Interdomain Traffic Engineering with Redistribution Communities

B. Quoitin [a], S. Tandel [a], S. Uhlig [b], O. Bonaventure [b,*],

[a] *Infonet group, University of Namur, Belgium*

[b] *Dept. CSE, Catholic University of Louvain, Belgium*

Abstract

Various traffic engineering techniques are used to control the flow of IP packets inside large ISP networks. However, few techniques have been proposed to control the flow of packets between domains. In this paper, we focus on the needs of stub ISPs that compose 80% of the global Internet. We first describe the typical traffic pattern of such a stub ISP. Then we briefly describe the methods that those ISPs currently use to engineer their interdomain traffic. We focus on the control of the incoming traffic and show the increasing utilization of the BGP Community attribute and discuss the limitations of this approach. We then propose the redistribution communities as a deployable method to control the flow of the incoming interdomain traffic. We implement those redistribution communities in zebra and evaluate its performance.

Key words: Traffic engineering, Border Gateway Protocol (BGP)

1 Introduction

Initially developed as a research network, the Internet has been optimized to provide a service where the network does its best to deliver packets to their destination. In the research Internet, connectivity was the most important issue. During the last years, we have seen a rapid growth and an increasing utilization of the Internet to carry business critical services such as e-commerce, Virtual Private Networks and Voice over IP. To efficiently support those services, several Internet Service

* Corresponding author.
Email addresses: bqu@info.fundp.ac.be (B. Quoitin), sta@info.fundp.ac.be (S. Tandel), suh@info.ucl.ac.be (S. Uhlig), Bonaventure@info.ucl.ac.be (O. Bonaventure).

Providers (ISP) rely on traffic engineering techniques to better control the flow of IP packets.

During the last years, several types of traffic engineering techniques have been developed [1]. Most of these techniques have been designed for large IP networks that provide transit services and need to optimize the flow of IP packets inside their internal network. These techniques are of very limited use for smaller IP networks that constitute most of the Internet today [2]. For these networks, the costly resource that needs to be optimized is usually their interdomain connectivity. Although some BGP-based techniques are used today ([3,4]), it is difficult t use them to efficiently control the flow of the incoming interdomain traffic. In this paper, we propose a deployable technique that can be used to provide useful traffic engineering capabilities targeted at, but not limited to, those small stubs.

This document is organized as follows. We first discuss in section 2 the requirements for implementable and deployable interdomain traffic engineering techniques. Then, we briefly describe in section 3 the existing interdomain traffic engineering techniques. We propose the utilization of the redistribution communities in section 4 to control the flow of the incoming interdomain traffic and modify the `zebra` BGP daemon to support those communities in section 5.

2 Requirements for interdomain traffic engineering

The Internet is currently composed of about 14.000 Autonomous Systems (AS) [5] and its organization is more complex than the research Internet of the early nineties. Those 14.000 ASes do not play an equal role in the global Internet. ASes can be distinguished on the basis of various characteristics like the connectivity one AS has with its peers, the services provided by one AS to its peers and the behavior of the users inside the networks of one AS.

First, ASes can be distinguished on the basis of their connectivity. [2] has shown that there are two major types of interconnections between distinct ASes: the *customer-provider* and the *peer-to-peer* relationships. The *customer-provider* relationship is used when a small AS purchases connectivity from a larger AS. In this case, the large AS agrees to forward the packets received from the small AS to any destination and it also agrees to receive traffic destined to the small AS. On the other hand, the *peer-to-peer* relationship is used between ASes of similar size. In this case, the two ASes exchange traffic on a shared cost basis. According to [2], the *customer-provider* relationship is used for about 95 % of the AS interconnections in today's Internet.

Relying on this connectivity, [2] makes a first characterization of ASes. There are basically two types of ASes: transit ASes that constitute the core of the Internet

and stub ASes which are regional ISPs or customer networks that do not provide transit. The core corresponds to about 15 % of the ASes in the Internet and can be divided in three different subtypes (*dense*, *transit* and *outer core* depending on the connectivity of each AS). Stub ASes correspond to 85 % of the Internet and they maintain only a few *customer-provider* relationships with ASes in the core and some *peer-to-peer* relationships with other small ASes.

In this paper, we do not address the traffic engineering needs of ASes in the core but rather focus on stub ASes. The interested reader is referred to [6] for a discussion of the needs of transit ASes in the core.

A second important element used to characterize an AS is the type of customer it serves. If the AS is mainly a content provider, it will want to optimize its outgoing traffic since it generates more traffic than it receives. On the other hand, if the AS serves a population of SMEs (Small and Medium Enterprises), dialup, xDSL or cable modems users, it will receive more traffic than it sends. Such ASes will typically only need to control their incoming traffic.

Another point to consider is the "topological distribution" of the interdomain traffic to be engineered [7]. Although the Internet is composed of about 14.000 ASes, a given AS will not receive (resp. transmit) the same amount of traffic from (resp. toward) each external AS. The characteristics of the interdomain traffic seen from a customer AS have been analyzed in details in [7]. In this paper, we have analyzed the characteristics of all the interdomain traffic received by two small ISPs based on traces collected during one week. The first trace was collected at the interdomain routers of BELNET, an ISP providing access to universities and research labs in Belgium in December 1999. The second trace was collected during one week in April 2001 at the interdomain routers of YUCOM, a Belgian ISP providing a dialup access to the Internet. This study revealed two important findings that are summarized in figure 1. First, the left part of the figure shows the percentage of the number of IP addresses that are reachable from the BGP routers of the studied ASes at a distance of *x* AS hops. This figure shows that for both studied ASes, most reachable IP addresses are only a few AS hops away. Second, the right part of figure 1 shows the cumulative distribution of the traffic received from each external AS during the studied week.

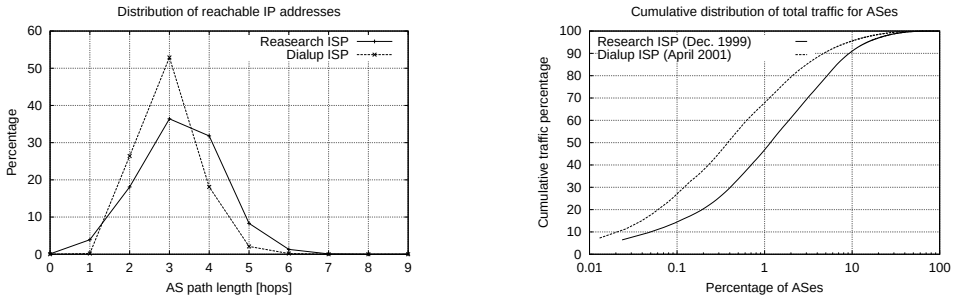

Fig. 1. BGP routing tables (left) and cumulative distribution of total traffic (right)

The figure shows that for both ASes, a small percentage of external ASes contribute to a large fraction of the incoming traffic. Hence, by influencing this limited percentage of ASes a large fraction of the traffic can be engineered. Similar findings were reported in [6] for an AS of the dense core.

3 Interdomain Traffic Engineering today

In this section, we review the traffic engineering techniques that are in use today in the global Internet. Since these techniques rely on a careful tuning of the BGP routing protocol, we first briefly review its operation.

3.1 Interdomain routing

The Border Gateway Protocol (BGP) [8,9] is the current de facto standard interdomain routing protocol. BGP is a *path-vector protocol* that works by sending *route advertisements* between Autonomous Systems (ASes). A route advertisement indicates the reachability of one IP network through the router that advertises it either because this network belongs to the same AS as this router or because this router has received from another AS a route advertisement for this network. Besides the reachable network, each route advertisement also contains attributes such as the `AS-Path` which is the list of all the transit ASes that must be used to reach the announced network.

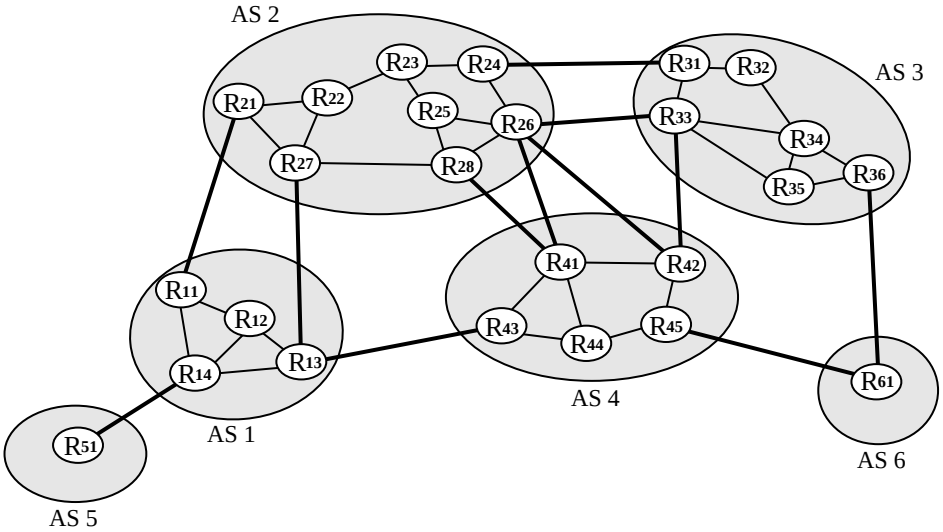

Fig. 2. A simple Internet

A key feature of BGP is that it supports routing policies. That is, BGP allows a router to be selective in the route advertisements that it sends to neighbor BGP

routers in remote AS. This is done by specifying on each BGP router a set of input and output filters for each peer.

3.2 BGP-based traffic engineering

The BGP-based traffic engineering solutions in utilization today rely on a careful tuning of the BGP decision process [1] that is used to select the best-route toward each destination. This process is based on a set of criteria that act as filters among all the BGP routes known by the router.

3.2.1 Control of the outgoing traffic

The control of the outgoing traffic is often a requirement for content providers that wish to optimize the distribution of their content. For this, they can rely on the `local-pref` attribute to control the routes that will be chosen for the packets that leave each BGP router of the content provider [4]. The actual distribution of the outgoing traffic will depend on the quality of the setting of the `local-pref` on the BGP routers of the AS. The setting of this parameter can be done manually based on the knowledge of the interdomain links or automatically with tools that rely on traffic measurements.

3.2.2 Control of the incoming traffic

A customer AS serving a large number of individual users or small corporate networks will typically have a very asymmetric interdomain traffic pattern with several times more incoming than outgoing traffic. These ASes typically need to engineer their incoming traffic only. For this, a first method that they can use is to announce different route advertisements on different links. For example in figure 2, if `AS1` wanted to balance the traffic coming from `AS2` over the links $R_{11} - R_{21}$ and $R_{13} - R_{27}$, then it could announce only its internal routes on the $R_{11} - R_{21}$ link and only the routes learned from `AS5` on the $R_{13} - R_{27}$ link. Since `AS2` would only learn about `AS5` through router R_{27}, it would be forced to send the packets whose destination belongs to `AS5` via router R_{27}.

A variant of the selective advertisements is the advertisement of more specific prefixes. This advertisement relies on the fact that an IP router will always select in its forwarding table the most specific route for each packet (i.e. the matching route with the longest prefix). This fact can also be used to control the incoming traffic. In the following example, we assume that prefix `16.0.0.0/8` belongs to `AS3` and

[1] Due to space limitations, we cannot detail the BGP decision process in this paper. A description of the BGP decision process may be found in [6,3,4].

that several important servers are part of the 16.1.2.0/24 subnet. If AS3 prefers to receive the packets toward its servers on the R_{24}-R_{31} link, then it would advertise both 16.0.0.0/8 and 16.1.2.0/24 on this link and only 16.0.0.0/8 on its other external links. An advantage of this solution is that if link R_{24}-R_{31} fails, then subnet 16.1.2.0/24 would still be reachable through the other links. However, an important drawback of advertising more specific prefixes is that it increases the number of BGP advertisements and thus the size of the BGP routing tables [10].

Another method would be to allow an AS to indicate a ranking among the various route advertisements that it sends. Based on the BGP decision process, one possible way to introduce a ranking between routes is to artificially increase the length of the AS-Path attribute. Coming back to our example, AS1 would announce the routes learned from AS5 on links $R_{11} - R_{21}$ and $R_{13} - R_{27}$, but would attach a longer AS-Path attribute (e.g. AS1 AS1 AS1 AS5 instead of AS1 AS5) on the $R_{13} - R_{27}$ link. The required amount of prepending is often manually selected on a trial and error basis. The manipulation of the AS-Path attribute is often used in practice ([10] reports that this affected 6.5 % of the BGP routes in November 2001). However, it should be noted that this technique is only useful if the ASes that we wish to influence do not rely on local-pref.

3.2.3 Community-based traffic engineering

In addition to these techniques, several ASes have been using the BGP Community attribute to encode various traffic engineering actions [11]. This attribute is often used to add markers to announced routes and to simplify the implementation of scalable routing policies on BGP routers. The community attribute is a transitive attribute that contains a set of community values, each value being encoded as a 32 bits field. Some community values are standardized (e.g. NO_EXPORT), but the Internet Assigned Numbers Authority (IANA) has assigned to each AS a block of 65536 community values. The community values are usually represented as ASx:V where ASx is the AS number to which the community belongs and V a value assigned by ASx. The community attribute is often used to encode the following traffic engineering actions [11] :

(1) Do not announce the route to specified peer(s);
(2) Prepend n times the AS-Path (where we have found values for n generally ranging from 1 to 3) when announcing the route to specified peer(s);
(3) Set the local-pref value in the AS receiving the route [12];

In the first case, the community is attached to a route to indicate that this route should not be announced to a specified peer or at a specified interconnection point. For example, in the left part of figure 3, AS2 has configured its routers to not announce to AS4 routes that contain the 2:1004 community. AS2 has documented the utilization of this community to its peers so that AS1 can attach this value to

the routes advertised to AS2 to ensure that it does not receive packets from AS4 through AS2. In a detailed survey of the RIPE whois database [11], we have shown that this type of communities was often used by ISPs.

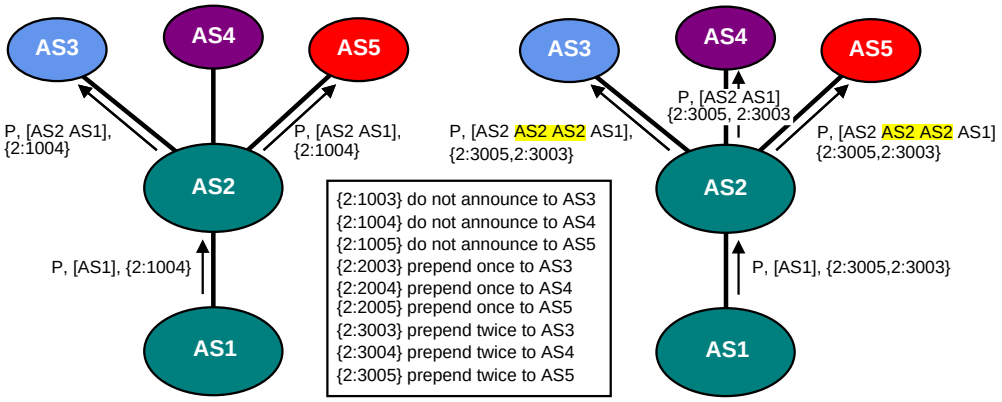

Fig. 3. Examples of traffic engineering with communities.

The second type of community is used to request the upstream AS to perform AS-Path prepending for the associated route. The right part of figure 3 shows how AS1 uses the 2:3003 and 2:2005 communities documented by AS2 to request that the AS-Path of the route it announced be prepended twice when announced to AS3 and AS5. To better understand the usefulness of such community values, let us consider again figure 2, and assume that AS6 receives a lot of traffic from AS1 and AS2 and that it would like to receive the packets from AS1 (resp. AS2) on the R_{45}-R_{61} (resp. R_{36}-R_{61}) link. AS6 cannot achieve this type of traffic distribution by performing prepending itself. However, this would be possible if AS4 could perform the prepending when announcing the AS6 routes to external peers. AS6 could thus advertise to AS4 its routes with the community 4:5202 (documented by AS4) that indicates that this route should be prepended two times when announced to AS2.

Finally, the third common type of community used for traffic engineering purposes is to set the local-pref in the upstream AS as described in [12].

4 Redistribution communities

The community based traffic engineering solution described in the previous section has been deployed by several dozens ISPs [11], but it suffers from several important drawbacks that limit its widespread utilization. First, each AS can only define 65536 distinct community values. While in practice no AS today utilizes more than 65536 community values, this limited space forces each AS to define its own community values in an unstructured manner. We note however that facing the need for structured community values, some ASes like AS9057 have started to utilize

community values outside their allocated space [11] and that other ASes are using community values reserved for standardization. This shows that there is a clear need for more structured community values.

A second drawback is that each defined value must be manually encoded in the configurations of the BGP routers of the AS. This is usually done by defining a filter for each supported community. In practice, a large number of filters are required to support complex policies. Defining a large number of such filters increases the complexity of the configurations. Furthermore, since those filters need to be applied to each BGP message, this can significantly increase the processing time of the BGP messages.

A third drawback is that each AS willing to utilize such communities must advertise the semantic of its own community values to external peers. Unfortunately, there is no standard method to advertise these community values. Some ASes define their communities as comments in their routing policies that are stored in the Internet Routing Registries. The RPSL language [13] used for these specifications does not currently allow to easily define the semantic of the community attribute values. Other ASes publish the required information on their web server or distribute it directly to their clients. This implies that an AS willing to utilize the traffic engineering communities defined by its upstream ASes needs to manually insert directives in the configurations of its BGP routers. Once inserted, these directives will need to be maintained and checked if the upstream AS decides for any reason to modify the semantics of some of its community values. This increases the complexity of the configuration of the BGP routers and is clearly not a desirable solution. A recent study has shown that human errors are already responsible for many routing problems on the global Internet [14]. An increasing utilization of this type of community-based traffic engineering would probably cause even more errors.

A fourth drawback of the BGP community attribute is its transitivity. It implies that once a community value has been attached to a route, this community is distributed throughout the global Internet. The growth of the BGP routing tables has been an important operational problem during the last few years due to the limited memory capacity of several deployed routers. Several studies [5] have analyzed the growth of the BGP routing tables based on the number of routes contained in those tables and have provided explanations for this growth. However, those BGP routing tables also store optional BGP attributes such as the community attribute that may contribute to the growth of the memory occupancy of those tables.

4.1 Current usage of the community attribute

To evaluate the impact of the communities on the size of the BGP tables, we have analyzed the utilization of the community attribute in the BGP routing tables col-

lected by RIPE RIS [15] and the Route Views projects [16] from January 2001 until January 2003 (The Route-Views project started in November 2001). For this analysis, we downloaded one BGP table dump per week until the end of June 2002 and then one BGP table per day. We parsed each BGP table to determine the number of routes containing community values and the number of distinct communities.

A first observation of those BGP table dumps reveals that although most of the community values have a local semantics [11], a large number of community values appear in the BGP routing tables. Figure 4 shows the evolution of the number of distinct community values found in those routing tables. The evolution of the utilization of the communities reveals a sustained growth since the availability of the first dumps with community information in January 2001. For instance, in recent dumps, we can find more than 900 different community values in routing tables collected by RIPE and more than 2300 distinct values in routing tables collected by Route-Views. This large number of distinct values shows the need for a structured and standardized encoding of the communities since there are many different utilizations of communities but also different values used for a similar meaning.

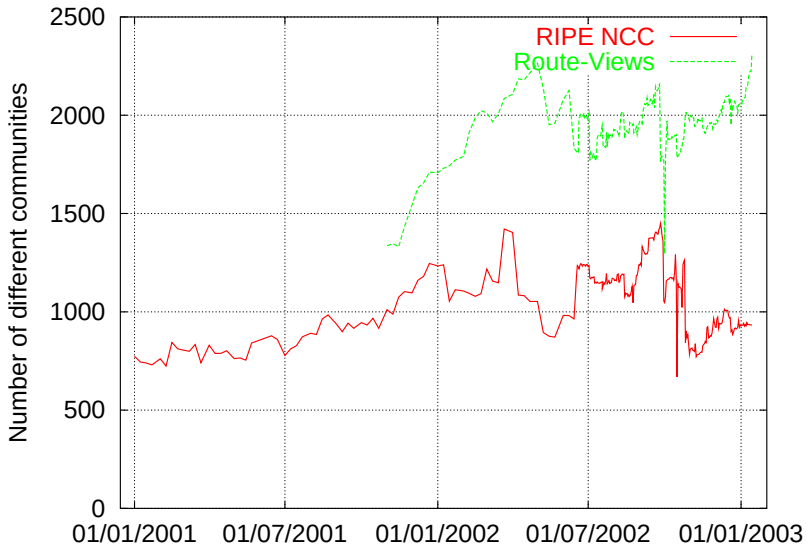

Fig. 4. Evolution of the number of distinct community values

Another important element to consider is the number of routes carrying the community attribute (figure 5 shows the evolution of the percentage of routes which carry at least one community). From January 2001 to January 2003, the number of routes in the RIPE tables has grown from 1.2M to nearly 1.4M routes while the number of routes which have at least one community attached has evolved from 366k (29%) to 568k (41%) routes. From November 2001 to January 2003, the size of the tables collected at Route-Views changed from 1.8M to 2.8M routes while the number of routes that carry at least one community has evolved from 1M (55%) to 1.5M (53%) routes. If we count the total number of communities present in the tables during the above periods, we can see that it has grown from 840k to 1.4M

communities at RIPE and from 2.2M to 5.5M communities at Route-Views. These large amounts are mainly due to the transitivity of the community attribute. Most of the community values contained in those tables have no meaning for the global Internet.

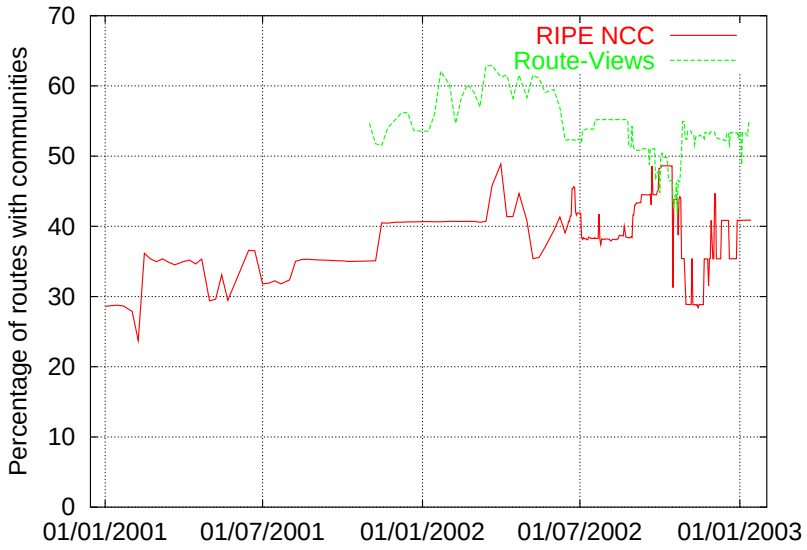

Fig. 5. Evolution of the percentage of routes carrying communities

4.2 The redistribution communities

In order to avoid the problems of the classical communities which we have identified in the above section, we propose a new type of extended community attribute, the redistribution communities. First, the extended community attribute defined in [17] provides a more structured and larger space, 8 octets, than the community attribute. Second, one flag in extended communities can be used to request non-transitivity. The redistribution communities we defined can be used to encode a set of redistribution actions that are applicable to a set of BGP speakers. The current definition of the redistribution communities [18] supports the following actions:

- the attached route should not be announced to the specified BGP speakers.
- the attached route should only be announced to the specified BGP speakers.
- the attached route should be announced with the NO_EXPORT community to the specified BGP speakers.
- the attached route should be prepended *n* times when announced to the specified BGP speakers.

The encoding of the redistribution action as well as the various ways to specify the target BGP speakers are defined in [18]. Compared with the classical communities, an interesting feature of the redistribution communities is that there are two methods to specify the BGP speakers affected by an action. The first method is

to explicitly list all those BGP speakers (e.g. their AS number) inside the redistribution communities. This method can be used for example when a route must be prepended when announced to a few specific ASes. The second method is to explicitly list only the eBGP speakers that will not be affected by the specified action. In this case, it is possible to specify that a route should be prepended once when announced to external peers except for a few specified peers.

It should be noted that the BGP redistribution communities that we propose are not equivalent to the `DIST_LIST_INCL` and `DIST_LIST_EXCL` attributes supported by ISO's InterDomain Routing Protocol (IDRP) [19]. IDRP's transitive `DIST_LIST_INCL` (resp. `DIST_LIST_EXCL`) attribute allows a domain to specify the list of domains that are allowed (resp. prohibited) to receive the attached route. Our redistribution communities support more actions, but are non-transitive. Using those two IDRP attributes in today's Internet would probably cause a severe security concern since a transit AS could create severe routing problems by modifying the content of such attributes inside received routes.

5 Implementation of the redistribution communities

In order to evaluate the cost of supporting the redistribution communities in a BGP router, we have modified the `zebra` BGP daemon [20]. The implementation of the redistribution communities requires two distinct functionalities. The first one is to allow a network operator to specify the redistribution communities that must be attached to given routes and the second one is to influence the redistribution of the routes that have such communities attached.

First, in order to allow a network operator to attach redistribution communities to routes, we have extended the `route-map` statement available in the command-line interface (CLI) of `zebra`. The `route-map` statement is an extremely powerful and versatile tool for route filtering and attribute manipulation that is composed of a filter and a list of actions. Our extension consists in the addition of a new action that can be used to attach a list of redistribution communities to routes that match the `route-map` filter. An example of a `route-map` using our new action is given below. The example presents the configuration in routers of `AS6`. This configuration attaches a redistribution community to every route announced to `AS4`. This community requests that `AS4` prepend 2 times the `AS-PATH` of routes announced by `AS6` when redistributing to `AS2` (see example in section 3.2.3).

```
neighbor <as4-neighbor-ip> route-map prepend2_to_as2
route-map prepend2_to_as2 permit 10
  match ip address any
  set extcommunity red prepend(2):as(2)
```

Then, we have modified `zebra` so that redistribution communities are automatically taken into account. The implementation extracts the redistribution communities attached to the route and on the basis of their content, decides to attach the `NO_EXPORT` community, to prepend n times or to ignore the route when redistributing to specified peers. This automatic treatment requires that all the redistribution communities attached to a route be treated together in order to check that conflicting actions are not taken in account. Moreover, it is possible to setup in different redistribution communities overlapping filters associated to different actions. In this case, the implementation is responsible to apply the action which implies the smallest amount of modifications to the treatment of the route, as stated in the specifications. These modifications in the source code of `zebra` were quite limited compared to the amount of work required to configure by hand redistribution policies similar to what redistribution communities provide.

5.1 Performance evaluation

In order to estimate the performance of the implementation of the redistribution communities in `zebra`, and to compare their efficiency to routing policies based on classical communities, we have performed black-box measurements of the time taken by a Linux machine running the `zebra` BGP daemon to propagate an UPDATE message received from one peer to other peers. For reference purpose, we first performed measurements with routes without the community attribute. Then we performed a second set of measurements with routes containing classical communities and a BGP router supporting the corresponding routing policies. Finally, we performed measurements with routes containing redistribution communities.

The testbed used for the measurements is shown in figure 6. Four Linux machines have been interconnected. In the center of the topology, the Device Under Test (DUT) runs a `zebra` BGP daemon supporting the redistribution communities. On the left, the *Source Router* also runs `zebra` and has an eBGP session established with the DUT. The *Source Router* produces test BGP UPDATE messages. On the right, the *Destination Router* is a PC which simulates the multiple downstream BGP peers of the DUT. The *Destination Routers* runs one `sbgp` daemon per simulated peer instead of full BGP daemons. Finally, the *Sniffer* runs `tcpdump` to capture the messages that arrive/leave to/from the DUT.

For our first measurements, we measured the time required to process the routes learned from and external peer and to announce them to a single *Destination Router*. Without communities or redistribution communities, the DUT required on average 0.5 msec to process a single BGP UPDATE message (bottom line in figure 8).

We then used classical and redistribution communities to request the DUT to perform AS-Path prepending. With the classical communities, the configuration of the

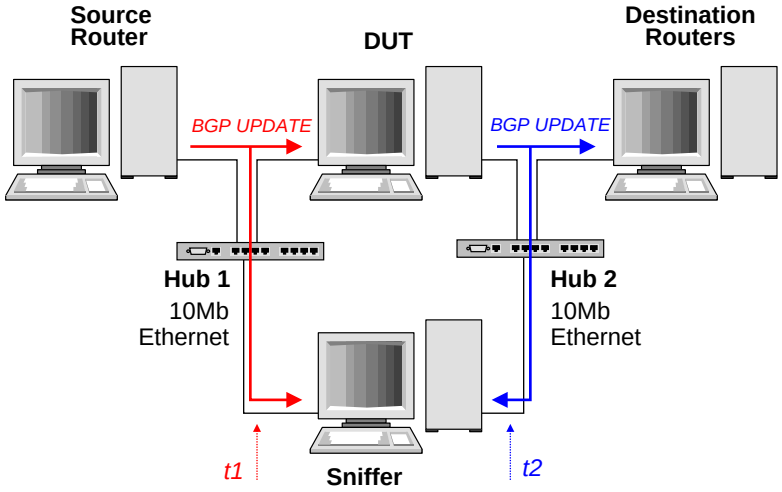

Fig. 6. Measurement testbed.

```
!
router bgp 65237
  bgp router-id 10.0.0.14
  timers bgp 0 0
...
  neighbor 10.0.10.1 send-community
  neighbor 10.0.10.1 route-map rm1 out
!
ip community-list rmc1 permit 501:1
!
route-map rm1 permit 1
  match community rmc1
  set as-path prepend 65237
...
route-map rm1 permit 8
```

Fig. 7. Configuration of the DUT with classical communities

DUT contains one `community-list` and one matching `route-map` statement for each supported classical community. Figure 7 shows an extract of the configuration of the DUT with those communities. No special configuration was required on the DUT to support the redistribution communities.

As expected, the utilization of optional BGP attributes increases the processing time. The measurements indicate that the DUT spends 0.69 msec on average to process and announce a route with a single classical community attached, 0.56 msec with a single redistribution community. With the classical communities, the time required to process and announce a route increases quickly with the number of distinct communities contained in the route and configured on the DUT. For example, the DUT spends 4.62 msec to process and announce a route containing 100 classical communities. On the other hand, the number of redistribution communities does not appear to increase significantly the processing time of the DUT. For instance,

the DUT spends on average 0.7 msec to process and announce a route containing 100 distinct redistribution communities.

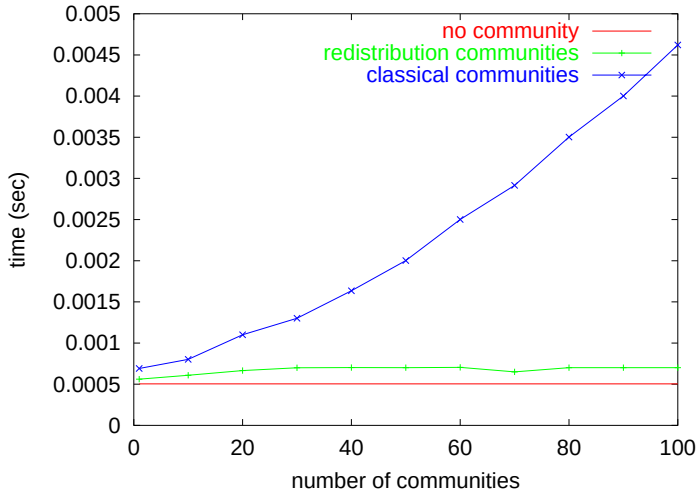

Fig. 8. Average time to process and announce routes to a single peer

We then performed a second experiment to model a router connected to a large number of peers such as on a public Internet eXchange (IX). For this experiment, we used a single community but varied the number of external peers. Figure 9 represents the average time to process and announce this route to m peers. This experiment shows that the utilization of redistribution communities does not affect the time to process and announce a route to m peers. However, when classical communities are used, this time increases quickly with the number of peers. For instance, with 50 peers, the average time taken by the DUT to process and announce a route is 17.28 msec with a single classical community and 87.5 msec with 100 classical communities. In the case of redistribution communities, these times are 6.55 msec and 10.3 msec respectively. We can conclude that the processing time is more sensitive to the number of communities attached when classical communities are used than with redistribution communities.

The important difference observed in the time taken to process and announce routes with classical or redistribution communities is mainly due to the requirements that were in mind when those two techniques have been designed. The classical communities are processed like other BGP attributes in the routing policies and it is possible to build filters that match the communities attribute in complex ways. The provision of such a generic filtering framework requires more processing time than the specific scheme of redistribution communities. Moreover, with `zebra` or commercial BGP daemons [3], a large number of filters need to be defined to provide, with classical communities, the same functionalities as redistribution communities. With `zebra` one filter is required per peer and per redistribution action. This explains the need for more and more time when the number of peers or/and the number of communities increase.

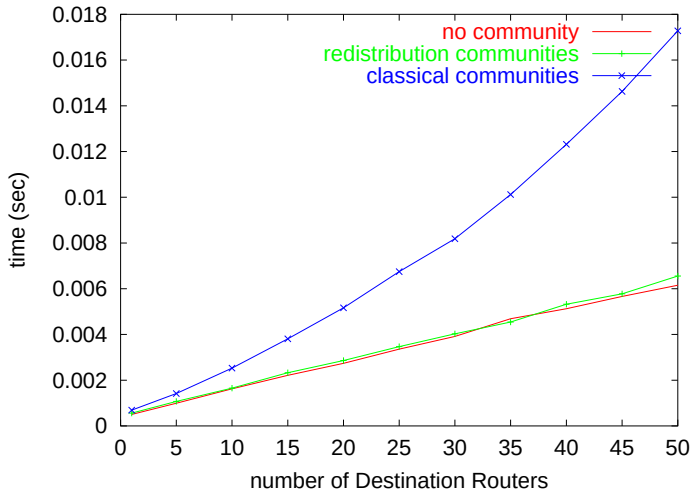

Fig. 9. Average time to process and announce routes to a m peers

6 Perspectives

Many stub ISPs need to engineer the flow of their interdomain traffic. While content providers can rely on the `local-pref` attribute to engineer their outgoing traffic, access providers do not have techniques to control their incoming traffic. In this paper, we have proposed the utilization of redistribution communities to allow ISPs to control the flow of their incoming traffic by controlling the distribution of their route advertisements by their peers or upstream providers. Redistribution communities have two major advantages compared to the utilization of classical community values. First, the redistribution communities have a standardized semantics. This simplifies significantly the configuration of BGP routers and thus reduces the risk of errors. Second, the redistribution communities are non-transitive. This suppresses the risk of polluting all BGP routing tables.

We have then modified the `zebra` BGP routing daemon to support our proposed redistribution communities and evaluated the performance of this daemon. Our measurements indicate that the redistribution communities introduce a much smaller processing overhead that the classical communities.

We expect that the redistribution communities will be quickly deployed by ISPs to replace their complex BGP configurations with classical communities. It can be expected that initially the redistribution communities will be set manually by the network operators like the classical communities today. However, given the standardized semantics of the redistribution communities, it would be possible to define heuristics to automatically set redistribution communities to achieve a given distribution of the incoming traffic based for example on the BGP routing tables received from the upstream peers and traffic statistics.

Acknowledgments

This work was partially supported by the European Commission within the IST ATRIUM project. We would like to thank Russ White, Stefaan De Cnodder and Jeffrey Haas for their help in the development of the redistribution communities. The traffic traces were provided by Benoit Piret and Marc Roger. We thank, RIPE for their whois database and their RIS project and Route Views for their routing tables.

References

[1] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao. Overview and principles of internet traffic engineering. RFC3272, May 2002.

[2] L. Subramanian, S. Agarwal, J. Rexford, and R. Katz. Characterizing the internet hierarchy from multiple vantage points. In *INFOCOM 2002*, June 2002.

[3] B. Halabi. *Internet Routing Architectures*. Cisco Press, 1997.

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

[5] G. Huston. AS1221 BGP table statistics. available from `http://www.telstra.net/ops/bgp/`, 2003.

[6] N. Feamster, J. Borkenhagen, and J. Rexford. Controlling the impact of BGP policy changes on IP traffic. Technical Report 011106-02, AT&T Research, November 2001.

[7] S. Uhlig and O. Bonaventure. Implications of interdomain traffic characteristics on traffic engineering. *European Transactions on Telecommunications*, January 2002.

[8] J. Stewart. *BGP4 : interdomain routing in the Internet*. Addison Wesley, 1999.

[9] Y. Rekhter, T. Li, and S. Hares. A border gateway protocol 4 (BGP-4). Internet draft, draft-ietf-idr-bgp4-20.txt, work in progress, April 2003.

[10] A. Broido, E. Nemeth, and K. Claffy. Internet expansion, refinement and churn. *European Transactions on Telecommunications*, January 2002.

[11] B. Quoitin and O. Bonaventure. A survey of the utilization of the BGP community attribute. Internet draft, draft-quoitin-bgp-comm-survey-00.txt, work in progress, March 2002.

[12] E. Chen and T. Bates. An Application of the BGP Community Attribute in Multi-home Routing. Internet Engineering Task Force, RFC1998, August 1996.

[13] C. Alaettinoglu, C. Villamizar, E. Gerich, D. Kessens, D. Meyer, T. Bates, D. Karrenberg, and M. Terpstra. Routing Policy Specification Language (RPSL). Internet Engineering Task Force, RFC2622, June 1999.

[14] R. Mahajan, D. Wetherall, and T. Anderson. Understanding BGP misconfigurations. In *ACM SIGCOMM 2002*, August 2002.

[15] Routing Information Service project. Réseaux IP Européens, http://www.ripe.net/ripencc/pub-services/np/ris-index.html, January 2002.

[16] D. Meyer. Route Views Archive project. University of Oregon, http://archive.routeviews.org, January 2002.

[17] S. Sangli, D. Tappan, and Y. Rekhter. BGP extended communities attribute. Internet draft, draft-ietf-idr-bgp-ext-communities-05.txt, work in progress, May 2002.

[18] O. Bonaventure, S. De Cnodder, J. Haas, B. Quoitin, and R. White. Controlling the redistribution of BGP routes. Internet draft, draft-ietf-ptomaine-bgp-redistribution-02.txt, work in progress, February 2003.

[19] ISO/IEC. Information processing systems - telecommunications and information exchange between systems - protocol for exchange of inter-domain routeing information among intermediate systems to support forwarding of iso 8473 pdus. ISO/IEC 10747:1993, 1993.

[20] K. Ishiguro. Gnu zebra – routing software. available from `http://www.zebra.org`, 2001.