Application of anycast for the implementation of enhanced Tunnel Brokers IPv6

Jordi Palet Martínez, Miguel Angel Díaz Fernández

CONSULINTEL - San José Artesano 1, 28108 Alcobendas, MADRID, SPAIN

Tlf: +34 91 151 81 99 FAX: +34 91 151 81 98
{jodi.palet, miguelangel.diaz}@consulintel.es

Abstract. *Tunnel Brokers are one of the commonest transition mechanisms used for letting user gain IPv6 connectivity. However this is not a definitive solution since they can be improved in order to be more friendly, more easy to locate them and to let the end user have the best performance on the link. Using IPv4 anycast addresses can be considered as a good alternative for addressing such challenges. This paper pretends to make a description of the barriers that can be found when we implement an anycast TB implementation. The paper also proposes both some solutions for overcoming those barriers and some topics for continuing the research where there is not a clear solution for them. At the end, some anycast TB architectures are proposed that illustrate the concepts that have been shown in this paper.*

1 Introduction

Currently, the deployment of the IPv6 protocol is not yet on the initial stages and it counts already with some native IPv6 networks, mainly in Asia. Because they are not yet globally deployed, if users need to get IPv6 connectivity they still need to use some transition mechanism.

In order to communicate two IPv6 hosts through the current IPv4 networks, the commonest transition mechanism is encapsulate IPv6 datagrams over IPv4, although this requires that users have at least minimum technical knowledge to be able to setup the tunnel on its host. Furthermore, the management of tunnels can get very hard in case of organizations which there exist a lot of tunnels.

For solving this situations the Tunnel Broker (TB) concept [11] has been defined, which is shown in figure *1*, because it helps to both no expert users and networks administrators to easily setup the tunnels. The main idea that is behind the TB concept is to provide one server that manages the tunnel requests from the users.

The TB registers and actives new tunnels, it registers new users, it shows statistic information related to each tunnel that each user has, and so on. TB has associated one or more Tunnel Server (TS), which are the end point of the tunnel, and they are assigned to the user requesting a new tunnel by the TB. The configuration of the TS is made by the TB which also sends to the user a configuration script in order to up the tunnel on the user side. Finally the TB makes also the DNS configuration in order to bind the new user IPv6 address to the DNS name provided by the user.

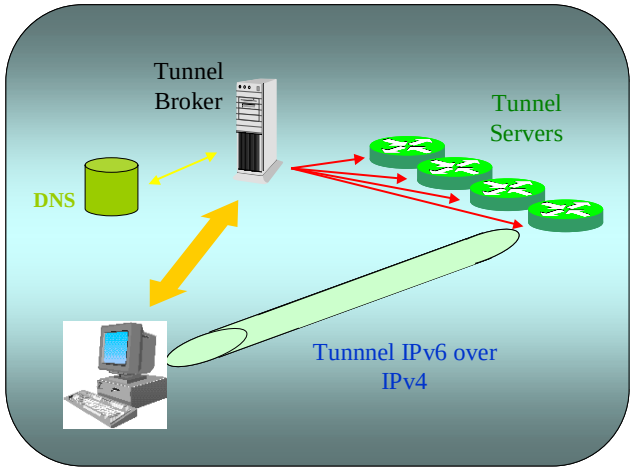

Figure 1: Concept of TB

One important advantage of transition mechanisms based on IPv6 over IPv4 tunnel compared to other ones is that they allow the tunnel work when users have a private IPv4 address behind of NAT boxes, even although these mechanisms have not specifically designed for that. The reason that gets these mechanisms work is that some NAT boxes implement the process described in [12] which realizes the private IPv4 translation to public one when common protocols like TCP, UDP or ICMP are used but also when the labelled by the IETF as 41, that is the IPv6, is used. In this way, tunnels which ends in hosts that have private IPv4 address can cross this kind of NAT boxes.

Because of the TB features, the usage of them is very frequent and they have became one of the most used transition mechanisms not only due to its friendly use but also because it begins now to appear a lot of them, like http://ipv6tb.he.net/, http://tunnelbroker.ipv6.net.au, or http://tb.consulintel.euro6ix.org

However, the TB are not the definitive solution while the transition stage lasts because they admit some improvements in order to make them more friendly, more easily to use and for improving the IPv6 connectivity that the end user can obtain. The usage of the anycast architectures to implement TB/TS is one possible alternative that has to be taken into account to address those improvements. This work tries to show a description of the obstacles that we can find when we implement this alternative. The work exposes some solutions for overcoming such obstacles when it is possible and it points research lines for those barriers that have not a clear solution. Finally, some TB architectures based on anycast addresses are shown in order to illustrate the concepts and ideas that are described in this work.

2 Deficiencies of the TB

As was before stated, the concept used by the TBs makes that these can be considered as one of the most useful transition mechanisms due to the easy of use when users try to setup the IPv6 tunnel by using only one based-web interface, independently of the operating system that is being used.

However, the TBs cannot be considered as the ideal transition mechanism because they show some deficiencies that can be studied in order to achieve a more efficient as well as a more transparent working for the users.

At least three scenes can be described where we can see some deficiencies when users try to get IPv6 connectivity or when TBs have to manage the connections. These scenes are described bellow.

2.1 Scene 1

During an initial IPv6 deployment stage, the ISPs will not provide native IPv6 connectivity but there will be a lot of entities that are going to offer this kind of connectivity by means of TB. Most of these entities belong to either the academic or research scope, so they offer this service for free to the end users.

In this situation, users do not need to be bound to a particular TB in order to get IPv6 connectivity, but they can change of TB at anytime.

Let us imagine one situation where people is crowned, i.e. either national or international submits, airports, urban areas with high population density, etc. In this scene is very likely that most of users choose a particular TB, usually because it is well known. It is possible that while there exists a few TB attending many connections, there can exists a lot of them that are not being used. In this way, most of the users have poor performance in their connections while users using TB without congestion will have good performance.

Given the fact that the users are not commercially bound to any TB, it would be desirable that there were some kind of load balance in order to uniformly distribute the IPv6 tunnel requests among all available TB. It would also be desirable that the load balance mechanism were as much transparent as possible for the users for avoiding extra difficulties while they try to get IPv6 connectivity.

2.2 Scene 2

Transient users needing connectivity to Internet at anytime at anywhere is today a reality: meetings, submits, holidays, etc cause that users change theirs locations very frequently.

In these circumstances to get native IPv6 connectivity is not easy, so users used to visit the web pages of the TB that they usually know, even if they are far of the its current location. But this implies enormous inefficiencies when users move among different countries or continents.

This is the first trouble: why does the user have to get connected to a TB that is likely thousand of kilometres separated if a possibly nearer TB can be used? Long distances usually mean that datagrams have to cross a lot of routers until they arrive to the tunnel end. This also mean high delays that can be increased if there is congestion on the networks.

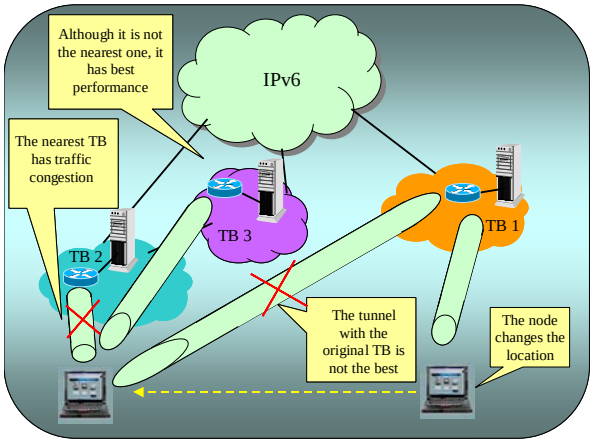

Figure 2: Connection to the nearest TB

Thus, it would be desirable that there exits a mechanism that cause that users can get IPv6 connectivity from the nearest TB. Even if the nearest one is overload, it would be nice to get IPv6 connectivity from other one with less congestion. This concept is shown in figure *2*.

But how can the user know which is the nearest TB? This is the second inconvenience because users will not have a list of available TB, sorted by geographic areas, so getting connection with the nearest TB when it is not known is a challenge that needs to be solved.

2.3 Scene 3

Let us think now in a scene where the IPv6 deployment is more advanced. It is possible that entities providing IPv6 connectivity need to start broad deployment, even by using different TS distributed in different locations, all of them managed by the same TB.

In this situation the management of the tunnels assigned to each user gets more complex because now the TB must decide what TS has to be assigned to each user. Taking into account both the load balance and proximity criteria, this management is not easy.

When the user is located in an area where the TB has a TS it should be used because it is not only the nearest one but also it belongs to the TB which the user is already registered, so this makes more easy the tunnel management. The whole process for having a new IPv6 tunnel with the new TS should be as much transparent as possible in order to avoid that users need to manually contact to the TB or they change the configuration in their host. It would be desirable that the TB/TS architecture make the users get connected to the nearest TS without manual intervention, as is shown in the figure *3*. This is also a challenge to be solved.

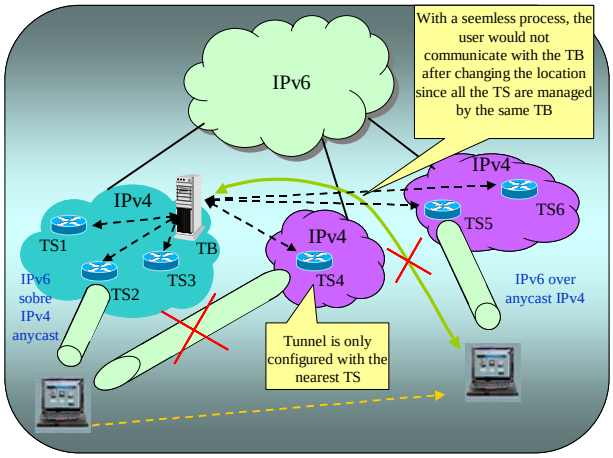

Figure 3: TB with several TS

3 Solutions to obtain the improvements

Thus, it is clear that the weakness of TB can be summarized in the following points: difficulty to determine the nearest TB/TS to get IPv6 connectivity, load balance to have uniform distribution of connections and transparency with the tunnel management when users are moving. There are several solutions that can be arisen in order to asses the improvement of the TB functionality. At least, the following can be proposed.

3.1 Solutions based on centralized servers

With this solution a server would exist which would coordinate a TB group that would accept tunnel requests from users. The centralized server would have real time knowledge of the status of all the TB associated to it, in order to realized a load balance for the requests that have to be attended. Alternatively, the centralized server could inform to the users what is the nearest TB depending on the user location.

However it does not seem that this solution can be globally implemented because there is at least one significant drawback: it is required that all the entities that provide TB should be associated in order to allow a common management of all the resources involved, which seems to be unlikely.

3.2 Solutions based on distributed servers: DNS

Using the DNS system is very tempter because it is a system globally deployed, it has not centralized control and it would avoid that users would run special applications in order to setup the tunnel [7].

With this approach, the user only has to connect with a standardized name and the DNS would redirect the connection to the nearest TB to be used. This kind of system could use the more appropriate metrics related to the commonest routing protocols (BGP, IGP, etc.) [1] in order to choose the best TB, not only in accordance to proximity criterion but also to delay and available bandwidth ones. Even, the load balance could be made by creating a list of candidate TB and after each request is attended the system could modify the list order either randomly, by applying like “Round-Robin” techniques or “hashing” ones, according to the user IPv4 address [2].

Currently there already exist this kind of systems [1] which are designed for other services like distributed web servers. Although these systems present interesting features like good scalability, efficiency, etc. they have some drawbacks. First of all, any system based on DNS making “caching” will never offer real time information, which can be a serious inconvenience when a TB gets down. If caching feature is disabled, then the traffic regarding DNS requests will increase, which is not insignificant [6]. On the other hand, these systems are based on special routers running protocols that belong to the company that developed them, so it does not favour the global deployment of this system.

3.3 Solutions based on the use of anycast addresses

An anycast address identify a group of hosts, usually server hosts. When a client host send a datagram to an anycast address, it is delivered to one of the anycast servers. According to this definition, the anycast concept seems nicely fit as solution to the

problems that TB present as was mentioned in previous sections. With this approach the user requiring the creation of a new tunnel would connect to either a well known address or a well known domain name, i.e. http://www.tunnel-broker.net and the connection would be redirected to the more appropriate TB.

However, according to [3] because of the features of the IP networks, when an IP datagram is sent to an anycast address, this can be delivered to one or more hosts belonging to the anycast group. Even, there is no guarantee that two consecutive datagrams sent from the same host towards the same anycast address are going to be delivered to the same server, so this solution does not seem either the best approach to improve the TB functionality.

4 Obstacles to be overcome to implement an anycast TB

In spite of the use of anycast addresses cause some difficulties to the TB implementation, if we were able to design a based-anycast system, it would obtain several advantages like: (1) simplicity to client's configuration because they only would need to know one destination to get the best IPv6 connectivity. (2) Transparency in the communication in case that a server gets down because datagrams would be automatically redirected to the nearest alternative host. (3) Load balance in order to have an uniform resource share. (4) Facility for the scalability of new TB. Therefore, given the advantages that this solution offers it is worth to have it into account and to make a study of the obstacles that would be overcome so that the TB could be benefited from their use. In the following sections problematic happiness are analyzed and some solutions pointed.

4.1 Anycast in "best-effort" networks

As it has already been mentioned, given the characteristics of the networks IP, if they do not modify the shipment of a datagram to an anycast address, they can cause two different situations in which we have to analyze the implications of using TB anycast, as well as of using a TB with an architecture based on TS anycast. These situations are: (1) delivery of datagrams to more than one anycast host, (2) delivery of datagrams belonging to the same connection to different anycast nodes.

Delivery of datagrams to more than one anycast host

The consequences of this situations depend on the behaviour of the protocols that are above the network layer. These can be classified in two categories: stateless and statefull. As example of first group we can mention TCP, whereas as example of the second one we have UDP.

The distinction is important by the following thing: in the case of using a protocol like TCP when the connection between a client node and a server node has already established, the fact that a datagram arrives at two or the more anycast nodes does not cause any problem, since connection TCP will only be opened in one of them, and it will be such node the one that takes care of the datagram, whereas the rest will discards it of a quiet way. In the case of using a protocol like UDP, the datagram will be processed by all the server nodes where it arrives since there is no information of the state of the connection. This can cause a problem if the processing of the datagram by the corresponding application originates in the server some type of answer that can be based on previous datagrams.

In the case of connections with a TB, although standards do not exist, the most usual is to use TCP connections so that once the connection is established with a server node of the anycast group, it does not seem that it is problematic the fact that later datagrams can arrive at other different serves, except by the fact to waste bandwidth in the connections, since these datagrams will be discarded by the servers which do not maintain the connection with the client node and the communication between client and TB server is not affected. This situation does not cause any newness respect to any other anycast TCP service that already is working at the present time.

In the case in which the architecture of a TB is designed with an anycast group of TS, the client establishes an IPv6 tunnel over IPv4 to connect itself with one of the anycast TS. The protocol that is immediately over IPv4 is IPv6, which due to fact that it also is a network protocol, it does not neither maintain information of the state of the connection and its operation is similar to the case of making a connection between client and server using UDP protocol.

In this case, a datagram using the IPv6 tunnel can reach more than one anycast TS at the same time, nevertheless due to the preventive measures of the TS ("address spoofing"), these only extract IPv6 datagrams of tunnels that were configured previously. In this way with a good tunnel management, only one of the anycast TS will be the tunnel end for each user and all the datagrams that arrive to wrong TS will be discarded in a quiet way [10]. On the other hand, if an erroneous configuration exist and more than one anycast TS is configured like tunnel end, when arriving to them the same datagrams, then these would be processed by all the wrongly implied TS and they would be directed towards the IPv6 node destination through different paths. Therefore the same IPv6 datagram can arrive duplicated to the IPv6 destination host, which does not cause any distinction with respect to the case of an unicast communication

between two IPv6 hosts, given the characteristics of IP networks. They will be the protocols of the higher layers, those that know what to do with this situation.

Therefore except by the multiplication of generated traffic, it does not seem that the fact that the datagrams arrive to more than one anycast server become a problem, not only if we consider the connection with anycast TB but also if we consider the establishment of tunnels with anycast TS. Despite it would be very recommendable that only one anycast host (TB or TS) were used in the communication.

Delivery of datagrams belonging to the same connection to different anycast hosts

In this assumption, unlike the previous one, datagrams belonging to the same connection are delivered to different anycast nodes, even without having duplicity in the delivery of the datagrams, that is, if NA1, NA2, NA3 are nodes belonging to the same anycast group and NC is a node that is establishing communication with the group anycast by sending the datagrams D1, D2, D3, D4, D5, etc., it can arise situations of this type: D1 it is delivered to NA1, D2 is given to NA3, D3 is given to NA2, D4 is given to NA3, etc.

In the case of being used a communication with a TB anycast, since TCP like connection protocol would be used, this would cause a problem that would avoid the attainment of the connection between the client host and the anycast TB, since the datagrams sent to the server hosts that did not establish the connection with the client host would be discarded directly.

In case the client makes a tunnel with the properly configured anycast TS and without considering other considerations like authentication, security, etc., an important problem appears in case the IPv4 datagrams were fragmented, since each TS would have a different fragment and it would not be possible to join them. The only solutions to avoid this situation are (1) to setup the IPv6 MTU to the minimum value that is possible, which is 1280 and to enable the “Don’t Fragment” bit of the IPv4 header or (2) to force that all the datagrams flowing through the tunnel arrive to the same TS, which can be considered as more convenient solution.

Therefore it seems recommendable, even mandatory, that in a scene where either the TB or TS are based on anycast addresses, all the communication between the client and the server hosts (TB or TS) are always made through the same anycast server.

Some solutions to overcome this trouble are:

(1) to modify the behaviour of the TCP/IP stack [3] so that when a client node sends a message TCP SYN to contact with a anycast server hosts, this replies with a message TCP SYN-ACK but placing in it the unicast address that the anycast hosts has rather than the anycast address. As well the client instead of rejecting the datagram coming from an unicast address different to the anycast one which it sent the connection request, it will accept it and will send the TCP ACK message to the anycast server hosts to finish the connection establishment process. Nevertheless this solution does not adapt to the anycast TB implementation since it requires the modification of the TCP/IP stack in both the client and server sides besides to open the doors to possible attacks to the security of the communication.

(2) Another solution proposed in [2] is more interesting since it only would require to make modifications in the network, that is, in the anycast TB/TS hosts. It works as follow: when the anycast server hosts receives a TCP SYN message for starting the connection, it responds with a TCP SYN-ACK message including in the IPv4 header the "Loose Source Route" option with only its unicast address. When the datagram arrives at the client, it includes the same option in the rest of datagrams that it will send. In this way, the “Loose Source Route” option will force that all the datagrams arrive at the same anycast server host. This seems a quite reasonable solution with the only disadvantage that the communication can have a slightly lower throughput because all the routers that are crossed by the datagrams must examine and interpret the option field of these datagrams.

(3) Also conversion techniques from anycast addresses to unicast ones have been proposed, which are carried out in a separated way to TCP[2]. The HTTP protocol would perfectly fit in this solution: the client tries a TCP connection with an anycast server host and when this already has been established, the server makes an HTTP redirection towards the most appropriate TB.

(4) Another solution [8] consists on controlling in the routers the anycast flows that they cross them, by classifying them according to the source and destination addresses, used protocols, etc. By identifying the flows, the routers could make arrive the datagrams at the appropriate anycast hosts, although this would cause an extraordinary load of processing in the routers.

(5) Finally, we can consider the option to modify the IP protocol to include a new option [9], denominated "Source Identification Option" with the purpose of identifying the unicast source of an anycast sender host, but again this option is not recommendable by the fact to have to modify the IP stack in all the nodes that take part in the communication.

4.2 Routing and load balance with anycast addresses

The routing of the anycast addresses is fundamental to obtain that the datagrams arrive at the more appropriated anycast host. For that, the simplest way would be to treat these datagrams as if they were unicast ones and to be based on the metrics that the routing protocols usually used (smaller number of jumps, route with less cost, etc.) to select the proper path. This method usually is known with the name of network layer anycast [4] in contrast with the well-known as application layer anycast which is based on metrics like the available capacity, number of active connections, delay time, etc. that they are calculated by means of applications specially developed to be run in the servers [2,5]. For a service like anycast TB it seems recommendable to use combining of both methods. Nevertheless although the connection of the clients was made always with the more suitable TB, it would exist no guarantee that the negative effects described in 4.1 are not going to happen due to network topology changes or routing failures. In this way, a good routing method should be combined whit some of the methods mentioned in 4.1, which guarantees the exclusive communication with the anycast TB more appropriated.

4.3 Announce of the anycast addresses

Another important aspect within the routing is the consideration or not of anycast address space within the current IPv4 space. It would be possible to define a space of anycast addresses completely differentiated from the rest of unicast addresses, as is suggested in [3]. With this approach it would easily facilitate the identification by the client nodes that the communication is going to be initiated with an anycast address. This type of address could also be exported by the routing protocols as network routes, instead of nodes. Nevertheless given the fact that the node clients already know that connections with TB are a service that requires anycast communication and given the current restrictions in the IPv4 address space, it does not seem that it is advisable that anycast addresses have its own separated space. Therefore, for the case of anycast TB it would be recommendable that the treatment of the anycast addresses by the routing protocols was the same to the unicast case, although this supposes a greater load of processing and increase of the routing tables. In this way, the anycast server hosts only have to announce to he nearest routers that they can be considered as destination for the anycast addresses that they announce. Then, the routers will export the route to the network. Other alternatives to this method exist[3], but from the point of view of the anycast TB the differences are not significant. This approach also means that we have to overcome two new obstacles due to the measures that many ISP use in a preventive way: (1) with the purpose of avoiding an extraordinary increase in the routing tables some ISP filter routes whose destination is not networks but nodes. (2) In addition most of the ISP filter all the traffic whose source address, in this case the anycast address, does not match to the network prefix that the network has assigned.

4.4 Management of TB/TS with mobile users

Since the access to a based anycast TB is always made by using only one address, which can have the corresponding DNS entry, this approach could admit two different treatments related to both the registry and authentication process of the user each time the user wishes to create a new tunnel when he is away from its initial home connection.

(1) Simple management procedure. It is unlikely that all the organizations that give the TB service want to associate each other to facilitate the management of users. For this reason when the user willing to create a new tunnel is redirected to a new TB in which he has not been registered previously, the user would have to make again the whole registry process to give the necessary data that they made possible the creation of a new IPv6 tunnel with the new TB. This situation neither causes no newness compared to the use of a TB unicast.

(2) Advanced management procedure. This solution tries to facilitate the management of the user. In fact with this procedure all the process related to the change of TB would be transparent for the user, but before it would be necessary to asses at least the solution of the following aspects:

a) To maintain, independently of the TB used for the access, of the user historical data regarding the statistics, tunnels that are up, tunnels used in the past, traffic, etc. This data are important to make an appropriate invoicing in case that the service is invoiceable.

b) For that, the user should be authenticated with any AAA procedure by using the original TB where the user is already client. In order to achieve this, the user should give the proper data that identify its original TB in order to make the authentication process in it.

c) With a new IPv6 address is assigned, the host will loose the possibility that other IPv6 node contact to it using the old IPv6 address provided by the previous TB before the TB change. In order to avoid this situation it is possible the use of mechanisms of IPv6 mobility (MIPv6) that inform the new IPv6 address to the Home Agent, - which would have to be implemented and located in the original TB. Thus it would be avoided to have to make modification in the DNS.

When the TB consists of several anycast TS, if the architecture works according to a no transparent movement management, users would need to contact with the TB to create a new tunnel and the TB would make all the necessary configurations in the new assigned TS that the user is going to use. The IPv6 address that the TB would provide to the user could be same or the different one depending on the networks in which the TS were located. However, if the architecture implements a transparent movement management, if we want the user does not need to contact to the TB, then the following points must be solved:

a) the establishment of an IPv6 over IPv4 tunnel requires the appropriate configuration at both tunnel ends. In the client side, the configuration will not change, even the user change its location. However, the TS can be different, so it would be necessary to somehow notify the new TS that must authorize the broadcasting of the datagrams that begins to receive from a new user who has changed its location. It would be necessary to develop some type of automatic authentication (AAA) by means of script/application before allowing the use of the new TS.

b) Once the new user has been authenticated, the tunnel end has to be configured at the TS side.

c) To maintain the same IPv6 address. In order to make the change of location process absolutely transparent for the user, he would have to maintain the same IPv6 address, which would force to all the TS serve IPv6 addresses with the same prefix network since otherwise the IPv6 datagrams always would be discarded in the TS when verifying that a datagram with a different prefix network arrives to him [10].

4.5 Modification of network conditions that recommend another TB/TS

The change to a more suitable TB cannot only arise because the user changes of geographic location but in addition because some of the anycast server nodes can get down or the conditions and/or topology of the network change. In this case the user is not conscious of it. In order to the user always gets the connection with the best performance, it would be necessary to solve the following disadvantages:

(1) the network would have to inform to the user that it is recommendable to change of TB and to make a new tunnel to always have the IPv6 connection in the best conditions. The notification to the user could be made by e-mail or each time the user is authenticated in the TB. When the user accesses to the anycast TB to make a new tunnel, the user management problematic was already pointed in 4.4 section.

(2) the fact to change of TB makes nonviable that the process of the creation and establishment of a new tunnel is transparent for the user. However in case a TB is formed by several anycast TS, the transparency of the management when the conditions and topology of the network vary can be possible since the user would have at anytime an IPv6 tunnel towards a single IPv4 address (anycast). If all the TS belong to the same TB, it can facilitate the management, however it is necessary to solve the same troubles that were mentioned in 4.4, specially with respect to the mechanisms of automatic authentication. Even in this case there is one more difficulty because now on the user side any script/application could not be ran when the tunnel is up.

As summary of the potentiality of each architecture, the tables *1-A* and *1-B* are shown.

		TB anycast
Change of location	Manual configuration: it is not transparent for the user	- Simple management. User is registered as new user and a new tunnel is created in the new TB - Advanced management: User data are mantained and a new tunnel is created. Used of MIPv6 concepts
	Automatic configuration: it is transparent for the user	It is not feasible if connection with the nearest TS is desirable since the new tunnel establishment is made through the TB, so this avoids that the management is transparent.
Change of network conditions	Manual configuration: it is not transparent for the user	- Manual management needs to be previous to the user notification. - It is feasible both types of managements: simple and advanced.
	Automatic configuration: it is transparent for the user	It is not feasible if connection with the nearest TS is desirable since the new tunnel establishment is made through the TB, so this avoids that the management is transparent.

Table 1-A: Possibilities offered by an anycast TB architecture

5 Anycast TB architectures

Considering the diverse concepts that have been shown in the previous sections, some architectures can be proposed that implement TB/TS based on anycast addresses and they allow somehow the load balance.

		Anycast TS
Change of location	Manual configuration: it is not transparent for the user	- User only has to run a script to get the tunnel up - All the TS have the same network prefix: if all at he TS have all the data regarding all the tunnels then it is not necessary to contact to the TB - All the TS have different network prefix: it is necessary to contact to the TB to create the new tunnel.
	Automatic configuration: it is transparent for the user	- User only has to run a script to get the tunnel up - All the TS have the same network prefix: only the user authentication is needed to use the new TS - All the TS have different network prefix: it is not possible because the IPv6 address would be different
Change of network conditions	Manual configuration: it is not transparent for the user	It is not feasible because if the management is manual, after the notification to the user is made, the new configuration would be made using the TB by the user
	Automatic configuration: it is transparent for the user	- All the TS have the same network prefix: It is only feasible if all the TS have available the data regarding all the tunnels created at the TB. - All the TS have different network prefix: it is not possible because the IPv6 address would be different

Table 1-B: Possibilities offered by an anycast TS architecture

5.1 TB with a single TS

Anycast based on the network layer

This architecture, that is shown in the figure *4*, consists of an anycast TB pertaining to the same group. They announce the anycast address by means of the usual routing protocols. The load balance would be obtained equipping with "intelligence" the routers so that they change the priority of each route to the anycast destination by following any solution enunciated in 4.2. On the other hand, the connection with a single TB of the anycast group would be obtained with the addition of the option "Loose Source Option" in the IPv4 header, as it is explained in 4.1.

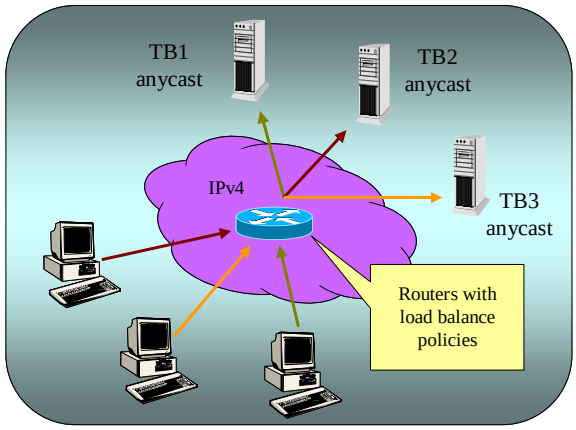

Figure 4: Network layer anycast TB

Anycast based on the application layer

In figure 5 this solution is shown where the concept of the Intermediary of TB (ITB) appears. Each ITB belongs to the same anycast group and they have associated an unicast TB group. ITB has real time data about each associated TB like number of active tunnels, traffic amount, delays and so on in order to make the balance of the connections. Users willing to create a new tunnel connect to the anycast ITB which makes the redirection of the HTTP connections to the appropriate associated TB according to the real time data. In this case, the anycast address of the ITB can correspond to a standardized dominion name, like http://www.tunnel-broker.net or similar in order to facilitate the user access.

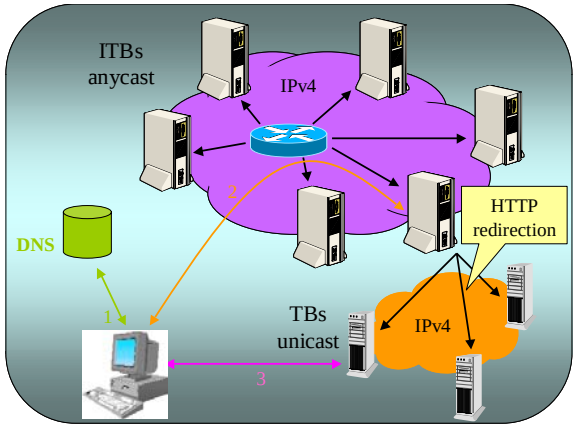

Figure 5: Application layer anycast TB

5.2 TB with an anycast TS group

If the TB consists of an anycast TS group, a possible architecture, which has been implemented in [13] is shown in figure *6*.

The key in this implementation is the selection of the TS. The procedure is based on the ICMPv4 ping requests as follows: when the user wants to create an IPv4 tunnel he contacts to the TB which informs to all the TS and then each anycast TS sends to the user a sequence of pings with a code that it identifies each TS. The source address of the ping datagrams is the anycast group. The user host sends ICMPv4 ping replies towards the anycast address which arrive to

different TS. However, the nearest one receive the most ping replies. By means of a complicated process the TB is informed about which is the TS that has received more ping replies and the tunnel configuration is made in it.

Although this architecture solves the problem to find the nearest TS to the user, it presents some disadvantages:

- The TS assignment is not made according to load criteria.
- It does not allow a dynamic load balance
- It does not make possible a transparent change process of TS for the user when the conditions of network are different
- It is not tolerant to failures of the TS

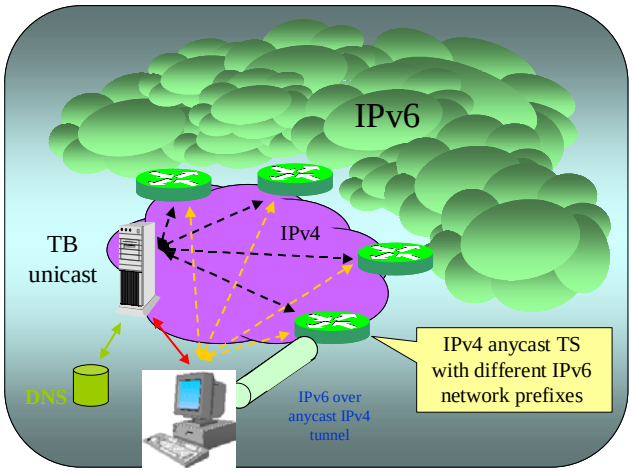

Figure 6: Network layer anycast TS architecture

6 Improvements and future lines of investigation

It is evident that the use of IPv4 anycast addresses is a matter little explored, and its application for the TB/TS implementation even less, reason why it is necessary still more research in this field to improve both the functionality of the architectures and the operation of the nodes that are implied in the communication with TB/TS anycast, as well as to overcome the existing barriers that avoid the transparent management when users change their location.

Concretamente, podría ser conveniente estudiar la viabilidad de la incorporación de los siguiente puntos dentro de la funcionalidad de TB/TS anycast:

Particularly, it could be advisable to study the addition of the following points within the anycast TB/TS functionality:

- Use of Neighbor Unreachability Detection packets on the "link local" tunnel interface
- Use of Neighbor Discovery packets on the "link local" tunnel interface
- Use of automatic tunnels with ::IPv4 addresses
- To implement anycast TS in LAN segments with load balance
- User authentication when a TB different than the usual one is used
- Load balance with network layer within:
  - Intra AS
  - Inter AS
- Transparent management of connections within anycast TS when either a TS gets down or there exist changes in the network conditions
- Use of IPv6 anycast addresses in the TS for the configuration of the tunnels in order to facilitate the transparent management of connection while users are moving.

7 Conclusions

Within this paper it has been presented the problems that arise in some scenes when TB are uses as transition mechanism. Also an anycast based architecture can be used to solve those problems. However the use of anycast requires to overcome some barriers for being useful. In previous sections it has been presented some ideas and concepts that can be used to overcome such barriers as well as to improve the TB functionality.

References

[1] Cisco Distributed Director 4700-M. http://www.cisco.com/en/US/products/hw/contnetw/ps813/products_installation_guide_chapter09186a008007d7c0.html

[2] Robert Engel, Vinod Perdis, Debanjan Saha. “Using IP anycast for load distribution and server location” http://www.zurich.ibm.com/~rha/papers/anycast-gi98.pdf

[3] C. Patridge, T. Mendez, W. Milliken. “Host Anycasting Service”. RFC 1546. Noviembre 1993.

[4] Chris Metz. “IP Anycast: Point to (Any) Point Communications” http://www.cisco.com/public/cons/isp/essentials/ip-anycast-cmetz-03.pdf

[5] S. Bhattacharjee, M. Ammar, E. Zegura, V. Shah and Z. Fei. “Application-Layer Anycasting” IEEE Infocom 1997. Kobe Japan.

[6] P. Danzig, D. Delucia, K. Obracza. “An analysis of wide-area name server traffic”. Proceedings of ACM SIGCOMM ’92, Baltimore, MD, Agosto 1992

[7] T. Brisco. “DNS support for lad balancing”. RFC 1794. Abril 1995

[8] D. Dias, W. Kish, R. Muhkerjee, R. Tewari. “A scalable and highly available web server”. Proceedings of the IEEE Computer Conference (COMPCON)”. Santa Clara. Marzo 1996.

[9] M. Shand, M. Thomas. “Multi-homed host support in IPv6” Internet draft. Junio 1997

[10] E. Nordmark, R. E. Gilligan. “Basic Transition Mechanisms for IPv6 Hosts and Routers”. Internet draft. Enero 2004

[11] A. Durand, P. Fasano, I. Guardini, D. Lento. “IPv6 Tunnel Broker”. RFC 3053. Enero 2001

[12] J. Palet, C. Olvera, D. Fernández. “Forwarding Protocol 41 in NAT Boxes”. Internet draft. Octubre 2003

[13] X. Liu, X. Li, “Tunnel Broker System Using IPv4 Anycast”. Proceedings of Network Research Workshop in 16th APAN Meetings, 2003.9-14 http://www.apan.net/2003_busan/21.pdf