding transient loops during the convergence of link-state routing protocols. *IEEE/ACM Transactions on Networking*, 15(6):1280–1932, Dec 2007.

[5] K. Lakshminarayanan, M. Caesar, M. Rangan, T. Anderson, S. Shenker, and I. Stoica. Achieving convergence-free routing using failure-carrying packets. In *SIGCOMM*, pages 241–252, 2007.

[6] S. Nelakuditi, S. Lee, Y. Yu, Z.-L. Zhang, and C.-N. Chuah. Fast local rerouting for handling transient link failures. *IEEE/ACM Trans. Netw.*, 15(2):359–372, 2007.