Characterizing and Mitigating Inter-domain Policy Violations in Overlay Routes

Srinivasan Seetharaman and Mostafa Ammar
Networking and Telecommunications Group, College of Computing
Georgia Institute of Technology, Atlanta, Georgia 30332
{srini,ammar}@cc.gatech.edu

***Abstract*— The Internet is a complex structure arising from the interconnection of numerous autonomous systems (AS), each exercising its own administrative policies to reflect the commercial agreements behind the interconnection. However, routing in service overlay networks is quite capable of violating these policies to its advantage. To prevent these violations, we see an impending drive in the current Internet to detect and filter overlay traffic. In this paper, we first present results from a case study overlay network, constructed on top of Planetlab, that helps us gain insights into the frequency and characteristics of the different inter-domain policy violations. We further investigate the impact of two types of overlay traffic filtering that aim to prevent these routing policy violations: *blind filtering* and *policy-aware filtering*. We show that such filtering can be detrimental to the performance of overlay routing. We next consider two approaches that allow the overlay network to realize the full advantage of overlay routing in this context. In the first approach, overlay nodes are added so that good overlay paths do not represent inter-domain policy violations. In the second approach, the overlay acquires transit permits from certain ASes that allow certain policy violations to occur. We develop a single *cost-sharing* framework that allows the incorporation of both approaches into a single strategy. We formulate and solve an optimization problem that aims to determine how the overlay network should allocate a given budget between paying for additional overlay nodes and paying for transit permits to ASes. We illustrate the use of this approach on our case study overlay network and evaluate its performance under varying network characteristics.**

I. Introduction

Overlay networks have recently gained attention as a viable alternative to overcome functionality limitations (e.g., lack of QoS, difficulty in geo-positioning) of the Internet, or to enhance the services currently being offered (e.g., rerouting around failures). The basic idea of overlay networks is to form a virtual network on top of the physical network so that overlay nodes can be customized to incorporate complex functionality without modifying the native IP network[1]. Typically, these overlays route packets over paths made up of one or more overlay links to achieve a specific end-to-end objective. Each intermediate overlay node is referred to as a *relay* and the forwarding operation at each hop is referred to as *relaying*.

We address a fundamental question in this paper with regards to the impact of overlay routing on the enforcement of *native network routing policies*. These policies are in place as a result of the economics involved in the interconnection between the different autonomous systems (AS) in the Internet today. More specifically, we are interested in the extent to which overlay paths violate AS transit policies.

This work was supported in part by NSF grant ANI-0240485.

[1]Our work pertains to infrastructure or service overlays, rather than peer-to-peer networks.

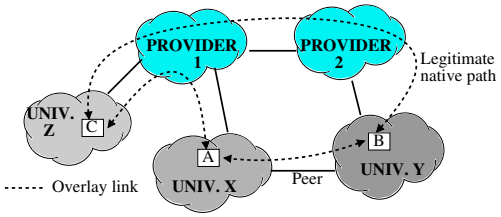


Fig. 1. Policy violations using overlay routing

Consider, for example, a hypothetical AS-level connectivity graph as show in Fig. 1. In that figure, nodes A, B and C are overlay nodes trying to obtain the best possible route to each other. Node B can route data to node C using the overlay path *BAC*, which results in University X's AS being used for transiting traffic between Universities Y and Z. This is a violation of the AS transit policy. From an economic perspective, we see that University Y saves money paid to Provider 2, by not using the legitimate route between nodes B and C. This saving comes at the expense of University X, which is not part of any transit agreement. Because overlays operate at the application layer, this violation typically goes undetected by the native layer.

We start our investigation by evaluating a case study overlay network constructed over the PlanetLab testbed[1] which provides insights into the frequency and characteristics of the different violations (results are reported in Section III). It is interesting to note that close to 70% of the multi-hop overlay routes in our dataset violate the native layer policies. It is worthwhile to observe that these native policy violations are not a serious issue if the overlay traffic is a minor component of the overall traffic. But, in some cases it is already a significant portion of Internet traffic. It is also likely that overlay traffic will experience significant growth in the future.

As awareness of the impact of overlay applications increases, there is an impending drive to incorporate extra complexity at the native layer to manage the overlay traffic [2], [3], [4]. There have been two types of commercial solutions proposed for both enterprise networks and service provider networks:

- Those that help manage overlay traffic without impacting the user experience[4], [5], [6]
- Those that help filter out overlay traffic without concern for the user experience[7], [8], [9]

This second class of solutions motivates us to investigate the impact of their deployment to counter the inter-domain policy violations committed by overlay routes. Specifically, we focus on two types of overlay traffic filtering – *blind filtering* and *policy-aware filtering*. We show that such filtering can be detrimental to the performance of overlay routing.

There exists two forms of overlays that can cause native policy violations - i) service overlays[10], where an overlay service provider (OSP) purchases resources from the underlying native layer ISPs in order to offer a value-added network service to actual end-systems, and ii) end-system overlays (e.g., Skype[11]). In the presence of filtering, the user experience will suffer in both these overlays. However, because a service overlay is managed by a single operator, it is feasible for the overlay network to regain the full advantage of overlay routing by adopting one of two approaches we propose. In the first approach, overlay nodes are added so that good overlay paths do not represent inter-domain policy violations. This approach attempts to insure that overlay paths conform to native policy. In the second approach, the overlay acquires transit permits from certain ASes that allow certain policy violations to occur but only for permitted overlay traffic. This approach attempts to "legitimize" native policy violations through commercial agreements between the overlay service provider and the native network.

We further develop a single *cost-sharing* framework that allows the incorporation of both these approaches into a single strategy. We formulate an optimization problem that aims to determine how the overlay network should allocate a given budget between paying for additional overlay nodes and paying for transit permits to ASes. We develop a heuristic solution to this problem and illustrate its application on our overlay case study. Further, we evaluate its performance under varying network characteristics.

The remainder of the paper is organized as follows. We describe the different types of transit violations possible in Section II. We characterize the extent of native policy violations using our case study overlay network and present associated results in Section III. Section IV investigates the effect of native layer enforcement of routing policy on the performance of the overlay. We present our cost-sharing approach for mitigating the effect of packet filtering in Section V. Previous research related to our work are briefly described in Section VI. This paper is concluded in Section VII.

II. A Classification of Policy Violations

The current Internet is made up of thousands of autonomous systems that coexist, cooperate, and compete for usage of its various resources. Each AS establishes some form of native layer policy to express its willingness to allow or deny traffic from its neighboring ASes. The Border Gateway Protocol (BGP) is the policy-based routing protocol that runs between autonomous systems, implements the various policy constraints and helps each AS select the routes to a destination.

The native network policies are primarily motivated by economic costs and performance gains[12]. These policies predominantly reflect the commercial agreements between ASes and are encoded into the router configuration to be enforced at ingress and egress points of the administrative domain. The combination of these individual policies determine the *AS-path* used by the flows in the Internet; the AS-path is defined as the ordered list of ASes a packet needs to traverse to reach the intended destination. The interconnection of over 20,000 different ASes in the Internet leads to a complex structure with path inflation[13] and unpredictable routing behavior at times[14].

Overlay layer packet forwarding is quite capable of violating the native policy restrictions. Fig. 2 gives an example of a native layer policy violation we noticed during our overlay route measurement process. In this example, the overlay path shown in the figure was determined to be optimal and in particular was superior to the native network path from Colorado State University to the University of North Carolina. We can clearly see that the situation might be undesirable for the Harvard University native network, while being beneficial to the overlay nodes at Colorado State University (by providing it an alternate path with 6.3ms lower latency, a 10.48% gain over the native route).

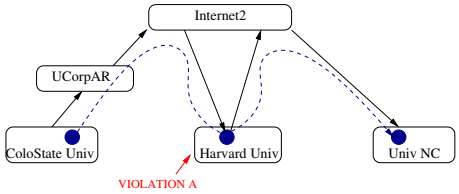


Fig. 2. Example of a plausible violation in reality. The solid circles represent the overlay nodes and the dashed line represents the end-to-end overlay path.

There are different types of native layer policy violations possible, based on what policies the border router enforces. Overlay routing can potentially violate any of these policies at will. However, we restrict our work to the violation of the *valley-free* property of AS-paths, which states that no AS will act as a transit for traffic originating from its provider or its peer, unless the traffic is destined to its customer[15]. We are only interested in the violation of the valley-free property because this is the only case in which the violation is experienced by an AS not involved in any way with the end-to-end communication. Hence, we consider this to be the most reproachable. When we view the example in Fig. 2, we can see that Harvard University is having to use its access bandwidth to/from the Internet to act as a transit between Colorado State University and the University of North Carolina.

Note that if the overlay node at Harvard University were also a *consumer* of the data being forwarded (in addition to being a relay), we do not consider this a violation since this becomes true application-layer forwarding. In the above example, if the overlay node at Harvard University were a consumer of the data, then it will be part of the communication even in the absence of relaying. More examples of this include end-system multicast and P2P file-sharing, where the intermediate node also uses the content. Hence, a transit violation is when the actual end-to-end AS-path used by the overlay is a violation of the native routing policy and the relay nodes on the overlay path are non-consumers of the data being forwarded.

There are multiple reasons for using a multi-hop overlay path, rather than the direct overlay link (essentially the native network path between two nodes) e.g., achieve better performance or resilience, circumvent limitations imposed by firewalls and NAT boxes, or load balancing, to name just a few. Our paper investigates violations that arise in such multi-hop overlay paths.

Previous studies on BGP misconfiguration highlighted four different types of route export violations prevalent in the current Internet[14], each of which violates the valley-free property of AS-paths. In the same spirit, we investigate the type of violations caused by overlay routing, which control the route for a certain end-to-end connection by using intermediate relays. Note that the basic native route between two overlay nodes (referred to as the *overlay link*) never constitutes a policy violation; under the assumption that the native routing conforms to policy. We proceed to investigate multi-hop overlay paths.

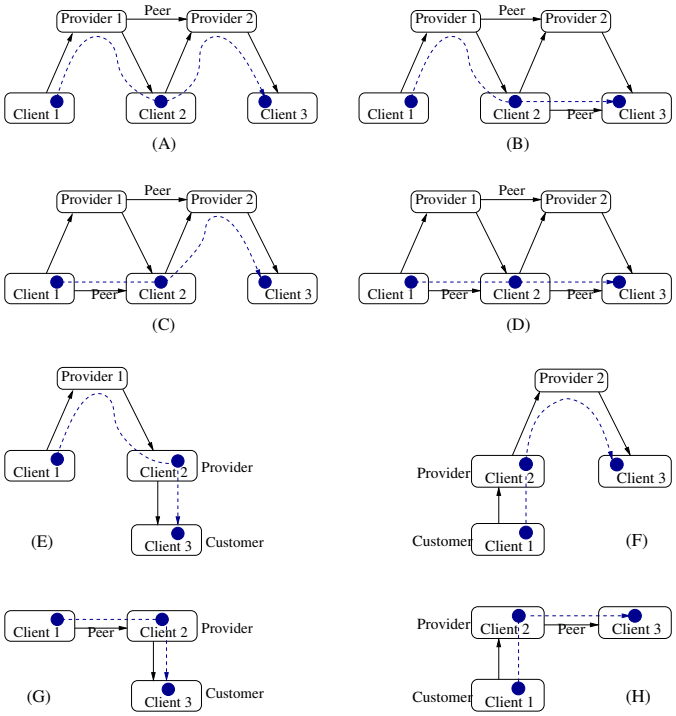


Fig. 3. AS relationships within each overlay path. The solid circles represent the overlay nodes and the dashed line represents the end-to-end overlay path.

Fig. 3 illustrates eight different forms of relaying possible. We argue that cases A, B, C and D represent a violation of native layer policies as the overlay traffic uses the intermediate overlay node to transit through client 2. Thus, client 2 acted as a transit between its provider and peer, which is a violation of the commercial relationships between the ASes. However, cases E, F, G and H do not represent violations because one of the overlay nodes was located in a provider (non-stub) network. In general terms, no violation exists if the native routing policy condones or allows client 2 to be a transit between client 1 and client 3.

Based on the above discussion, we make the following two observations:

Observation 1: A single-hop overlay path between two overlay nodes does not represent a native transit policy violation.

Observation 2: When the overlay nodes are located in stub domains (domains with no clients), every multi-hop overlay path in which the relay nodes are not data consumers represents a transit policy violation.

We address the following two important questions in the next two sections:

- What is the extent of native layer policy violation in a typical overlay routing situation? - Section III
- If the native routing policies are enforced and we disallow certain routes, how much is the overlay routing efficiency affected? - Section IV

III. Characterizing Overlay Violations

In this section, we provide insights into the extent of native policy violations in overlay networks. We do this using an experimental case study overlay network deployed over Planetlab[1]. We first describe the overlay case study and investigate the characteristics of its overlay paths. Next, we evaluate the performance gains that overlay routing provides. Lastly, we examine the extent of policy violations that show up in the case study.

TABLE I

Multi-hop paths in Planetlab measurements

Metric	**Measurement Scheme**	**# multi-hop paths (of 3306 total paths)**	**%**
Hop count	Scriptroute[18]	394	11.91
Latency	Ping RTT	1868	56.50

(a) Summary of number of multi-hop paths

Direct	Hop count of multi-hop paths						
	2	3	4	5	6	7	8
1438	1053	375	315	85	20	17	3
43.5%	31.9%	11.3%	9.5%	2.6%	0.6%	0.5%	0.1%

(b) Latency-based multi-hop paths

A. Overlay Network Case Study

We choose 58 Planetlab nodes that are geographically distributed (based on latitude/longitude) over North America, with only one node per AS. We refer to the AS in which an overlay node is located as the *host AS*.

We assume complete mesh connectivity of overlay links between the overlay nodes, for all the results presented in this paper. Following standard terminology, an *overlay link* represents the direct native route between two overlay nodes, which in turn comprises one or more native links, and an *overlay path* is made up of one or more overlay (virtual) links. This overlay path represents the end-to-end route taken by the overlay application traffic. There are a total of 3306 overlay paths possible in our topology.

The best overlay path between two overlay nodes may not always be the direct path. In many cases, it is beneficial to route the overlay connection through other overlay nodes, than adopt the direct route[16], [17]. Such a decision is typically made by running a routing algorithm at the overlay layer using application-specific routing performance objectives.

For the case study overlay network, we ran a shortest path routing algorithm using hop counts and latency. Table I(a) presents the different multi-hop overlay paths observed. Table I(b) further classifies such multi-hop paths when the routing metric is latency. It is interesting to note that almost 56.5% of the overlay paths use a multi-hop route. This provides us ample data to analyze. Moreover, end-to-end latency is a metric that many applications would like to optimize. Hence, through the rest of the case study analysis, we study the violations observed for the particular routing metric of latency.

B. Overlay Routing Performance

Here we address the performance improvement obtained in our case study through the use of multi-hop overlay routes. We quantify the efficiency of overlay routing by using the *gain* metric. The gain achieved for a path is defined as:

$$\text{Gain for path AB} = \frac{(\text{Overlay link latency})_{AB} - (\text{Overlay path latency})_{AB}}{(\text{Overlay link latency})_{AB}}$$

where the $(\text{Overlay link latency})_{AB}$ is the latency of the direct native route between nodes A and B, and $(\text{Overlay path latency})_{AB}$ is the latency of the shortest path through the overlay network between nodes A and B. Note that $(\text{Overlay link latency})_{AB} \geq (\text{Overlay path latency})_{AB}$.

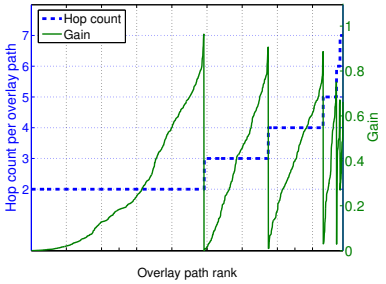


Fig. 4. Relation between gain observed for each overlay path and the overlay path hop count.

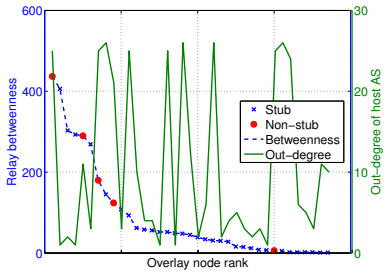


Fig. 5. The betweenness of each overlay node, plotted along with the out-degree of its host AS.

The gain metric represents the reduction in end-to-end latency achieved relative to the native route. The value ranges between 0, when the direct overlay link is the optimal one, and 1, when the multi-hop overlay path latency (which is the optimal one) is very small relative to the direct path.

Fig. 4 plots the values of gain observed for each multi-hop overlay path in our data set, sorted based on the hop count of the overlay path and ordered in the increasing order of gain. In the same graph, we plot the corresponding hop count of that overlay path. We observe that the individual curve for each hop count is similar and comparable. This indicates that in our case study there is not much dependence between the hop count and the gain. In other words, a higher hop count does not indicate a lower gain as one would guess.

The *betweenness* of a node is the number of overlay shortest paths that pass through it[19]. Fig. 5 plots the values of (non-zero) betweenness that were observed for each overlay node in our data set, sorted based on the decreasing value of betweenness. We clearly observe a non-uniformity in the relay popularity. From the shape of the betweenness curve, we conclude that there are a few overlay nodes that are the main reason for multi-hop overlay paths being preferred over the direct route. In the figure, we also mark the nodes located in stub domains and those located in non-stub domain. We can note that our dataset has a number of non-stub domain nodes with high betweenness. This is the main reason for the high percentage of non-violating overlay paths (as seen in the following subsection). However, the number of overlay nodes located in non-stub domains in our dataset is not restricted to those indicated in the figure, as the figure plots only nodes with non-zero betweenness that are currently being used in the overlay paths. In Fig. 5, we also plot the value of out-degree for its host AS and its siblings. We do not observe a particular correlation between the betweenness and the AS out-degree, which indicates that overlay routing is strictly driven by the latency-based edge costs and not the number of AS neighbors.

C. Overlay Policy Violations

Estimating the extent of policy violations in our case study requires the end-to-end AS-level path of each overlay path and the individual relationship between each consecutive pair of ASes in the end-to-end AS-path.

1) Inferring AS Path: The Scriptroute[18] data from the Planetlab measurements provides the IP address of each native hop in the overlay link[2]. We use the publicly available IP-prefix-to-AS mapping generated by the dynamic algorithm in [20]. This work primarily extracts the origin AS of each IP prefix from the BGP routing tables (from sources like the Routeviews servers[21], RIPE servers[22]) and refines the entries further using a dynamic algorithm. By performing a longest prefix match, we obtain the IP-to-AS mapping for each of the native hops. We also cross-verified our IP-to-AS mapping with those generated by *undns* in the Scriptroute tool.

After resolving the AS number of each IP hop in the overlay link, we have the end-to-end AS-path for each overlay path. Any usage of the term AS-path, henceforth, will represent the sequences of ASes in an overlay path, derived by concatenating the AS-path of individual overlay links, unless specified otherwise.

2) Obtaining AS Relationships: In order to obtain AS relationships, we adopt Gao's algorithm[15], supplemented by the partial AS relationship information[23], and our own heuristics to eliminate most of the algorithm's inaccuracies. Gao's algorithm makes inferences based on the AS-paths extracted from the BGP tables and identifies each relationship as being either a customer-provider relationship, a peering relationship, or a sibling relationship. The output from Gao's algorithm is more accurate when we input a more complete view of the AS-paths currently used in the Internet. Hence, as suggested in [24], we obtained the BGP tables from 6 RouteViews servers[21], 14 RIPE RCCs[22], 30 public routeservers and 1 lookingglass server[25].

As the algorithm is forced to use heuristics to classify some of the ASes as a provider, the relationships inferred are not guaranteed to be error-free. Firstly, it has been established that Gao's algorithm does not have a good level of accuracy with inferring peering and sibling relationships[23]. Secondly, the BGP table does not necessarily have information about all the possible inter-AS connections, as some ASes (representing stub networks that are most often simple customers) do not export routes to its peers. To solve these issues, we apply three corrections to the output of Gao's algorithm:

- Partial AS relationship information extracted from the RADB and the RIPE databases of the Internet Routing Registries (IRR)[26]. We followed a procedure similar to that in [23] to obtain these partial relationships.
- Implications from observation 1 that *the AS-path of all overlay links must be valley-free*. For instance, Gao's algorithm inferred that Global Crossing is a customer of Level3 Communications. However, this inference violated the valley-free property of the AS-path of some overlay links. Hence, we resolved the relationship according to what might lead to the valley-free property.
- Hypothesis that unknown relationships between a pair of stub ASes are peering relationships.

3) Violations Observed: We used the AS information obtained above to characterize the policy violations in our case study. The

[2]In certain anomalous cases, where the rockettrace did not work at certain nodes, we performed the traceroute operation.

TABLE II

ANALYSIS OF OVERLAY PATHS FOR POLICY VIOLATIONS

Type	Description	Number	%
A	Provider-AS-Provider	1925 relays	63.09
B	Provider-AS-Peer	74 relays	2.43
C	Peer-AS-Provider	61 relays	2.00
D	Peer-AS-Peer	73 relays	2.39
None	No violation	918 relays	30.09

(a) Summary of violating relay operations

Type	2	3	4	5	6	7	8
A	41.7	64.7	75.1	76.6	77.6	77.3	80.0
B	5.3	2.7	0.8	0.0	0.0	0.0	0.0
C	3.0	3.4	1.1	0.0	0.0	0.0	0.0
D	5.6	1.0	0.1	3.9	1.0	0.0	0.0
None	44.4	28.2	22.9	19.5	21.4	22.7	20.0

(b) Split up of violations for paths of a certain hop count

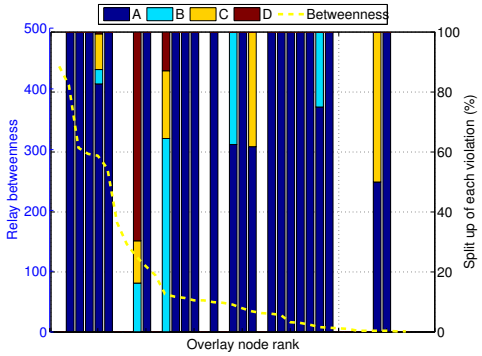


Fig. 6. Partitioning of the 4 different policy violations present at each relay

statistics are summarized in Table II(a). Note that each overlay path might commit multiple native policy violations, thereby giving a total of 2629 violations for 1868 multi-hop overlay paths. From the table, we observe that a predominant portion of the violations are of type A (as described in Section II), followed by those of type B. It is also worth noting that 30.09% of the 3115 intermediate relaying operations performed by the multi-hop overlay paths do not constitute a violation.

We also observed that about 30.19% of the 1868 multi-hop overlay paths do not commit any native policy violation. This is because all intermediate overlay nodes of those paths are located at a non-stub AS. In our dataset, all multi-hop overlay paths with more than 2 relays (3 hops) represent a violation.

Table II(b) shows the individual percentage of violations for overlay paths of different length (in terms of the overlay hop count). It is interesting to note that overlay paths with high hop count display more violations of type A, and rarely any of the other three types, implying that peering relationships are rarely present in long overlay paths in our case study. This could be because the peering relationships do not always offer a path with better latency, but rather offer paths with lower economic cost.

Fig. 6 presents the partitioning of the different violations observed in the domains that host the intermediate overlay nodes, along with the betweenness of the corresponding overlay node. We observe that most of the overlay nodes with high betweenness have a non-zero number of policy violations and most of these violations are of type A. We can observe from Fig. 5 and Fig. 6 that overlay nodes located at non-stub ASes do not commit native policy violations.

IV. EFFECT OF FILTERING ON OVERLAY PERFORMANCE

In Section I, we briefly described the motivation behind various administrative domains (AS) aiming to filter overlay layer traffic. Such filtering is propelled by the negative impact of overlay routing on the native layer, like: defeat of traffic engineering[27], [28], route instability[27], [28], [29], and eventually upsetting the economics of AS interconnection.

This filtering may defeat the purpose of overlay networks and eliminate the flexibility in overlay routing. Nevertheless, it is essential that the native layer have some basic control over such cases of policy violation and then exercise its discretion in determining what can or cannot be allowed. This need is illustrated by the number of commercial solutions that provide such overlay traffic filtering functionality[7], [8], [9].

We start with the premise that native network filtering is possible (Refer to Section VI for a survey of existing strategies) and consider two types of filtering – 1) *Blind filtering*, in which an AS blocks all overlay relaying through it, and 2) *Policy-aware filtering*, in which an AS blocks transit overlay traffic only if it violates native routing policies. Note that blind filtering may be easier to implement since it does not require knowledge of native routing policy.

When all ASes perform blind filtering, we observe that no overlay path can take a multi-hop overlay route between two end-points. This makes it essential that all overlay nodes are mesh-connected to ensure that all nodes can reach each other. However, when all ASes perform policy-aware filtering and disallow all types of native policy violations, we notice from results in Section III-C.3 that some (30.19% in our dataset) of the multi-hop overlay paths, which commit absolutely no policy violation, are not blocked. Hence, we still maintain the benefits derived from overlay routing. This situation is more desirable for the overlay network and its users.

ASes filter only relayed (multi-hop) traffic and do not filter overlay traffic that use the direct native route. Consequently, the filtering happens only at domains that host the intermediate overlay nodes of a multi-hop overlay path.

We now evaluate the impact of these two forms of filtering on overlay routing using a *penalty* metric. The penalty incurred for each path is defined as:

$$\text{Penalty for path AB} = \frac{\text{Post-filtering latency of overlay path AB}}{\text{Best possible latency of overlay path AB, assuming no filtering}}$$

The penalty value is a good indicator of the negative effect when an intermediate relay node in the shortest overlay path disallows relaying and the overlay traffic is forced to take a longer path.

We next use our Planetlab overlay, characterized earlier, as a case study to evaluate the effect of filtering on overlay routing performance.

A. Blind Filtering

In Fig. 7, we plot the average penalty (averaged over all multi-hop overlay paths) that would be incurred if the host AS of a particular overlay node blindly filters out overlay traffic, alongside the value of betweenness associated with that overlay node. To compute the value of the penalty for a particular overlay node, we rerun the shortest path algorithm after disallowing relaying at that node and compute new end-to-end latency values. When

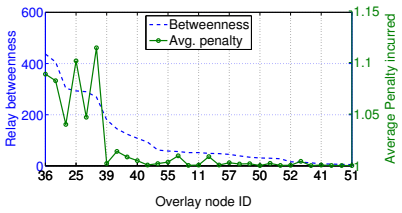


Fig. 7. Performance when overlay relaying is blindly disallowed at a particular host AS, one node at a time

an overlay node with high betweenness is disallowed, we seem to incur a high penalty. This indicates that few overlay nodes with high betweenness provide a substantial incentive to each overlay path passing through it and are quite irreplaceable. Hence, depending on whether an excessively used relay is being disabled the overall penalty incurred varies.

Fig. 8 presents plots for the penalty incurred and the number of violations committed when the number of ASes performing this filtering operation is varied. When the number of host ASes performing the filtering operation is n, we have $\binom{58}{n}$ possible scenarios. In cases where this number is over 1000, we chose 1000 scenarios at random. Each penalty measurement in Fig. 8 is an average of the penalty incurred by all overlay paths, and average over all scenarios considered. We also plot the 95% confidence interval for each value.

We observe, from Fig. 8 that, as expected, when more ASes hosting overlay nodes perform blind filtering of overlay traffic, the penalty incurred increases, while the number of violations decreases. We also observe a drastic drop in the number of violations as more than 50 host ASes begin filtering because the last few ASes with high betweenness are finally being targeted.

When all 58 host ASes begin filtering, the overlay paths are forced to use the single-hop overlay link without any relaying, which is equivalent to native routing. This leads to the following three consequences:

- The number of violations observed is 0.
- The penalty value is at a maximum of 1.838.
- There is no advantage to using overlays, as the overlay path is the same as the direct native route.

B. Policy-Aware Filtering

Fig. 9 presents plots for the penalty incurred and the number of violations committed when native policy violations are disallowed by a certain number of host ASes. We use the same evaluation methodology as that described in the previous subsection.

Similar to blind filtering, the penalty incurred increase and the number of violations decrease, with an increase in the number of host ASes performing the policy-based filtering.

When all 58 host ASes begin filtering, the violating overlay paths are forced to use the direct route without any relaying. Hence, the number of violations is 0. However, this does not imply that the gain is 0, as some multi-hop overlay paths are still allowed. In our dataset, we observe an average gain of 13.49% in this scenario (in contrast to 31.81% in the case where there was no filtering). The penalty value is at a maximum of 1.49.

When we compare the results in Fig. 8 and Fig. 9, we notice that policy-based filtering incurs substantially lower penalty compared to blind-filtering and a non-zero gain, making it still worthwhile to deploy overlays.

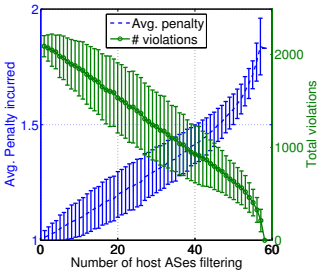


Fig. 8. Blind filtering

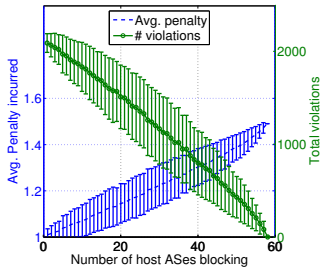


Fig. 9. Policy-aware filtering

V. Mitigating the Impact of Filtering

In the previous section, we investigated the effects of policy enforcement at the native layer. When the overlay traffic relaying is disallowed at the native layer, we notice a substantial deterioration in performance of the overlay networks. This incapacitation causes the overlay traffic to experience the same or worse treatment as traffic generated by other applications. It is also conceivable that the overlay network avoids using overlay paths with native policy violations in an effort to appease the underlying native network. This tends to have the same drop in overlay routing gain as filtering does.

In the context of service overlays, we propose two possible options that can allow the overlay regain some of its performance advantage, albeit at a cost:

- Add more overlay nodes at non-stub networks, so that good non-violating overlay paths can be created.
- Negotiate deals with ASes traversed by violating shortest overlay paths, so that overlay traffic is allowed to pass through. This in essence creates an overlay policy that supplements the native policy, but is independently managed.

In this paper, we consider a generalized approach where the two schemes above are deployed individually or in combination. In this general scheme, new nodes may be deployed in some parts of the network to create non-violating paths, while ASes in other parts of the network are paid to allow violations. One can also think of the two schemes used simultaneously: an overlay node is added to create some good violating paths and the ASes are paid to allow these paths. By adopting these two approaches, we obtain overlay paths that are better than what is achieved when all ASes perform policy-aware filtering.

This solution allows the overlay service provider (OSP) to share the cost originally incurred by the native network in return for obtaining a routing performance advantage. Hence, we refer to it as the *cost-sharing* approach. It is conceivable that such an approach can be adopted to relax any objection raised by the native network, thereby fostering a higher level of economic cooperation between the two layers. Adopting this cost-sharing approach is crucial to put an end to the selfish conflict between the two layers, which often leads to a deterioration in routing performance[30].

The cost-sharing solution involves two basic costs that are paid by the overlay service provider to the native network, *over the lifetime of the overlay*[3]:

- New node fee, N_i: Cost for adding a new overlay node in native AS i and for the associated network resource usage.
- Permit fee, P_i: Cost for making a native AS i allow the violating overlay traffic to be relayed through its network.

[3]We avoid usage-based billing to remove effects of traffic variability.

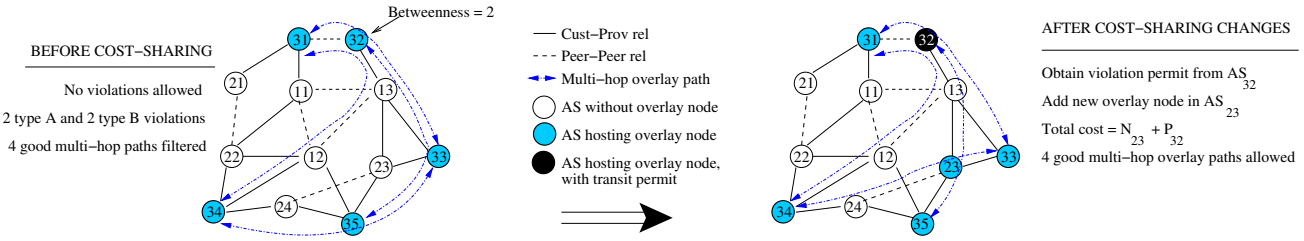


Fig. 10. Illustration of the cost-sharing approach, where we pick the optimal set of ASes to use for relaying. In the above figure, the AS numbers indicate which AS is the provider and which is the customer. For instance, an AS with AS number 12 will be a provider of AS with AS number 22.

Fig. 10 illustrates the cost-sharing approach in a typical overlay network spread over multiple native ASes. The figure shows the transition from an original network with four violating overlay paths to a network where these paths have been "legalized", by adding a new node to AS_{23} and obtaining a transit permit from AS_{32}. By making these purchases, the overlay service provider obtained 4 multi-hop overlay paths with good performance. Moreover, there were no new violations caused when the new overlay node was added into the overlay topology because AS_{23} is a non-stub domain. Consider for example the route between the overlay node in AS_{34} and that in AS_{33}. The shortest overlay path (AS_{34}-AS_{24}-AS_{35}-AS_{23}-AS_{33}) constitutes a violation of type A and is disallowed by the native layer. This causes the overlay route to adopt the direct native route AS_{34}-AS_{12}-AS_{13}-AS_{33}, which is substantially longer. However, the cost-sharing approach helped obtain a better route through the new node in AS_{23}. This leads to the new overlay route AS_{34}-AS_{24}-AS_{23}-AS_{34} and a corresponding path gain.

A. Deploying the Cost Sharing Scheme

We now consider the question of how to best deploy the cost-sharing scheme described above. The problem we address is the following: *Given a certain budget, new node costs and permit costs, how should an overlay service provider determine where to position new nodes and what permits to obtain, in order to maximize its performance advantage within the constraints of the budget.*

Based on this problem statement, we can see that the solution to the cost-sharing approach is comprised of two components:

$$\mathcal{N} = \text{Set of ASes where new nodes are placed}$$
$$\mathcal{P} = \text{Set of ASes being paid for permits}$$

We represent the overall solution as $\mathcal{S}$, where $\mathcal{S} = \{\mathcal{N}, \mathcal{P}\}$. A solution yields a new set of shortest paths in the overlay, $\mathbb{H}$. This set is made up of a set of non-violating or permitted paths, $\mathbb{H}_N$ and a set of violating paths, $\mathbb{H}_V$. The overlay paths in $\mathbb{H}_N$ can provide a gain in routing performance over native routing (as described in Section III-B). However, the violating paths in $\mathbb{H}_V$ cannot be used (because of the assumption that these are filtered) and the overlay resorts to using the single-hop overlay link. This provides no advantage to overlay routing. An ideal solution, hence, is where $\mathbb{H}_V = \emptyset$.

The cost-sharing deployment problem can be formulated as:

$$\max_{\mathcal{S}} \text{ Gain}(\mathcal{S}) \text{ such that Cost}(\mathcal{S}) \leq B, \text{ where}$$
$$B = \text{Budget allocated}$$
$$\text{Cost}(\mathcal{S}) = \sum_{i \in \mathcal{N}} N_i + \sum_{j \in \mathcal{P}} P_j$$
$$\text{Gain}(\mathcal{S}) = \frac{\sum_{i \in \mathbb{H}_N} \text{Gain}(i)}{|\mathbb{H}|}$$

To solve the cost-sharing problem in the context of inter-domain policy violations and policy-aware filtering, we need the following details about the native network and the original overlay network[4]:

- Overlay network topology (node location, link connectivity).
- Estimated length of each overlay link, based on the metric of choice.
- The AS-level path of each overlay link and the relationships between each pair of AS present in the AS-level path. This helps us determine which relaying operations are filtered by the native layer.
- The hypothetical shortest overlay path computed without concern for inter-domain violations. We denote these paths as $\mathbb{H}'$. They represent the highest achievable gain. These are the routes we characterized in Section III.
- The costs involved in adding nodes and for obtaining permits from ASes, computed from various native-overlay business agreements.

The cost-sharing problem is complicated because of the policy constraints that need to be satisfied. Moreover, the gain value is non-additive with respect to the different ASes used for relaying, i.e., if we know the gain $G1$ achieved when only $AS1$ is paid money for relaying and the gain $G2$ achieved when only $AS2$ is paid money for relaying, we cannot say that the gain achieved when both $AS1$ and $AS2$ are used for relaying is ($G1$ + $G2$). This makes our problem different compared to other conventional weight-constrained shortest path problems[31], [32], [33].

Obtaining an optimal solution $\mathcal{S}$ is a hard problem[5]. Hence, we use insights from our analysis in Section III and IV to derive greedy heuristics to obtain a reasonable solution.

B. Greedy Heuristic Solution

Our heuristic solution is shown in Fig. 11. It produces the solution in two phases. In the first, it obtains permits for violating paths in a particular order, until the cost of buying permits exceeds a threshold value B_{th}, which is less than or equal to the total budget B. In the second phase, the remaining budget (if any) is used to add new nodes to provide more non-violating paths. The ordering of the two phases is motivated later. The order of consideration of the individual ASes in each phase is motivated by the following insights that we obtained from our previous analysis:

1. We observed in Section III that most of the violations are in the form of a transit to an upstream provider (Type A and B). Hence, it is desirable to add overlay nodes at these

[4]Most of these can be obtained by a procedure similar to that we adopted in Section III.

[5]The cost-sharing problem can easily be shown to be NP-hard by performing a reduction to the set-cover problem, which is known to be NP-complete[34].

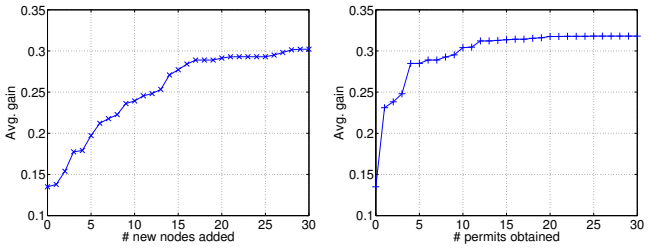
- Set $\mathcal{N} = \mathcal{P} = \emptyset$
- For each path i in $\mathbb{H}'$
  - For each relay node j in path i, betweenness(j)++
  - Compute Gain(i)
- Sort all overlay nodes in decreasing order of $\frac{\text{betweenness}}{P \text{ for host AS of node}}$
- while total_cost $< B_{th}$
  - j = Get Next Entry(Sorted list of overlay nodes)
  - k = Host AS of node j
  - Obtain permit from AS k
  - $\mathcal{P} = \mathcal{P} \cup k$
  - total_cost = total_cost + P_k
- Sort all overlay paths in $\mathbb{H}'$ with violation A/B in decreasing order of $\frac{\text{path gain}}{N \text{ for upstream provider AS}}$
- while total_cost $< B$
  - i = Get Next Entry(Sorted list of overlay paths)
  - k = Upstream provider AS in path i
  - If AS $k \in \mathcal{N}$, continue
  - If there exists no path between AS k and destination of path i, continue
  - Add new node to AS k
  - Compute latency between the new overlay node and the original ones
  - $\mathcal{N} = \mathcal{N} \cup k$
  - total_cost = total_cost + N_k
- Solution $\mathcal{S} = \{\mathcal{N}, \mathcal{P}\}$

Fig. 11. Greedy scheme for the cost-sharing problem.

upstream providers (relative to the point of violation), so as to bypass the overlay node associated with the violation. Our heuristic, therefore, adds overlay nodes to intermediate ASes in the unconstrained shortest overlay paths $\mathbb{H}'$, starting with the violating overlay paths which achieve the highest gain. If there exists an upstream provider in a violating path with very low new node fee N, then it would be in our best interest to give a higher preference to such placing a node there. We achieve this by normalizing the value of path gain by the new node fee for the upstream provider (unless the cost is zero, for which we just use the absolute value), as done in the approximation algorithm for the set-cover problem[35].

2. From the results in Section III, we know that most of the violations are committed at stub ASes hosting overlay nodes. Moreover, betweenness plots in Section III indicated that there are a few overlay nodes that are key to most of the overlay paths. Hence, by merging both observations, we negotiate deals with the stub ASes, in the decreasing order of relay betweenness in $\mathbb{H}'$, to permit the violating overlay traffic to be relayed through. Similar to the previous discussion, we normalize the betweenness value of a particular overlay node by the permit fee for the corresponding host AS.

When threshold B_{th} = budget B, we only obtain permits to improve overall gain. When threshold B_{th} = 0, we only add new nodes to improve overall gain. This shows that the threshold value B_{th} has a direct influence on the effectiveness of the cost-sharing approach, by controlling the decision of which heuristic to follow. Generally, the ideal threshold value can be found from the betweenness plot in Fig. 5, which shows that only a few nodes are repeatedly present in many overlay paths. Hence, we can look for a knee point in the betweenness curve to determine the appropriate threshold value. Based on the betweenness values observed in our case study and in other simulated overlay networks, we



(a) Only add new nodes (b) Only obtain permits

Fig. 12. Average gain achieved with the two individual heuristics.

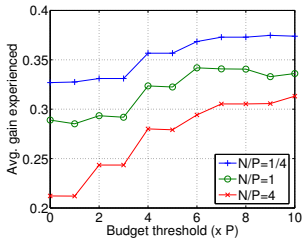


Fig. 13. Solutions for different values of B_{th}, when budget $B = 20 \times P$

recommend the following rule of thumb for setting the threshold value:

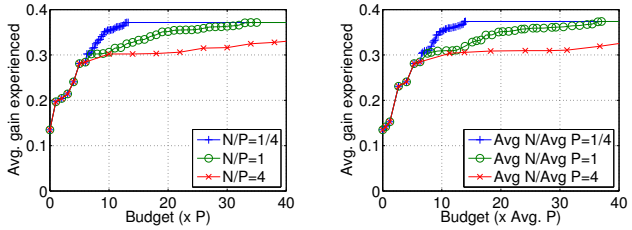
$$B_{th} \approx (\#\text{ relays with betweenness} > \frac{\text{max. betweenness}}{2}) \times P$$

C. Applying the Heuristic to Our Case Study

In this subsection, we show results from applying our heuristic to the overlay case study analyzed previously. In these results, we assume that the permit fee is the same for all ASes ($P_i = P \ \forall$ AS i) and the new node fee is the same for all ASes ($N_i = N \ \forall$ AS i). In Fig. 12(a) and 12(b), we first show the effect of adding nodes and permits, respectively, in the order specified by the individual heuristic. We make two observations about the heuristics. First, as expected, the ordering of new nodes to add and permits to obtain give the desired effect of producing the best gain in the first few nodes added or permits obtained. Second, we observe that the gain from the initial few permits can be substantial. Hence our decision to obtain permits first and then add nodes in the greedy algorithm of Fig. 11.

We next applied the cost-sharing algorithm for different values of the threshold value, while keeping the overall budget B at a constant value of $20 \times P$. We plot, in Fig. 13, the solutions obtained for different ratios of N/P. We can see that the knee of each curve lies around a threshold value of 5 or 6. This is coherent with our rule of thumb. We observed in Fig. 12(b) that the gain achieved by obtaining permits saturates after a certain point. Hence, having a high threshold value and filling up the budget with permit expenses is not desirable because we lose on any potential gain that can be achieved by adding new nodes. Keeping this in mind, we varied the threshold value only between zero and $B/2$.

We next consider the effect of the total budget B on the achievable performance. Fig. 14(a) shows the performance when we apply our greedy algorithm for $B_{th} = 6 \times P$, with varying budgets. The ratio between the new node fee and the permit fee determines how effectively the remaining budget will be utilized after obtaining sufficient permits. Therefore, it has a direct bearing on the achieved gain. As expected, for a fixed budget



(a) When P is equal ∀ ASes and N is equal ∀ ASes.

(b) When P and N for each AS is uniformly distributed.

Fig. 14. Solutions with the cost-sharing greedy scheme, when $B_{th} = 6 \times P$

threshold, the higher the N/P ratio, the lower the gain achieved. By comparing the plots in Fig. 12 and Fig. 14(a), we observe two important points - i) the greedy algorithm performs better than the individual heuristics for each value of the budget, ii) the highest gain achieved in the cost-sharing scheme is greater than what is achieved in $\mathbb{H}'$ (which is the case where all ASes permit violations). These two observations corroborate our combined cost-sharing approach.

All previous experiments assumed equal N_i and P_i. For the same overlay case study, we computed the solution $\mathcal{S}$ for a random distribution of the costs N_i and P_i. The costs were uniformly distributed between $[0.5\times N,\ 1.5\times N]$ and $[0.5\times P,\ 1.5\times P]$ respectively, thus maintaining the average values at N and P. Fig. 14(b) presents the gain achieved in this scenario. We observe that the gain achieved for a certain budget is comparable with the earlier simplified scenario. This shows that the algorithm is more influenced by the average costs, rather than the absolute value.

D. Network Characteristics

Our cost-sharing approach improves on a given scenario, without creating any new policy violations, by exploiting the following two properties:

1) The property that there exists non-stub ASes that can offer a good route to a destination.
2) The betweenness property of overlay nodes, wherein there exists a small set of overlay nodes that are present in many overlay paths.

We understand that many of the conclusions drawn in this paper may seem limited by the fact that they originate from a single Planetlab dataset. In this subsection, we establish the generality of our approach by showing that these two properties hold true in a wide variety of networks.

The first property can easily be reasoned to be true based on the knowledge that inter-domain routing is policy-constrained and does not always adopt the shortest route to a destination. By adding an overlay node at the upstream provider, we are able to force the AS to adopt the shorter route, thereby regaining the routing advantage.

To verify the second property, we simulated 90 random overlay networks with varying number of overlay nodes in stub ASes and varying out-degree (extent of multihoming) of the host AS. In particular, the number of stub ASes was set at 35, 40 or 45, and the maximum out-degree of the host AS was bound to 10, 25 or 40. This gives a total of 9 combinations, each of which was simulated 10 times.

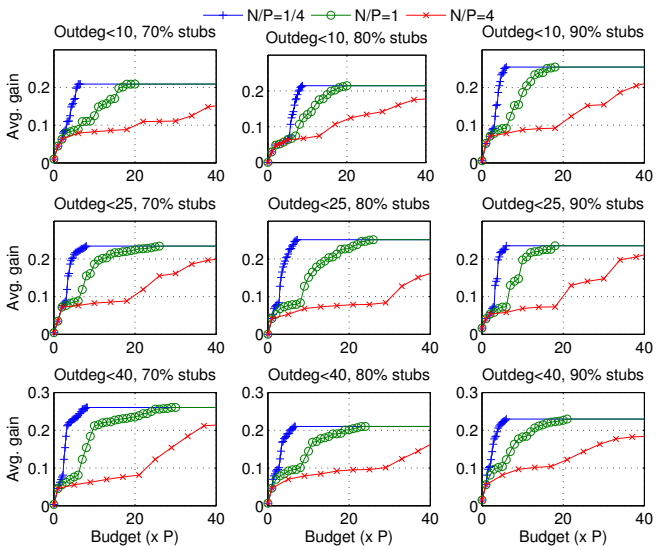


Fig. 15. Cost-sharing solutions for the 9 simulated scenarios

In each run, we picked 50 host ASes, catering to the above mentioned characteristics, from a list of 21,416 ASes observed in the different BGP route dumps used in Section III. We computed the AS-level route of each overlay link by using the BGP routes collected from multiple vantage points as input and performing policy routing between the two host ASes[6]. In addition, we assigned random latency values for each inter-AS link (in the range of 10-50 ms) and computed the shortest overlay path between each pair of host AS. When we inspected the betweenness of each overlay node, we observed that the betweenness property indeed holds in all scenarios.

Further, we applied our cost-sharing algorithm to each of the 9 scenarios and plotted the results in Fig. 15. The gain achieved was averaged over the 10 different overlay networks generated for each scenario. In all scenarios, we notice a sharp increase in the gain when the budget is low. After some point, though, adding more budget does not lead to significant gain improvement. As the initial permits obtained provided a higher gain than what the new node addition provided, our choice of B_{th} (according to the rule of thumb) in each run is justified. The individual plots in Fig. 15 are similar to those observed in our case study, indicating that our approach can be used to improve performance of many possible overlay topologies.

VI. Related Work

Our work is influenced by, or related to, research in the following three broad directions:

• **BGP measurement studies:** Our work classifies violations according to results in [14], which tries to understand the BGP misconfigurations that are prevalent in the current Internet. With advances in BGP measurement studies and data sources[15], [20], [21], [22], [23], [24], [25], [26], [23], it is now possible to determine the different AS policies endured by a packet exchanged between two end systems. We combine both these directions of work in the context of overlay routing to analyze violations of overlay traffic.

[6]We computed the shortest AS-path that does not violate native layer policy. This is an approximation as the actual routing tables and the policies are unavailable.

• **Native layer traffic management:** The second motivation of our work lies in the rapid development in the market for traffic management products[5], [6], [7], [8], [9], based on ASes' need to control the influx of overlay traffic. To identify and filter these overlay packets, most products adopt a flow-signature-based approach[36] that develops some form of correlation between the incoming and outgoing packets, or a communication-pattern-based approach[37]. However, there is very little understanding of the impact of filtering on the user experience. We address this issue in our paper. [4] addresses the impact of P2P networks on ISPs' costs. We address the impact of service overlays on inter-domain policies.

• **Overlay topology design studies:** Numerous studies have investigated the improvement in overlay routing performance achieved by careful placement of overlay nodes and links[38], [39]. Our work builds on top of the past research by using the basic overlay topology as an input to our analysis. The work in [38] performs a gain-cost analysis similar to ours, with the aim of picking the least number of servers and achieving the required gain. However, their work does not consider native policy restrictions. Two other efforts on overlay topology design focus on specific routing objectives - J. Han et al.[40] propose ways to aid the robustness of the overlay network, and H. Zhang et al.[41] propose ways to obtain optimal routing cost and utilization.

VII. Concluding Remarks

In this paper, we investigate the concern that overlay routing derives performance advantages by violating native routing policies. As more overlay applications are introduced to subvert the functionality limitations of the Internet, the frequency of policy violations can become substantial, which would increase the relevance of this work. To our knowledge, our study is the first to characterize the potential extent of native policy violations in overlay routing and to analyze the impact of native layer traffic filtering attempting to prevent these violations. We showed that a clear tradeoff exists between the number of policy violations and the penalty incurred when the native layer enforces these policies. It is conceivable that more networks will start filtering overlay traffic. We showed that even discriminating policy-aware filtering can be detrimental to the overlay routing efficiency, while blind filtering can completely remove any incentive to use overlay routing. In this context, we propose a cost-sharing approach that allows the overlay service provider to recover the overlay routing advantage through payments to native network operators. Further, we prescribed a heuristic-based algorithm for solving the cost-sharing problem with a certain budget. We believe that this approach provides a framework to legitimize native policy violations and allow the benefits obtained by the overlay to be directly related to costs incurred by the overlay service provider.

References

[1] "Planetlab," http://www.planet-lab.org.

[2] Aaron Falk, "Not quite the differentiated services I was thinking of," Note sent to e2e mailing list, October 2005.

[3] E. Mier, D. Mier, and A. Mosco, "Assessing Skype's network impact," *Network World*, vol. http://www.networkworld.com/reviews/2005/121205-skype-test.html, Dec 2005.

[4] T. Karagiannis, P. Rodriguez, and K. Papagiannaki, "Should Internet Service Providers Fear Peer-Assisted Content Distribution?," in *Proceedings of ACM Internet Measurement Conference*, Otober 2005.

[5] "Cachelogic's Cachepliance 4100," http://www.cachelogic.com/p2p/p2pchoices.php.

[6] "Sandvine's PPE 8200," http://www.sandvine.com/products/p2p_element.asp.

[7] "Verso Technologies' NetSpective P2PFilter," http://www.verso.com/enterprise/netspective/p2pfilter.asp.

[8] "Packeteer's PacketShaper," http://www.packeteer.com/prod-sol/solutions/p2p.cfm.

[9] "SonicWALL's Unified Threat Management," http://www.sonicwall.com/products/utm.html.

[10] Z. Duany, Z. Zhangy, and Y. Houz, "Service Overlay Networks: SLAs, QoS and Bandwidth Provisioning," in *Proceedings of ICNP*, Nov 2002.

[11] Salman A. Baset and Henning Schulzrinne, "An Analysis of the Skype Peer-to-Peer Internet Telephony Protocol," in *Proceedings of IEEE INFOCOM*, 2006.

[12] W. B. Norton, "A Business Case for ISP Peering," White Paper, http://www.equinix.com, February 2002.

[13] H. Tangmunarunkit et al., "The Impact of Routing Policy on Internet Paths," in *Proceedings of IEEE INFOCOM*, 2001, pp. 736–742.

[14] R. Mahajan, D. Wetherall, and T. Anderson, "Understanding BGP misconfiguration," in *Proceedings of ACM SIGCOMM*, 2002.

[15] Lixin Gao, "On inferring autonomous system relationships in the internet," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 733–745, 2001.

[16] D. Andersen et al., "Resilient Overlay Networks," in *Proceedings of 18th ACM SOSP*, October 2001.

[17] S. Savage et al., "Detour: a Case for Informed Internet Routing and Transport," Tech. Rep. TR-98-10-05, U. of Washington, Seattle, 1998.

[18] N. Spring, D. Wetherall, and T. Anderson, "Scriptroute: A facility for distributed Internet measurement," in *4th USENIX Symposium on Internet Technologies and Systems*, March 2003.

[19] L.C. Freeman, S.P. Borgatti, and D.R. White, "Centrality in Valued Graphs: A Measure of Betweeness Based on Network Flow," *Social Networks*, vol. 13, no. 141, pp. 141–154, 1991.

[20] Z. Mao, D. Johnson, J. Rexford, J. Wang, and R. Katz, "Scalable and accurate identification of AS-level forwarding paths," in *Proceedings of IEEE INFOCOM*, March 2004.

[21] "Route Views Project," http://www.routeviews.org/.

[22] "RIPE Routing Information Services," http://www.ripe.net/ris/.

[23] J. Xia and L. Gao, "On the evaluation of AS relationship inferences," in *Proceedings of IEEE GLOBECOM*, December 2004.

[24] Beichuan Zhang, Raymond Liu, Daniel Massey, and Lixia Zhang, "Collecting the Internet AS-level topology," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 35, no. 1, pp. 53–61, 2005.

[25] "Looking Glass servers," http://www.traceroute.org.

[26] "Internet Routing Registry," http://www.irr.net/docs/list.html.

[27] Y. Liu, H. Zhang, W. Gong, and D. Towsley, "On the Interaction Between Overlay Routing and Traffic Engineering," in *Proceedings of IEEE INFOCOM*, 2005.

[28] R. Keralapura et al., "Can ISPs take the heat from Overlay Networks?," in *Proceedings of HotNets-III*, November 2004.

[29] S. Seetharaman and M. Ammar, "On the Interaction between Dynamic Routing in the Overlay and Native Layers," in *Proceedings of IEEE INFOCOM*, April 2006.

[30] L.Qiu, R.Y.Yang, Y.Zhang, and S. Shenker, "On Selfish Routing in Internet-Like Environments," in *Proceedings of ACM SIGCOMM*, August 2003.

[31] P. Hansen, "Bicriterion path problems," in *Multiple Criteria Decision Making: Theory and Applications, LNEMS 177*, pp. 109–127. Springer-Verlag, Berlin, 1980.

[32] J.C.N. Clímaco and E.Q.V. Martins, "A bicriterion shortest path algorithm," *European J. of Operational Research*, vol. 11, 1982.

[33] H.W. Corley and I.D Moon, "Shortest paths in networks with vector weights," *Journal of Optimization Theory and Application*, vol. 46, no. 1, pp. 79–86, May 1985.

[34] M. R. Garey and David S. Johnson, *Computer and Intractability: A Guide to the Theory of NP-Completeness.*, W. H. Freeman, 1979.

[35] D. Hochbaum, *Approximation Algorihtms for NP-Hard Problems*, Brooks/Cole Publishing Co., 1996.

[36] Kyoungwon Suh, Daniel Figueiredo, James F. Kurose, and Don Towsley, "Characterizing and detecting relayed traffic: A case study using Skype," in *Proceedings of IEEE INFOCOM*, April 2006.

[37] T. Karagiannis et al., "BLINC: multilevel traffic classification in the dark," in *Proceedings of ACM SIGCOMM*, 2005.

[38] S. Shi and J. Turner, "Placing Servers in Overlay Networks," *SPECTS*, July 2002.

[39] Z. Li and P. Mohapatra, "The Impact of Topology on Overlay Routing Service," in *Proceedings of IEEE INFOCOM*, March 2004.

[40] J. Han, D. Watson, and F. Jahanian, "Topology Aware Overlay Networks," in *Proceedings of IEEE INFOCOM*, Mar 2005.

[41] H. Zhang, J. Kurose, and D. Towsley, "Can an overlay compensate for a careless underlay?," in *Proceedings of IEEE INFOCOM*, April 2006.