TED: Inter-domain Traffic Engineering via Deflection

Ming Zhu*‡††, Jun Li†§,Ying Liu*‖††, Dan Li*¶††, Jianping Wu*‖††

*Tsinghua National Laboratory for Information Science and Technology †University of Oregon
†† Department of Computer Science and Technology, Tsinghua University
‡zmvictorcruse@gmail.com §lijun@cs.uoregon.edu ¶tolidan@tsinghua.edu.cn ‖{liuying, jianping}@cernet.edu.cn

***Abstract*—As inter-domain routing on today's Internet does not and basically cannot consider traffic load when determining best traffic forwarding paths, it is not always optimal for a router to forward packets along its default path, especially when the router's default output port incurs a long queuing delay. In this paper, we design a new approach called TED in which border routers of autonomous systems (AS) adaptively deflect outbound traffic from a congested default path to an alternative path to significantly improve inter-domain traffic engineering (TE) and end-to-end throughput. With TED, every router only needs to examine the queue length of its own outgoing ports to orchestrate its deflection operation and ensure traffic forwarding is at line speed. It does not need to communicate or coordinate with other TED-capable routers or modify packet content, making TED incrementally deployable. Our evaluation shows that TED significantly increases the average throughput of traffic flows, and the improvement is comparable to directly upgrading router hardware and capacity. Finally, a prototype of TED on NetFPGA is also implemented.**

I. Introduction

A salient phenomenon facing today's Internet is the rapid growth of inter-domain traffic volume [1]. Daily traffic in large Internet Service Providers (ISP) is often in tens of petabytes (PB). For example, AT&T is carrying nearly 30 PB of data traffic on an average business day [2]. Traffic Engineering (TE), which adjusts the routing of traffic according to the prevailing demands, is therefore critical in order to improve the performance of traffic delivery and the efficiency of network resource usage.

What has been studied most, however, is intra-domain TE that is comprised of centralized optimization of routing configuration based on a full router-level topology, link capacity restriction, and predicted traffic matrix. Inter-domain TE, in contrast, is less studied and more difficult given the lack of global topology knowledge and cooperation. Based on our investigation, it has only a few limited options: actively sending BGP announcements to control inbound traffic [3], upgrading device capacity with a high monetary cost [4], or relying on intra-domain TE [5] with complicated optimization model.

In this paper, we design and evaluate a simple and practical routing system called **TED**, i.e., **T**raffic **E**ngineering via **D**eflection, that enables inter-domain TE on autonomous systems (AS) border routers. More specifically, TED-enabled AS border routers can adaptively offload outbound traffic from a congested default path to an alternative path. While the hierarchical structure of the Internet is becoming more flat [6] and more paths exist between every pair of ASes, BGP routers often fail to handle the burstiness of Internet traffic[7] by single path selection and the performance degradation due to congestion on bottleneck links. TED addresses this limitation by enabling a BGP router to explore its RIB to obtain alternative paths. As the RIB provides multiple paths towards each destination along different outgoing links, the path selection daemon in TED translates multiple paths into multiple links and forwards packets through the link with smallest forwarding delay, where the delay can be gauged by monitoring the queue length of each outgoing port.

TED introduces many design choices to optimize its performance. For instance, as looking up the *RIB* at a BGP router can greatly slow down packet forwarding performance, TED builds an auxiliary FIB in the data plane which is updated by the path selection daemon in the control plane. It is also worth noting that TED is completely compatible with legacy routers and is incrementally deployable. It can be deployed with no requirement of cooperation between routers, and it forwards packets without any extra modifications of packets.

Our evaluation shows that TED can significantly improve inter-domain traffic engineering and we have also implemented a prototype of TED on NetFPGA. Our simulations on a dumbbell topology shows that TED's performance is directly comparable to upgrading router hardware and capacity: using N alternative paths is the same as expanding default path bandwidth by N times! Our evaluations of TED on real Internet topology demonstrates that TED considerably outperforms BGP routing, even with partial deployment. For example, with a 50% deployment, the average throughput of 61% of flows will be more than half of link capacities, while without TED only 32% flows can be so. Moreover, no loop is found during the evaluation and the tracing result shows that TED is harmless in packet reordering.

The work was supported by the National Natural Science Foundation of China under Grant No.61161140454, No.61170291, No.61133006, and the National Key Basic Research Program of China (973 program) under Grant 2014CB347800

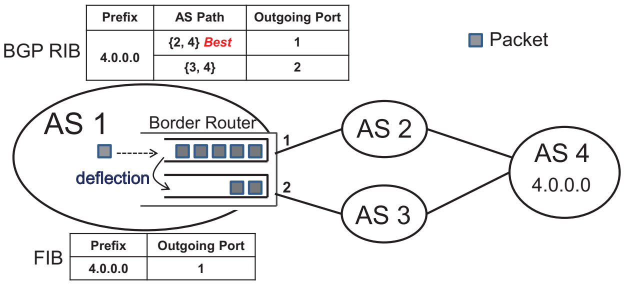


Fig. 1: Traffic deflection via TED.

The rest of the paper is organized as follows. Section II-B presents the challenges and the basic idea of TED. Section III describes the design details of TED. Section IV analyzes the packet reordering issue. Section V evaluates TED's performance. Section VI describes our prototype implementation of TED on NetFPGA. Section VII discusses related work. Finally, Section VIII concludes the paper.

II. Overview

A. Challenges

Load-sensitive traffic deflection via TED can be easily deployed on border routers as well as being compatible with legacy routers. The key idea is to fast forward packets to alternative paths other than undergoing large queuing delays on default forwarding paths. However, several challenges must be addressed in designing a practical solution.

Alternative path selection. This challenge consists of two issues: where the candidate paths are and what the selection rule is. For the first issue, TED learns candidate paths by exploring the RIB at a BGP router. Each AS border router stores its BGP RIB in the control plane, which contains multiple paths towards any destination. Each path in the RIB is valid since it conforms to BGP policy such as being loop-free and valley-free. For the second issue, TED introduces traffic-awareness in selecting paths. However, it is impractical to identify which path has the best performance for every packet. We therefore translate path selection into link selection, and send packets to the link with smallest forwarding delay.

Frequent switching between data plane and control plane. While path selection is accomplished by looking up the RIB on the control plane, packet forwarding is done in the data plane. Switching between two planes per packet deflection will degrade forwarding performance. We address this problem by building an auxiliary FIB at every TED-capable router to help forward packets to alternative paths in the line speed. The auxiliary FIB is updated by the path selection daemon in the control plane.

Open Challenges. With TED, packets belonging to the same flow may travel along different paths to cause packet reordering. We analyze it in Section IV-A and evaluate it in Section V-B, and argue that TED is harmless in terms of packet reordering.

B. Basic Idea

We now illustrates the basic idea of TED. As shown in Figure 1, AS 4 announces network $4.0.0.0$ (for convenience we ignore the length of prefix in our notation) to the Internet and AS 1 learns two AS paths towards $4.0.0.0$, i.e., $\{2,4\}$ and $\{3,4\}$. Assuming that AS 1 selects $\{2,4\}$ as the default path, the border router of AS 1 then builds its RIB and FIB, as shown in Fig. 1. If traffic volume towards $4.0.0.0$ through AS 1 is increasing rapidly, congestion may happen on the default path. More specifically, at AS 1's border router more and more packets will be waiting in the queue of the default output port, i.e., port 1, leading to large queuing delay or even packet loss. With TED, the router could deflect subsequent packets toward $4.0.0.0$ to output port 2 to make use of the alternative path $\{3,4\}$, which is under-utilized given fewer packets queuing in the buffer.

In Fig. 1, the default path and the alternative path go through the same border router, which thus can select a path by simply comparing the forwarding delay between output ports 1 and 2. However, the alternative path may be located via another router, naming that the operation of path selection should be achieved between iBGP peers. We detail the deflection process between iBGP peers in Section III-C.

III. Design

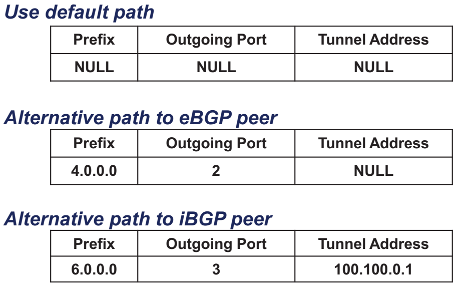
A. Alternative Path Selection

Fig. 1 shows that traffic deflection happens when the congestion happened on the link corresponding to a default path. In fact, link utilization can be simply observed by tracking the buffer of outgoing ports. For example, 100% queuing ratio indicates that a link is congested and subsequent packets will be dropped, while an empty queue indicates that a link is under-utilized.

TED runs a forwarding engine, and deflects packets before the buffer is full by exploring link forwarding delay. *The forwarding engine always prefers the alternative path with the smallest forwarding delay*. We divide link forwarding delay into three parts: queuing delay in a buffer, transmission delay on an outgoing port, and link propagation delay. We let p denote the incoming packet size, q the current queuing size (with the unit of size being either bytes or the number of packets), r the NIC sending rate (e.g., 1Gbps) and l denotes the link propagation delay. In addition, we take B as the buffer size for an outgoing port. Therefore, the forwarding delay F_d is the sum of queuing delay(Q_d), transmission delay(T_d) and link propagation delay(L_d), as shown in Equation 1. If $q = B$, the buffer is full and subsequent packets will be dropped, thus an infinite forwarding delay.

$$\begin{cases} F_d = \infty & \text{if } q = B, \\ F_d = Q_d + T_d + L_d & \text{if } q < B. \end{cases} \tag{1}$$
$$\textbf{where} \quad Q_d = \frac{q}{r}, \quad T_d = \frac{p}{r}, \quad L_d = l$$

It is worth noting that sending rate, link propagation delay and buffer size seldom changes during the forwarding process. Moreover, the transmission delay (T_d) is negligible given its relatively small value. For example, if $p = 1KB$ and $r =$



Use default path

Prefix	Outgoing Port	Tunnel Address
NULL	NULL	NULL

Alternative path to eBGP peer

Prefix	Outgoing Port	Tunnel Address
4.0.0.0	2	NULL

Alternative path to iBGP peer

Prefix	Outgoing Port	Tunnel Address
6.0.0.0	3	100.100.0.1

Fig. 2: Auxiliary FIB.

$1Gbps$, T_d will be only $8\mu s$. Therefore, to compare F_d, a router only needs to monitor the queuing size of its output buffers.

B. Deflection in Practice

Up to now, an ideal traffic deflection is to forward the packet to an alternative path with the smallest forwarding delay. However, while packet forwarding is running in the data plane, searching for an alternative path requires querying the *RIB* at a BGP router in the control plane. On one hand, processing speed is much slower in the control plane. On the other hand, switching between two planes per packet deflection is time-consuming, especially when handling bursty traffic. Therefore, *we build an auxiliary FIB to maintain traffic deflection in the data plane*. While an outgoing port in the regular FIB still corresponds to the default path, entries in the auxiliary FIB point to alternative paths. Our routing system is thus described as follows:

In the data plane, a TED router accesses the regular FIB and the auxiliary FIB in parallel for each packet. The forwarding engine prefers to use the search result—if not NULL—from the auxiliary FIB. (Note that the searching result from the regular FIB will never be NULL.) Basically, traffic deflection happens if a packet is forwarded by following the auxiliary FIB.

In the control plane, a TED router employs the path selection daemon to update its auxiliary FIB in real time. If a default path becomes under-utilized and a faster alternative path exists, the daemon loads a new entry into the auxiliary FIB, as well as removes the old entry if any.

Fig. 2 illustrates the structure of an auxiliary FIB. It adds an extra item, Tunnel Address, to handle the deflection of packets to an iBGP peer. We will explain the tunnel methodology in Section III-C. In searching its auxiliary FIB, if the result is NULL, a RED router will deduce that either no "faster" alternative paths exists or no congestion happens in the default path, thus the packet should be forwarded following the default path. If the result is not NULL but the tunnel address is NULL, the router can deflect the packet directly to the alternative path since it is connected to an eBGP peer. Then otherwise, the router encapsulates the packet with the tunnel address from the result and deflects the packet to an iBGP peer.

The auxiliary FIB is updated by the path selection daemon in the control plane. The daemon persistently monitors the queuing ratio ($U = \frac{q}{B}$) of each outgoing port. Once the queuing ratio U_P of an outgoing port P is greater than the threshold (e.g., 90%), the daemon will seek other outgoing ports with smaller forwarding delay than P in order to reduce the amount of packets to be forward via P. If such ports exist, the daemon then inserts entries into the auxiliary FIB so that packets can be forwarded via alternative paths instead of the default paths. To build such entries, the daemon must find the best alternative path for each network prefix in the regular FIB. Meanwhile, if any entry already exists in the auxiliary FIB for the prefix, the daemon will remove it to prevent packet deflection according to the obsolete entry. Alg. 1 illustrates the path selection daemon. For the congested outgoing port P (Line 2), it removes the related entries in the auxiliary FIB (Line 3-6). Then, for each network prefix related to P (Line 7-9), it finds the best alternative path with the smallest forwarding delay (Line 10) and inserts the entry into the auxiliary FIB (Line 11).

Algorithm 1 Path Selection Daemon

```
Input: $T$ is the threshold of queuing ratio
 1: procedure PathSelect($T$)
 2:     $P \leftarrow GetCongestPort(T)$
 3:     foreach entry $e$ in $AuxFIB$ do
 4:         if $e.outgoing\_port = P$ then
 5:             $Remove(AuxFIB, e)$
 6:     end foreach
 7:     $E[] \leftarrow GetEntries(FIB, P)$
 8:     foreach $e$ in $E[]$ do
 9:         $prefix \leftarrow e.prefix$
10:         $alt\_P, tun\_addr \leftarrow FindAltPath(prefix)$
11:         $Insert(AuxFIB, prefix, alt\_P, tun\_addr)$
12:     end foreach
13: end procedure
```

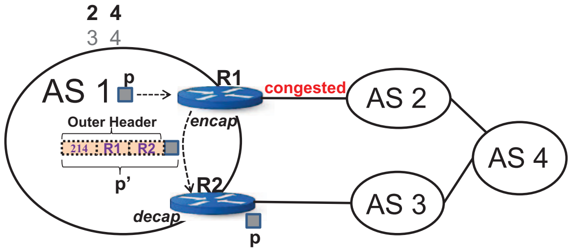


Fig. 3: IP-in-IP Tunneling.

The path selection daemon (Alg. 1) is running on the control plane. Its update to the auxiliary FIB is asynchronous to the forwarding process in the data plane. We apply this model to avoid frequent switching between the two planes so as to keep packet forwarding fast.

C. Deflection between iBGP peers

To implement traffic deflection between iBGP peers, we employ IP-in-IP tunneling, a technique that commercial routers have widely implemented. We illustrate it in Fig. 3. In R_1, packet p towards AS 4 is supposed to go to AS 2 by following the default path. If the corresponding link is congested, the packet should be deflected to an alternative path. However,

R_1 needs to send p to the iBGP peer R_2, in the first place. R_1 encapsulates p in a new packet p'. In the outer IP header of p', the source address is R_1, the destination address is R_2, and the protocol number is assigned to indicate it is our protocol. When R_2 receives p', it decapsulates outer header and forwards the origin packet p towards AS 4. Note that the default path to AS 4 in R_2 is also $\{2, 4\}$ since all the iBGP peers have a consistent route view, implying that R_2 would send p back to R_1. Instead, by checking the protocol number in the outer header of p', R_2 is informed that packet p is deflected from the default path and supposed to go along an alternative path. R_2 hence forwards p to AS 3.

IV. Discussion

A. Packet Reordering

While transmitting packets along both default and alternative paths may cause packet reordering, according to the measurement in [8], packet transmission along parallel paths only cause a relatively small proportion of packets to be out of sequence. In fact, TED does not aggravate packet reordering. A formal proof in [9] shows that if the interval between two successive packets in a flow is larger than the maximum delay between parallel paths, the second and subsequent packets from the flow can route along any available paths without reordering. Furthermore, a lower frequency in path switching leads to fewer packet reordering. In TED, the frequency of path switching depends on the duration of congestion phenomenon. TED's forwarding process starts to send packets to an alternative path when a default path becomes congested, and it continues using the alternative path until the default path becomes normal. Our evaluation in Section V-B shows that most flows have only *one* path switching in TED, thus causing virtually no harm due to packet reordering.

V. Simulation

We study the performance of TED and compare it with the normal BGP routing by using an NS-3 simulation. We implement the forwarding engine and path selection daemon on each routing node. The packet size is set to 1KB.

We mainly focus on the cumulative distribution of flow throughput. We first investigate the quality of TED in a *dumbbell topology*. The result shows that TED displays similar performance to directly upgrading the link capacity. We also evaluate the performance of our routing system by using *real AS topology* [10] and inferred AS-level traffic matrix. Due to business secrecy real AS-level traffic matrix is hard to obtain. We regard popular content providers (such as Google and Facebook) as traffic generators, and by mining AS relationships from traced data [10], take stub ASes as traffic consumers ordered by their number of providers and peers. The more providers and peers a stub AS has, the more traffic it consumes.

A. Dumbbell Topology

The dumbbell topology is illustrated in Fig. 4. Each node represents an AS. There are 100 TCP flows and each flow is 1MB and sent from S_i to D_i where $i \in [1, 100]$. The start time of each flow follows a Poisson process such that the average number of started flows in one second is λ. We also call the bottleneck link LR as the default link. The left link S_iL, right link D_iR and the default link LR are configured as 10Mbps link capacity and 1ms link propagation delay. Also between L and R are K alternative links. $K = 0$ denotes the regular routing that only the default link can be used. $K > 0$ denotes that TED is in place with packet deflection. By default, alternative links are set as 10Mbps link capacity and 1ms link propagation delay. The buffer size B is shown as black solid circle. We set $\lambda = 50$ and $B = 100$KB as their default values.

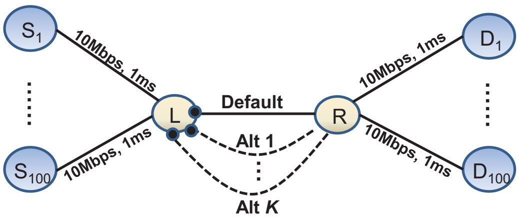


Fig. 4: Dumbbell topology.

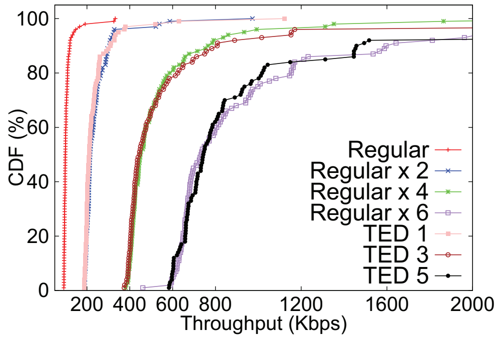


Fig. 5: CDF of throughput with 400KB buffer size.

We compare the regular routing, TED, and "enhanced" regular routing to study the flow throughput. The "enhanced" regular routing refers to upgrading the capacity of the default link (LR); for example, "Regular x 2" indicates the default link is 20Mbps while "Regular x 4" is 40Mbps. TED is denoted by the number of alternative links deployed. For example, "TED x 1" indicates 1 alternative link ($K = 1$) while "TED x 3" indicates 3 alternative links ($K = 3$).

Fig. 5 illustrates the CDF of flow throughput with 400KB buffer size. The regular routing, which is leftmost in each figure, provides limited throughput for each flow. Considering a 10Mbps default link is competed by 100 flows, each flow only obtains 100Kbps bandwidth on average. By increasing the default link's capacity, an "enhanced" regular routing should provide higher throughput accordingly. For example, it is expected to be 200Kbps on average in "Regular x 2", or 400Kbps in "Regular x 4". Fig. 5 illustrates the result of "enhanced" regular routing by setting the default link capacity to be 20Mbps, 40Mbps and 60Mbps respectively. We found that TED with a different number of alternative links has similar effect to "enhanced" regular routing. As Fig. 5 shown, the curves of "TED x 1", "TED x 3" and "TED x 5" are nearly overlapped with "Regular x 2", "Regular x 4" and "Regular x 6", respectively. *This result indicates that our traffic deflection methodology is comparable to directly upgrading link capacity.*

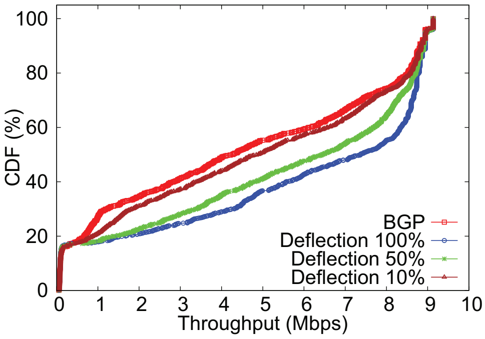
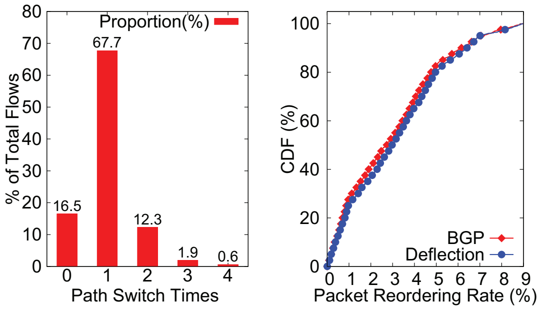


Fig. 6: BGP vs. partially deployed deflection.



(a) Path Switch Distribution

(b) Packet Reordering Trace

Fig. 7: Traffic flow stability with TED.

More specifically, in such a dumbbell topology, TED with N alternative links is as effective as enlarging the default link capacity in the regular routing for N times.

B. Internet Topology

We compare TED with the standard BGP on a realistic Internet topology. We use the January 2014 CAIDA AS-level graph [10] gathered from RouteViews BGP tables [11], which includes 45,845 ASes and 187,527 links, with every link set to 10Mbps capacity and 1ms propagation delay. We use standard “valley-free” [12] export policies to match economic incentives, means customer routes preferred over peer routes, which in turn are preferred over provider routes. If multiple routes fall in the same category, the first tie breaker is the length of AS paths, and the second is the lowest next-hop AS identifier. In addition, while every AS is represented as a single node on the AS-level graph, we pick a tier-1 AS and expand it to capture all of its internal topology at router level; in doing so, we assume all the border routers (iBGP peers) in this tier-1 AS are connected in a full mesh topology.

We introduce a traffic matrix at the AS level. Every flow is 1MB and travels from a content provider to a stub AS. The pattern of traffic follows the power-law distribution. More specifically, the probability of consuming traffic from the i-th generator is $f(i) = a \times i^{-\alpha}$, where $\frac{1}{a} = \sum_{i=1}^{N} i^{-\alpha}$ and N is the total number of generators. In our simulation, we choose $N = 1000$, $\alpha = 1.0$, and the total number of flows is 3000.

Fig. 6 illustrates the comparison between TED and the general BGP in terms of the cumulative distribution of flow throughput. To investigate the performance of TED, we partially deploy it in the AS graph. For example, 100% deployment indicates that each AS is running TED while 50% deployment refers that only half of the ASes are TED-capable. Fig. 6 shows that TED gains larger throughput than the normal BGP routing. For example, the average throughput of 61% of flows will be more than half of link capacities (higher than 5Mbps), while without TED only 32% flows can be so. It also indicates that better performance is achieved with a higher proportion of TED deployment as more alternative paths are utilized.

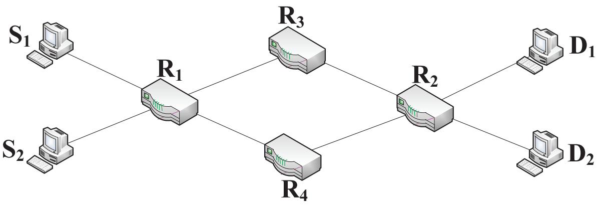


Fig. 8: Experiment topology with TED prototype.

We also evaluated the stability of flows. A path switch happens if a router deflects a packet from the default path to an alternative path, or resumes using the default path after using an alternative path. As Fig. 7(a) shows, more than half of flows (67.7%) had a path switch only once and most flows (97.5%) switch paths no more than twice. More specifically, by directly tracing the packet reordering rate on the receiver side of each flow, Fig. 7(b) illustrates that TED presents nearly the same result as the normal BGP routing, meaning that it is harmless in packet reordering.

VI. Prototype Implementation and Experimentation

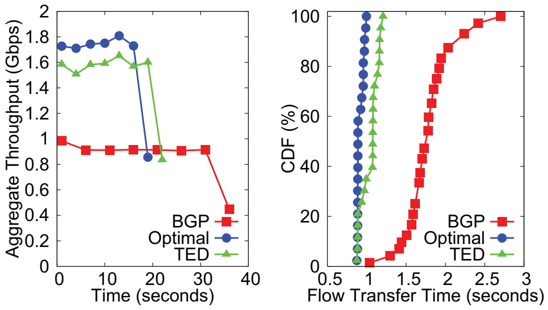
A. NetFPGA-based Implementation

We have implemented a TED prototype on a NetFPGA card. The implementation includes programming in Verilog and C with totally 1200 lines of code. It is divided into a hardware plane and a software plane, corresponding to the data plane and the control plane of routing, respectively. The hardware plane works at high speed but is only able to execute simple logic. For complex processing logic, we need to resort to the software plane. In our implementation, forwarding data packets is processed in hardware, while path selection daemon is handled by software.

B. Experiment Setup

We run TED in a testbed composed of eight desktop machines. Four machines (S_1, S_2, D_1, D_2) are end hosts sending and receiving traffic flows, while the other four machines (R_1–R_4) function as routers equipped with TED implementation. We use each router to represents an AS. The servers and routers are connected by Gigabit Ethernet links in a simple dumbbell topology, as shown in Fig. 8.

In our experiments, S_1 generates 20 TCP flows to D_1 one after another. So does S_2 toward D_2. Each flow is 100MB and every data packet is 1KB. The buffer size of each outgoing port in every router is set to 500KB as limited by the NetFPGA



(a) Aggregate Throughput (b) Flow Completion Time

Fig. 9: Experimental results.

configuration. S_1 and S_2 start packet transmission at the same time. With BGP routing, R_1 selects $R_1R_3R_2$ as the default path towards D_1 and D_2, meaning that R_1 forwards each data packet to R_3. For this topology, an optimal routing (on R_1) is to forward packets towards D_1 via R_3 and forward packets towards D_2 via R_4. We run BGP routing, TED, and optimal routing separately, and measure the aggregated throughput of the network as well as the completion time of each flow.

C. Results

TED significantly outperforms the legacy BGP routing. From Fig. 9(a), we see that TED presents similar aggregated throughput as the optimal routing. In BGP routing, flows towards D_1 and D_2 compete for the default path $R_1R_3R_2$, which results in a limited link capacity obtained by each flow. On the other hand, by leveraging the alternative path $R_1R_4R_2$, TED achieves a higher link capacity. The aggregated throughput in TED is around 1.7Gbps during flow transmission, while it is only 0.93Gbps in the legacy BGP routing. Furthermore, Fig. 9(b) illustrates the CDF of the flow completion time. Every flow is completed within 1.2s in TED, while in BGP routing 80% flows need more than 1.5s. Moreover, it takes only 20.5s in TED to complete all the flows, but takes 37.6s in BGP routing.

VII. Related Work

The Internet is Flat. Previous measurements show that the hierarchy of Internet is becoming flatter [6]. The inferred AS-level Internet graph in January 2014 has a large average node degree but a small diameter [10]. Popular content providers such as Google and Facebook have also been noted to have enormous amounts of peers [13]. All these evidence indicates that numerous paths exist between each pair of ASes, which benefits TED by allowing traffic deflection among multiple paths for better load balance.

Traffic Engineering. Intra-domain traffic engineering is widely studied since in a local ISP one can obtain a full-scale view of the network and deploy a solution comprehensively. Various approaches have been proposed to route traffic according to observed traffic demand matrices [5]. In contrast, considering the lack of global knowledge and cooperation, inter-domain TE is more challenging. Most of works on inter-domain routing focus on topology issue, such as assigning a customized backup AS PATH in case of link failure [14], or locating the root cause of path changes as soon as possible [15]. Few attentions have been paid toward traffic-aware routing. Rather than building complicated optimization models, TED is a load-sensitive routing system, and enables AS border routers to adaptively forward packets to different links. Since it achieves load balance by only shaping the outbound traffic, TED is able to combine with any type of intra-domain TE to better optimize traffic demands.

VIII. Conclusion

In this paper we propose Traffic Engineering via Deflection, or TED, that enables the border routers of autonomous systems to deflect traffic to alternative paths if the default paths are congested. TED employs a path selection daemon to explore the RIB table at a BGP router to identify the best alternative path for a destination, sometimes through an iBGP peer, and updates the auxiliary FIB for forwarding deflected traffic to keep line-speed. Our evaluation of TED shows that it can significantly improve traffic throughput, and the improvement is comparable to directly upgrading router hardware and capacity. We also implemented a NetFPGA-based prototype of TED, and our experiments with the prototype further demonstrates that TED considerably outperforms the legacy BGP routing even with partial deployment. Our future work includes evaluating TED in a large-scale testbed.

References

[1] C. Labovitz, S. Iekel-Johnson, D. McPherson, J. Oberheide, and F. Jahanian, "Internet inter-domain traffic," in *SIGCOMM*, 2010.

[2] "AT&T company information," http://www.att.com/gen/investor-relations?pid=5711, 2012.

[3] B. Quoitin, S. Uhlig, C. Pelsser, L. Swinnen, and O. Bonaventure, "Interdomain traffic engineering with BGP," *IEEE Communications Magazine Internet Technology Series*, 2003.

[4] "Cisco routers," http://www.cisco.com/en/US/products/hw/routers.

[5] H. Wang, H. Xie, L. Qiu, Y. R. Yang, Y. Zhang, and A. G. Greenberg, "COPE: traffic engineering in dynamic networks," in *SIGCOMM*, 2006.

[6] P. Gill, M. Schapira, and S. Goldberg, "Let the market drive deployment: A strategy for transitioning to BGP security," in *SIGCOMM*, 2011.

[7] H. Jiang and C. Dovrolis, "Why is the Internet traffic bursty in short time scales?" in *SIGMETRICS*, 2005, pp. 241–252.

[8] S. Jaiswal, G. Iannaccone, C. Diot, J. F. Kurose, and D. F. Towsley, "Measurement and classification of out-of-sequence packets in a tier-1 IP backbone," in *Internet Measurement Workshop*, 2002.

[9] S. Kandula, D. Katabi, S. Sinha, and A. Berger, "Dynamic load balancing without packet reordering," *SIGCOMM Comput. Commun. Rev.*, 2007.

[10] "The CAIDA AS relationships dataset," http://www.caida.org/data/as-relationships/., 2013.

[11] "Routeviews," http://www.routeviews .org/.

[12] L. Gao and J. Rexford, "Stable Internet routing without global coordination," *IEEE/ACM Trans. Network.*, 2001.

[13] Y. Shavitt and U. Weinsberg, "Topological trends of Internet content providers," *CoRR*, 2012.

[14] N. Kushman, S. Kandula, D. Katabi, and B. Maggs, "R-BGP: Staying connected in a connected world," in *NSDI*, 2007.

[15] U. Javed, I. Cunha, D. R. Choffnes, E. Katz-Bassett, T. Anderson, and A. Krishnamurthy, "PoiRoot: Investigating the root cause of interdomain path changes," in *SIGCOMM*, 2013.