DTIA: an Inter-domain Reachability Architecture Technical Report

Pedro Amaral, Luis Bernardo, and Paulo Pinto, *Member, IEEE*

***Abstract* - This paper proposes a reachability architecture and algorithms for the inter-domain environment of the Internet, called DTIA – Dynamic Topological Information Architecture. It is based on the knowledge of a static network formed by the Autonomous Systems (AS) and an algorithm to manage link failures. It goes a step further in the usage of a database with policies available from RIPE which is being used already by ASes in real-time. It answers to most of the current limitations of BGP, specially the growth of the routing table, multihoming, churn rate, range of routing events and scalability. Regions are defined as a mechanism to sustain scale. The system features most of the functionalities of BGP and enables multi-path routing to ASes.**

***Index Terms* - Internet routing; inter-domain routing; scalability; multihoming.**

I. Introduction

THE current protocol for inter-domain routing, BGP (Border Gateway Protocol), is a backbone of the current Internet. Therefore, any replacement or even any changes to it is a very sensitive matter. However, over the years several weaknesses and inefficiencies [1] have been identified that should deserve attention. Some examples covered in this paper are: the growth of routing tables and Forwarding Information Bases (FIB) on core routers; the slow convergence; the churn rate of route updates; and the range covered by routing events. The overall situation will get worse with time as multihoming might start to be used extensively increasing even more the number of prefixes in the network.

BGP is a fairly simple protocol and is very flexible. It uses prefix-based routing and the flexibility in using the attributes allows very precise manipulations prefix by prefix for common routing aspects, and even for others not so related to routing. Examples for the former are the precise behaviors for backup links (depending if they are between a customer and a provider, or between peers at stub level or at provider level, etc.) or, for the latter the construction of prefix-based VPNs. We should keep in mind that any new solution for inter-domain routing cannot feature all the facilities available today and still remain simple. Some features have to be considered secondary and be performed in other ways. The difficulty is the identification and agreement amongst the community on which features should be considered secondary.

This paper assumes that the main business relations remain the same as today and builds on them a simple scenario. Dynamic Topological Information Architecture (DTIA) proposes a "rupture" scenario that assumes knowledge of the static network topology at regional level (e.g. RIPE). Regions create a scope for the scalability of the algorithms that handle faults and routing. DTIA deals with connectivity information in accordance with the *common policies* [2] in the Internet. It addresses most of the limitations of BGP and compared with other proposals in the literature gives special attention to issues such as backup links, multihoming and sibling relationships between Autonomous Systems.

The paper begins with the statement of three assumptions and the rationale behind our choice of principles. Our purpose is to show that some of the BGP features lost their potential use over the years. Then the architecture is described with some preliminary experiments showing its feasibility. Main considerations about the deployment finish the paper.

II. Rationale

The two main starting assumptions to construct the architecture are the maintenance of the current business model based on Autonomous Systems (AS) and Internet Service Providers; and the hierarchical architecture based on customer-provider links forming a three-tier structure [3]. This hierarchical structure sits on the existence of greater bandwidth links at higher levels that transport higher amounts of traffic. Although the tier 2 is getting richly connected with peer links over the years the use of these links to transport third party traffic is the exception and not the rule.

A third assumption is the characteristics of the links that form the Internet. They are pretty stable over the time because they are based on business relationships. Any changes happen in a controlled manner. The time sensitive issue is whether the link failed or not, and not so much if it exists or not.

The fact that the BGP is prefix based has several consequences. Given the fact that each time only the best route is advertised the end result is the construction of several graphs (per prefix) over the physical links. The knowledge of the topology of the network is not a first class concern (although it can be inferred [4]). As the attributes are also based on prefixes the routing behavior (the actual graphs) can be very different in a region making the system very complex and hard to manage. Therefore, it is not easy to use the topological information to accelerate convergence when transient failures happen.

Working at prefix level enlarged the size of the routing tables and it is consensual that this growth must be contained. One way to reduce the growth rate is to rely on prefix aggregation. In architectural terms this will not work because BGP is really based on prefixes and they are the knots to change behaviors. For instance, traffic engineering and load balancing can be based on separating flows (prefixes) that belong to an AS, enlarging the routing tables. Note that performing these tasks using prefixes is quite inefficient because traffic for a prefix can change over the time. It is a rough solution to the problem highly suited to the characteristics of BGP. The use of multihoming makes aggregation even harder: consider an AS getting its prefixes from provider 1 and having other *n* providers. Every provider but provider 1 cannot aggregate the prefixes. Even provider 1 may not want to aggregate – if it does it might get no traffic because more specific longest match paths are preferred.

BGP uses attributes in the UPDATE packet to describe the characteristics of a prefix. Each UPDATE packet received goes through a filtering process and can have its attributes manipulated before its route is placed in the routing table. Routes in the table suffer a similar process (filtering and manipulation) before being sent to neighbors in UPDATE packets. The attribute manipulation provides most of the flexibility of BGP. Over the years attributes have been used to produce specific effects on routing enriching the ways ASes interact. The current reality is a complex system that is highly sensitive to the coordination and simultaneous implementation in all AS in a region [5]. Firstly because the building blocks (attributes) were not designed for certain purposes they are used now creating a cumbersome system (for instance, prepending AS number in the AS Path [6], or using the community attribute to define VPNs [7]); and secondly because some techniques make use of highly expressive semantics providing freedom on establishing rules, producing a large scope of intervention and difficulties in living without them (examples are the usage of regular expression manipulation on the AS Path, or the meaning of the community attribute numbers that are not standardized and can be anything an AS wants [7]).

BGP should be a protocol able to learn prefixes dynamically and act accordingly. If we look closer, the attribute manipulation destroyed this feature and some relevant manipulations assume a complete knowledge of the topology of the network in the region. There are many examples mainly involving AS prepending and multihoming. Some of them are: a) consider an AS with two providers and providers of these providers. In order to make load balancing the stub AS has to know the path until a NAP (Network Access Point) (or a common AS) in order to know how many times it should prepend the AS Path; b) the same arguments for the choice and meaning of numbers for the community attribute when used to achieve AS Path prepending; c) in multihomed scenarios prefix aggregation can completely drive away traffic if we do not take into consideration how prefixes are advertised through the other branches; d) consider the situation of two AS providers having each one a different stub AS client and a backup link between these clients. In order for each provider not to use the backup link to forward traffic to the other provider's client, local preferences must be carefully assigned and the knowledge of the topology is necessary. Configuring the system so tuned to precise topologies can make it unpredictable when links fail.

As we go up in the hierarchy certain aspects are even hidden. For instance, at a certain level in the hierarchy aggregation is performed because it makes no sense to advertise different prefixes to the entire Internet following the same link at that level. This aggregation implicitly sets boundaries for failure event notification in a way hard to control.

A side-effect of BGP being based on prefixes is the construction of a real data base of existing prefixes on routers all over the Internet. This database supports liveness, mobility, etc. subject to the convergence speed of the protocol.

Based on these considerations the following section sets the main principles we used to define our architecture.

III. Main principles

Our main principles are:

Reachability is based on AS connections and not on prefixes.

Given the number of ASes, the reduction of the routing table is significant. This decision is controversial with some opinions against it [8] and others following it [9]. It brings further advantages: traffic engineering and load balancing can be performed amongst ASes providing a more efficient solution based on a single graph compared to the prefix solution; multihoming is reduced to a choice of paths and ASes without any consequences to the size of the routing table. Two problems exist: 1) packets can follow different paths with different transit times making it necessary to adapt the congestion control algorithm of TCP (the calculation of the Round Trip Time becomes more complex and the reaction of TCP to the reception of a number of packets out of order must be reconsidered); and 2) a mapping between prefixes and ASes must exist.

We assume that there is a service to map prefixes to ASes. This service can support host multihoming. It can also support mobility in terms of prefix assignments to ASes to cope with mobility requirements seen in military networks.

Routers get a static map of the network and co-operate to learn about failures

A central entity (or various to provide reliability) delivers a static map of the network (or a region, see principle 3) to routers. There is no guarantee that the static map is the real picture of the network due to failures. Nevertheless, all routers know the same information and can act upon it. The reachability protocol assumes a static reality and a dynamic reality due to failures. This approach was followed with different purposes by [10]. As there is no need to discover the graph, the traditional routing paradigms do not apply (distance-vector, path-vector, and link state) and the dynamic part of the protocol is simplified in terms of messages exchanged. The major problems to solve are to warn routers about failures, re-route data packets that encounter a failure, and warn routers when the failure is solved. The dissemination of failure information should only "disturb" the relevant routers with precise rules about its scope.

Maps and co-operations are limited to regions

Most of the concerns in inter-domain routing are local to the ascending (and descending) paths. Real global events in BGP are related to the withdraw procedure of prefixes, an issue that our model does not have. Depending on their placement in the hierarchy and what aggregations exist, link failures in BGP are confined to regions.

One can enforce the notion of a "region" based on the characteristics of BGP (or more generally on the characteristics of inter domain routing). HLP [9] proposes the concept of a tree based on the customer-provider links and one hop peer-to-peer links to confine their algorithms. Due to the heavy use of multihoming at middle levels this concept can become complex with routers belonging to too many trees. HLP fails also to address backup links and does not take into consideration the real web of peer-to-peer links that exist already.

We propose a more rigid approach: divide the Internet in regions and for each region construct the static graph delivered to routers. Nowadays RIPE has already an embryonic database that can be used for this purpose[1]. This database [11] stores all policies of the European ASes. Its format is not suited yet for our purposes but we used it to construct the European static graph. We also used a topology from the CAIDA AS Relationships Data research project [12], and the method described in [13] to infer relationships.

Based on this approach the Internet is divided into regions and packets going from one region to the other use either a direct link from one internal AS connected to the destination region (if valid), or have to climb up the hierarchy and go down in the destination region. A certain degree of inefficiency might exist which is typical of multi-layer routing systems.

The precise size of a region has consequences and is still under study. In our experiments we used all ASes in the RIPE database, which includes all Europe and part of Asia. Regardless of the size regions always include tier-1 ASes.

IV. Architecture

The region graph is built by an entity (e.g. RIPE for the European region) and distributed to all nodes (ASes) of the region. Each time a new graph is generated an increasing sequence number is assigned to it. The graph G(V,A) is modeled as a directed graph with V(G) vertices that model ASes and A(G) arcs that model links between ASes. The arcs are labeled according to the commercial relationships between the ASes.

Information on commercial relationships is already partially available today in internet registry databases like RIPE, and can be easily inferred using this and other information sources (e.g. routing table dumps available like the ones in the Route Views Project). It is not considered as being secret.

BGP is a policy based path vector routing protocol. It has a high degree of *expressiveness* allowing many different policies that can model many different networks. However several *robustness* problems are known to occur due to the lack of policy coordination in BGP. In practice only a small set of policies are used extensively in the Internet today, the so-called *common policies* [2]. Our approach is to associate the *common policies* both to the labels of the graph and to a small set of rules. The result is a stable and robust base upon which more complex algorithms can be built. As an example, the graph is used to calculate valid paths from one AS to another. It is up to higher-level algorithms to perform routing, traffic engineering, load balancing, etc. over the path set.

We consider four types of inter AS relationships

- Provider-Customer. One AS (the provider) accepts all traffic from the other AS (the client).
- Peer-to-peer. ASes provide connectivity for their direct or indirect customers. No transit traffic from the peer is allowed.
- Peer-to-peer allowing backup. The same as before but allows transit traffic if no other path exists.
- Peer-to-peer allowing transit traffic. Transit traffic is allowed in any situation (this is not very usual but exists in the RIPE database).

These relationships are modeled in the graph using directed arcs between the two ASes:

Provider-Customer – One arc in the provider-customer direction (p2c) and one arc in the customer-provider direction (c2p)

Peer-to-peer – One arc in each direction (*p2p*)

Peer-to-peer allowing backup – One arc in each direction (*p2pbkup)*

Peer-to-peer allowing transit traffic – One arc in each direction (*p2patt*).

The Provider-Customer and peer-to-peer relationships are enough do deal with 99% [3,9] of the *common policies* used today in BGP. The two latter relationships increase our architecture *expressiveness* to include complex relationships such as backup using peer-to-peer relationships, as suggested in RFC 1998, and siblings' relationships.

The graph is used to calculate the set of all valid paths from one AS X to any other in the region, denoted as *P(X).* A path is valid if it complies with the set of rules. This set of rules represents the current economic relationships, the *common policies*, and guarantees that the reachability information is robust.

A table, *FH(X)*, generated from *P(X)*, contains the different first hop exits for each AS in the region. It is up to higher-level algorithms to perform routing, traffic engineering, load balancing, etc. using this *FH(X)* table. Note that it is up to the routing protocol to solve loops due to the existence of multi-paths to a destination. The only guarantee is that each path in *P(X)* is loop free.

A. General Principles

There are two general principles in the current Internet architecture:

1. No traffic is forwarded from one provider or peer to another provider or peer.
2. Customer routes are preferred over peer or provider

[1] It is used already by providers to verify prefixes advertisements from their clients.

routes.

Principle 2 is a preference rule. In terms of reachability and considering backup links, all paths are valid. It is up to the routing protocol to comply to rule 2 (e.g., a *p2pbkup* link can be used if no other paths are available).

The consideration of peer-to-peer links allowing backup or transit traffic might introduce exceptions to principle 1 (in BGP this is solved by AS Path prepending, for instance).An example of such exception is illustrated in Figure 1.

In Figure 1the connection between F and H failed. By principle 1 A is disconnected from G. However, path A-B-C-D-G violates principle 1 and is valid. If the relationship between B and C were *p2p* this path would be invalid.

B. Searching and Pruning process

Each AS explores the paths to all destinations in the graph in a hop-by-hop process. To control valley paths a qualifier, named Direction (D), is added to each path. Direction is set according to the first arc: if the first arc is *c2p* Direction is set to 1; if the first arc is *p2c* it is set to 0. If the first arc is *p2pbkup* or *p2patt*, two paths are considered: one with D=0 and another with D=1. Further processing will invalidate one of them. If it is *p2p* only the D=0 is considered.

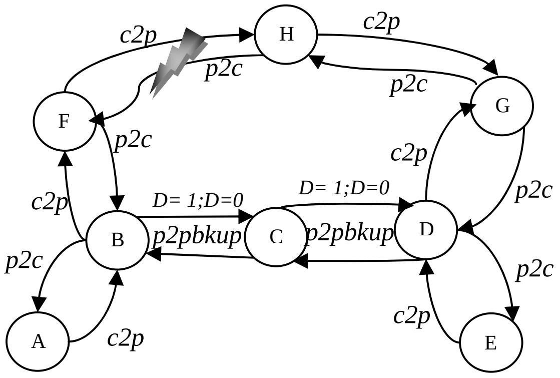

Figure 1 –Example topology

The value of D can change in the course of the path exploration. A descending path (D=0) never changes to an ascending path (no valley paths are allowed). An ascending path is changed to a descending path when the first *p2c* arc occurs in that path.

Result	p2c	c2p	*p2pbkup*	*p2p*	*p2patt*
p2c	V	X	V	X	V
c2p	-	-	-	-	-
p2pbkup	V	X	If (AS in set)X else V	X	If (AS in set)X else V
p2p	X	X	X	X	X
p2patt	V	X	If (AS in set)X else V	X	If (AS in set)X else V

Table 1 –Rules to validate paths for D=0.

Result	p2c	c2p	*p2pbkup*	*p2p*	*p2patt*
p2c	-	-	-	-	-
c2p	V;D=0	V	V	V	V
p2pbkup	V;D=0	V	If (AS in set)X else V	X	If (AS in set)X else V
p2p	V;D=0	X	X	X	X
p2patt	V;D=0	V	If (AS in set)X else V	X	If (AS in set)X else V

Table 2–Rules to validate paths for D=1.

Table 1 and Table 2 contain the validity rules (valid (V) or invalid (X)) for an arriving arc in the row and a departing arc in the column. Table 1 is for paths with D=0 (descending paths). In a descending path there cannot be *c2p* arcs and *p2p* arcs are always non valid. Table 2 is for the D=1 case (ascending paths). In an ascending path when the first *p2c* arc appears the Direction changes its value.

Peer to peer arcs pose extra problems in terms of guaranteeing no loops for the paths. To solve them whenever such an arc is followed the departing AS number is recorded in an AS set for that path. Whenever an AS is reached using such an arc a verification of whether this AS is in the set is performed.

Figure 1 shows the exception case when a path begins with a *p2p* like arc (two arcs in the case). The process is running on B. Two paths are set to C, and again to D. Both C and D are included in the AS sets of both paths. When going to G the path with D=1 is valid and the other is invalidated (cannot follow a *c2p* arc). When going to E the D=0 path is valid and D=1 is invalidated (a *p2pbkup* arc cannot be followed by a *c2p* arc).

C. Results

After the pruning process there is a set of valid paths to every AS in the region (and to other regions). If there is more than one path to an AS the routing algorithm will decide on their usage. An important characteristic is whether these paths contain loops or not. The following theorem proves there are no loops for the paths.

Theorem 1: *Assuming that:*

There are no cycles in the provider customer relationships[2].

A valid path between two AS in the region has no loops.

Proof: We prove Theorem 1 by contradiction. Let's suppose that we have a path P = $\{x_1, x_2, \ldots, x_{n-1}, x_n\}$ $x_k \in V$ in G=(V,A) with $x_n = x_1$. This path is formed if every AS x_k chooses x_{k+1} as the next hop to the destination and x_{n-1} chooses $x_n = x_1$. It also means that for all $1 \le k \le n-1$ there is a loop $L_k = \{x_k, x_{k+1}, \ldots, x_n = x_1, \ldots, x_k\}$. Note that for this path to be possible it must comply with the policy rules.

Let's consider the following set of values for each of the arc types defined: a value (3) for *c2p* arcs, a value (2) for *p2p* arcs

[2] I.e. no domain is a provider of one of its direct or indirect providers assuming that peers are also indirect providers.

of any kind (*p2p*, *p2pbkup*, *p2patt*) and a value (1) to *p2c* arcs. A path can be described as a series of values each corresponding to one of the arcs in the path.

If we exclude *p2pbkup* and *p2patt* arcs, the values of the consecutive arcs of a path are *non-increasing* due to the following conditions:

a) After a *p2c* (1) arc there can only be *p2c* (1) arcs.
b) After a *c2p* (3) arc we can only continue to have *c2p* (3) arcs or *p2p* (2) arcs followed by *p2c* (1) arcs.

In these conditions the only possible loop L_k happens if all arcs were of the same value (*p2c*, *c2p* or *p2p*). Paths with all arcs of *p2c* or *c2p* type would obviously be in violation of condition 1. Paths with all arcs of *p2p* type are invalidated by the AS set mechanism described above, hence no valid loop exists.

If we consider *p2pbkup* and *p2att* arcs we will have only 2 types of *increasing* path segments:

1. After a *c2p* arc, one or multiple *p2pbkup* or *p2patt* arcs can be followed by a *c2p* arc. We refer to these path segments as P_{i1} = {3, 2, 3} paths.
2. After a *p2c* arc, one or multiple *p2pbkup* or *p2patt* arcs can be followed by a *p2c* arc. We refer to these path segments as P_{i2} = {1, 2, 1} paths.

The first case is a step in an ascending (D=1) path, after which there is always a *c2p* arc (and we can consider that multiple steps might occur). The path contains a sub-sequence starting with 3 and ending with 3. If the Theorem assumption holds we can replace all inner 2 by 3 and get to the case of *non-increasing* sequence.

The second case is a step in a descending (D=0) path, after which there is a *p2c* arc (again, multiple steps can be considered). Once more if the assumption holds the values 2 inside a sequence of 1's can be changed to 1 and the *non-increasing* property holds.

One final possible loop is introduced by the *p2bkup* and *p2patt* arcs, a loop that starts with a 1 or 3 value and has all the other arcs of value 2, this loop is invalidated by the AS set mechanism and is in violation of the theorem assumption since an AS would have to be a peer to one of its direct or indirect providers.

By contradiction we proved that no valid loops can occur in paths with only *p2c*, *c2p* and *p2p* arcs. We also proved that *p2pbkup* and *p2patt* arcs do not introduce any valid loops. This concludes our proof.

Regions are connected at tier-1 or with direct links between lower level ASes. In the latter case these links can be either *p2p* or *p2patt*. *p2p* links are a private business of the ASes involved and not used by other ASes. *p2patt* can be used by any other AS in the region only if the path towards the AS in the border is in ascending direction. ASes connected by *p2patt* exchange the set of ASes reachable through them. When a packet arrives for a destination in the other region that is not in the set, it is sent to tier-1 ASes.

To enable the relation of an AS having a client in another region, a "dummy" AS has to be present in the client's region connected to the original one with a *p2patt* link as shown for AS K in Figure 2.

Our architecture allows the use of source-routing as well. Within a region it is easy to find the path a packet should follow (in a strict or loose sense). It is also possible to define the entire path of a packet by getting the maps of the various regions. It is a computational heavy task but it is equivalent to the semantics of the AS Path manipulation by matching to a regular expression to avoid certain ASes, for instance. In our architecture it even becomes stronger because a path avoiding (or using) certain ASes is enforced from the source.

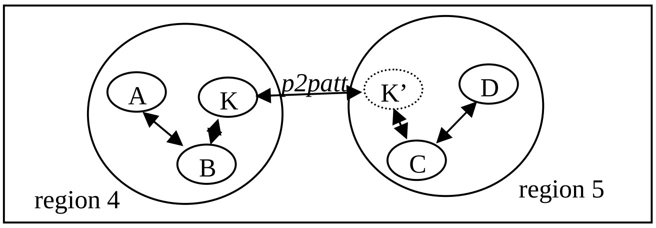

Figure 2–An AS with a client in a different region

V. Failure Recovery

The static graph is no guarantee that the links are up. The dynamic part of the protocol is used to create awareness on link failures and mark valid paths as not available during the failure. There are two goals: assure that only failure free paths are available to the routing protocol and assure that in terms of reachability no packets are lost if at least a failure free path exists.

Only links fail (a failing AS means all its links failed). Assume a link between B and J fails. Routers at the endpoints disseminate a control packet with the link identification, the serial number of the graph, and the Direction. Upon the reception of such a packet an AS checks if it can still reach all reachable ASes. If it can, the dissemination is stopped. If, at least one reachable AS becomes unreachable, the dissemination continues. The dissemination follows the rules of the previous tables. When the link comes up again a similar procedure is used (the dissemination continues if an AS was unreachable because of this link and becomes reachable). The dissemination uses reliable sessions. Note that the routing protocol in each router is warned about the link failure and there might be consequences at routing level.

The scope of the dissemination is directly related to the degree of multihoming in the region. A high degree of multihoming makes the disseminating region smaller. A failing AS always reaches the entire region. However, it is a rare event unless they are stub ASes (that are even more likely to fail). If it is a stub AS connected only to one provider AS no control packets are sent from that provider. This is consistent to the current Internet because even today packets can reach a destination AS just to know that the prefix might not be valid at that moment.

The serial number is used to force the synchronization on graph versions. It might happen that the number in the packet is smaller than the current version on the receiving AS. A synchronization graph packet is then sent backwards to force the synchronization.

During the control packet dissemination some of the nodes are already using alternative paths, while others are still using the failed path. We show that our mechanism does not lose any traffic if a valid path exists and that transient loops only last at most the control packet dissemination time.

Theorem 2: *The control packet flooding mechanism is guaranteed to inform every AS that has a previously reachable AS that becomes unreachable due to the failure.*

Proof: The AS detecting the event (x_1) informs every neighbor directly connected to it (x_2). The control packet wave continues hop by hop until it reaches ASes in hop *n* (x_n) with either no valid paths using the affected link or having alternative paths such that all reachable ASes still remain reachable. Let's assume that an AS at hop *n+1* should receive a control packet and it does not. It can only happen if:

1. This x_{n+1} AS will lose reachability to some other AS (it should receive the control packet), and
2. All its x_n have alternative paths around the failure and reach all reachable ASes (it does not get it).

If all x_n neighbors have alternative paths to every reachable ASes the x_{n+1} AS will not lose reachability because it uses one of its neighbors. Therefore, condition 1 and 2 cannot occur simultaneously and Theorem 2 is proven by contradiction.

If multiple failures are present the control packets might not reach all desired destinations, if an AS is unable to send a link down control packet to one of its neighbors it should store the packet in order to send it when the neighbor becomes reachable, a new link down control packet must also be forwarded to all other neighbors informing of this new failure, by the same process described above.

When the failed link is restored, a link up control packet is flooded in the same manner identifying the link that is now available. The AS that detects the link up event sends any pending link down packets it might have for that neighbor. The ASes that receive link up control packets remove the marking of failed from the valid path that contains the link. Theorem 2 also applies in this case, every AS that has a valid path using the repaired link is notified and therefore removes the marking from the path. When an AS receives a link up control packet it invalidates possible pending link down control packets for the same link in a multiple failure scenario.

It is possible that repeated control packets are received, since an AS can receive the control packet from more than one neighbor even without multiple failures, a second control packet with the same information (either a link down packet or a link up packet) is obviously ignored.

Theorem 3: *Transient loops caused by control packet inconsistency are contained to one hop and packets loop at*

Proof: Consider $P_{ij}=\{x_{i1},x_{i2},\dots,x_{ik},\dots,x_{in-1},x_n\}$ with $1 \leq k \leq n$ and $1 \leq i \leq number\ of\ valid\ paths\ to\ x_n$ the set of valid paths from AS j to AS x_n. At each x_k along the way a similar set exists. Assuming a multi-path routing algorithm any of these paths can be used. A failure invalidates one or more of these paths, an can cause loops. A loop occurs when one AS has processed the control packet but some of its neighbors did not. In this case it can happen that for a given x_{ik} the next hop after the failure is x_{ik-1} that still has x_{ik} as the next hop forcing the packet to return. The loop is contained to one hop, and occurs at most one time because if x_{ik} is already using alternative paths it will forward the control packet to x_{ik-1} just after processing the data packet, or the control packet is on hold due to link failure. If traffic arrives from x_{ik-1} no failure on the link x_{ik-1}- x_{ik} exists, the control packet arrives at x_{ik-1} when data packets have looped at most once. The link x_{ik-1}-x_{ik}, is invalidated and one of two situations can happen: an alternative path exists and the packet is forwarded to it (and not to x_{ik}), or no path exists and the packet is discarded.

Theorem 4:

Condition 1: There is at least one available valid path to the destination D during failures.

If condition 1 holds no packet p is lost during the failures

Proof: Let *G* be the region static graph, and *DG(t)* the region dynamic graph at time *t* (with the failed links marked as down). The set of all valid paths rooted at a given AS X is *P(X)*. *DP(X, t)* is obtained by marking failed paths at *t* in *P(X)*. In order to exist at least one valid path to a destination *D*, *D* must be a vertex of *DP(X,t)* (i.e. $D \in V(DP(X,t))$).

Condition 1 can then be stated as $D \in V(DP(X,t))$ for a given root node X in *V(DG(t))*.

For every packet *p* flowing from source AS A to destination AS D if *p* encounters a failure at AS X, there are only three possible reasons for packet *p* to be dropped:

1. If there is no valid path from X to D at time *t*. It means that $D \notin V(DP(X,t))$ which contradicts condition 1.
2. There is a valid path ($D \in V(DP(X,t))$) and assume the next hop is AS X_1 that forwards the packet back due to the configuration of its set of paths, $DP(X_1, t)$ (i.e. one of the ASes – X or X_1 – has received the control packet and the other one has not). Theorem 3 guarantees that the same AS is only visited at most twice. The AS receiving the packet will behave according to situation 1 or 3.
3. There is a valid path ($D \in V(DP(X,t))$) but a new failure occurs at t_1 that affects this new path. Again for a packet to be dropped at time t_1 we must have $D \notin V(DP(X,t_1))$ which violates condition 1.

Therefore, a packet can only be lost if at given time *t* if condition 1 is not true, this proves Theorem 4 by contradiction.

The dissemination of the control packets can change the reachable AS set exchanged between regions. If a failure occurs in a link between regions the set of reachable ASes is just emptied and all traffic has to follow the normal process (through tier-1).

VI. Experiments

Our first set of experiments used a topology from the CAIDA AS Relationships Data research project [12] trimmed to obtain the topology of 76 countries in Europe and part of Asia with 11,335 ASes and over 21,000 links. It might be an excessively large region but it provides an insight about the upper limits of the architecture.

Figure 3 shows for an AS X how paths are accounted for: X-B is a path, and so is S-T; D-H (but not D-B, because H-B belongs to another path); R-D; etc. One link with a certain D appears only once in *P(X)*. For instance, in Figure 3 the link H-K appears twice: once with D=0, and the second time with D=1.

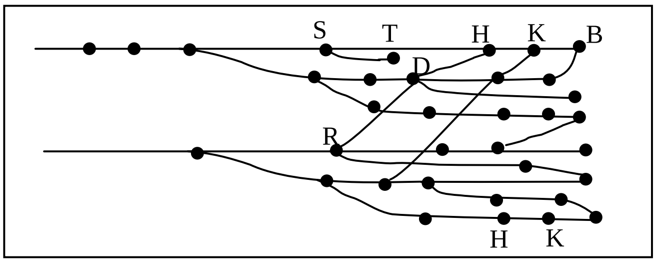

Figure 3– Illustration of paths in *P(X)*

P(X) had a mean value of 22,563 paths with a standard deviation of 7,275. The number of paths varied from 114 to 31,280 paths depending both on the position of the AS in the region (ASes with neighbors in other regions have a smaller number of paths) and the position in the provider-customer hierarchy. The calculation to obtain the largest *P(X)* took 0.2s with a 2,4Ghz microprocessor, 4G of memory and using JAVA, which is an acceptable time for a router.

The *FH(X)* table has a maximum of 169,293 different paths for some highly connected higher level ASes. It takes 0.7s to calculate. Note that the routing algorithm will work on the knowledge of the graph, the knowledge of *P(X)*, and the exchanged information from neighbors to rank the entries of *FH(X)* to create a multi-path system.

We made a second set of experiments to test a highly aggressive use of multihoming. The purpose is to see how the system scales with multihoming. The topology had 4 levels with 3, 81, 3,215, and finally 8,036 stub ASes (to have the same number of ASes as above). Each stub AS was connected to 3 level-3 ASes. Each level-3 AS was connected to 10 level-2 ASes, and each level-2 AS was connected to 3 level-1 ASes. A uniform distribution was used to make the network very regular.

P(X) has between 53k and 64k paths and it takes around 0.3s to calculate. From bottom up, *FH(X)* has around 34k, 113k, 34k, and 911k entries. The time to calculate the smaller ones was around 2s, the middle one took 4.5s, and the largest one took 35s. Naturally, multihoming increases the number of valid paths. But it is interesting to see that even with this aggressive topology the numbers do not explode. Although the number of 911k seems very large, *FH(X)* is used mainly by routing protocol to rank exit links. So, for each destination AS only a small number of entries is relevant.

Another purpose of the experiments was to test the failure management algorithm. For the first data (real data) a high percentage of single link failures produce no control packets (alternative paths exist). For the second data it is 100%. Note that although there are no packets at reachability level there surely are at routing algorithm level. Failures produce routing events to be sent by the routing protocol to avoid loops. Given the characteristics of the Internet graph we think the scope will not be large. This is the following step in our research.

VII. deployment

The deployment process cannot be based on a synchronized change of the entire world at the same time. We assume that the graph can evolve from the effort of RIPE and that there is a mapping service to know the ASes from prefixes.

ASes running BGP-4 stay as they are. The new system has to be deployed from bottom to up. An AS can only change if all its customers have changed. Regions start to exist with graphs containing one, two, three ASes. Each time an AS receives prefixes from the old world it translates them to ASes and learns destinations. It advertises prefixes from the valid ASes it can reach by mapping ASes to prefixes and choosing one valid path if more than one exists.

VIII.related work

The work most similar to ours is HLP [9].They use also AS identifiers instead of prefixes and use trees based on tier-1 ASes to contain the scope of the link-state protocol. However, if multihoming starts to be used extensively (as in our second experiment) all ASes in HLP will have to run three link-state protocols (with eleven thousand other ASes). HLP does not scale with multihoming. As HLP has to construct the graph, the link-state (and the path-vector) protocols are used, and the routing events for any change have a larger range than in our case. In our case, the scope of control packets for the reachability part was analyzed. For the routing part, we estimate a shorter range following the policy-restricted graph. HLP fails also to address backup links and does not take into consideration the web of peer-to-peer links that exists already (they only consider one-hop peer links). NIRA [10] is another related work to ours. Although their scope is different there are two subtle aspects that are present both in NIRA and in our system that we consider very important: the assumption of a static graph that is quite immutable (with some sort of failure management); and the construction of *one* topological graph based on business relationships instead of *various* graphs based on prefixes. Their choice of not having an entity to provide the graph complicates, in our opinion, the overall system. They use a path-vector protocol to construct the static graph and a policy-controlled link-state protocol to manage failures.

IX. Conclusions and Further Work

This paper proposed a possible architecture for Internet inter-domain reachability offering some advantages over BGP in issues such as multi-path routes, multihoming, backup links, and sibling relationships. There is a strong containment on churn and route events which is greater with the degree of multihoming. Packets could be forwarded quicker based solely on AS numbers that could exist in an optional IP header.

However there are no secrets. Compromises have to be made in some issues and certain features have to be considered secondary. Nevertheless, we think it preserves the main characteristics of the Internet, and specially its business model. Above all its deployment is also credible.

This work opens new directions of research. Algorithms used to calculate paths and work on the graphs can be improved. A new routing protocol that works with multi-path routes over the graph and featuring load balancing and traffic engineering (or having economic considerations) can be very challenging.

References

[1] Yannuzzi, M.; Masip-Bruin, X.; Bonaventure, O. *Open issues in Interdomain routing: a survey,* Network IEEE Nov.-Dec. 2005 Volume: 19, Issue: 6

[2] Matthew Caesar, Jennifer Rexford. *BGP Routing Policies in ISP networks*, IEEE Network Magazine, special issue on interdomain routing Nov/Dec 2005.

[3] L. Gao. *On inferring Autonomous System relationships in the Internet.* In IEEE/ACM Transactions on Networking, December 2001

[4] B. Zhang, R. Liu, D. Massey, and L. Zhang. *Collecting the Internet AS-level topology*. ACM SIGCOMM Computer Comm. Review (CCR), 35(1): 53–61, 2005.

[5] T. G. Griffin, F. B. Shepherd, and G. Wilfong, *The Stable Paths Problem and Inter-domain Routing*, IEEE/ACM Trans. Net., vol. 10, no. 2, Apr. 2002, pp. 232–43.40 papers in national and international refereed journals and conferences.

[6] R. K. C. Chang and M. Lo, *Inbound Traffic Engineering for Multihomed ASes Using AS Path Prepending*, IEEE Network, Mar. 2005

[7] Benoit Donnet and Olivier Bonaventure, *On BGP Communities* ACM SIGCOMM Computer Communication Review, vol 38. No. 2, 2008.

[8] Olivier Bonaventure, *Reconsidering the Internet Routing Architecture.* Internet draft, draft-bonaventure-irtf-rira-00.txt, 2007.

[9] Lakshminarayanan Subramaniam, Mathew Caesar Cheng Tien Ee Mark Handley *HLP: A next Generation Inter-domain Routing Protocol,* SIGCOMM 2005, Philadelphia

[10] Yang, X., Clark, D., Berger, W. A., *NIRA: A new Inter-Domain Routing Architecture,* IEE Transactions on Networking, Vol 15. No. 4, August 2007

[11] RIPE database, http://www.ripe.net/db/index.html.

[12] CAIDA.ASRelationshipsData.ResearchProject.http://www.caida.org/data/active/as-relationships/

[13] Xenofontas Dimitropoulos, et al. *AS Relationships: Inference and Validation*ACM SIGCOMM Computer Communication Review, 2007.