

# Route Control Platform Architecture and Practical Concerns

Nick Feamster  
MIT Computer Science & AI Lab  
feamster@csail.mit.edu

Jennifer Rexford, Aman Shaikh, Jacobus van der Merwe  
AT&T Labs—Research  
{jrex,ashaikh,kobus}@research.att.com

## Abstract

In this paper we describe a network architecture in which the inter-AS routing decision process is removed from routers into a separate logically-centralized control function which we refer to as the Route Control Platform (RCP). This architecture makes a clean separation between the packet forwarding function which by necessity has to be performed by network elements in the data path, and the control function performed by the routing protocols which directs how the data forwarding should take place. Removing the current tight coupling between these two functions thus removes the burden of route computation from routers so that it does not interfere with the packet forwarding process. More important, however, is the fact that removal of this function simplifies router configuration and enables flexibility in the route selection process that is simply impossible in the current infrastructure. We provide an overview of the RCP architecture and show how it can be realized within the existing routing infrastructure requiring *only* configuration changes on the routers. We present a high level view of the RCP software architecture and identify issues that need to be taken into account in its realization.

## 1. Introduction

Adding value-added services to data networks is challenging, in large part because (i) a network’s control plane is distributed across the individual network elements (such as routers) and (ii) changing or customizing this functionality is difficult. This forces service providers to work within, or around, the constraints imposed by the existing network equipment and protocol standards [4, 6, 8]. Placing more of the control state and logic into a logically centralized point “above” the network elements simplifies the operation of the network and enables a diverse collection of new services. We will refer to such a logically centralized point as a *Routing Control Platform (RCP)*. To enable backwards-compatibility with the installed base of network equipment, the RCP should communicate with the network elements using standard protocols, such as the Border Gateway Protocol (BGP) [10]. The RCP can use BGP to send and receive routes for any defined address family, such as IPv4 or VPN, and set route attributes to drive the routers’ decisions (e.g., for path selection, QoS policies, data collection, and traffic filtering). By necessity logically centralized does not mean physically centralized and an instance of the Route Control Platform (RCP) is made up of a number of distributed components namely *Route Control Servers (RCSes)*.

In the next section, we describe the key architectural components of an RCP. We show the need for a distributed realization of this logically centralized function and consider how the architecture will impact the protocol dynamics compared to the current fully dis-

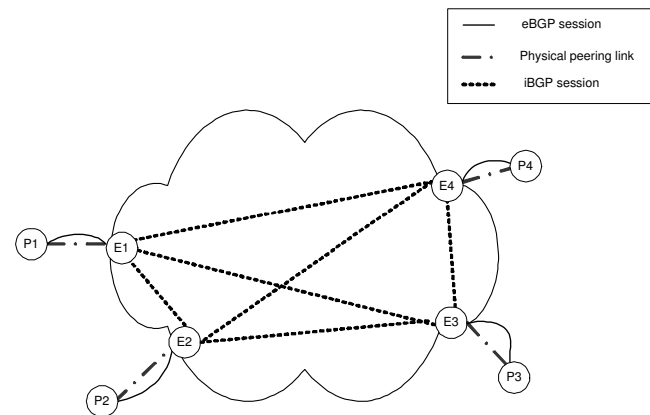


Figure 1: Current Inter-AS Routing Architecture

tributed case. We then present mechanisms that ensure backwards-compatibility with existing BGP-speaking routers and allow for incremental deployment of the RCP in a network. We present an overview of the functional components that make an RCS and consider the scalability challenges of such a realization. The paper ends with a conclusion.

## 2. Route Control Platform Architecture

The Route Control Platform (RCP) architecture is best explained by showing how it changes the current inter-domain routing architecture. The current inter-AS routing architecture is depicted in Figure 1. This figure shows an autonomous system (AS) with four border routers (E1 to E4). Each border router is connected to a border router in a neighboring AS (P1 to P4), through a “physical” peering link. Each pair of border routers (i.e., E1 and P1, E2 and P2 etc) runs an exterior BGP (eBGP) session between them over the physical peering link. Within the AS, the border routers (E1 to E4) exchange routing information through interior BGP (iBGP) sessions. The iBGP sessions are routed through the AS using an interior gateway protocol (IGP) such as open shortest path first (OSPF). In order to make all available routes “visible” to all border routers, the iBGP sessions are required to form a full mesh between the border routers in the AS. Note that in this description we are ignoring the potential use of iBGP scaling mechanisms like route reflectors [1]. Indeed, not requiring the use of route reflectors and thereby avoiding the problems associated with that is one of the motivating factors for the RCP architecture.

The RCP can be applied to either the iBGP sessions in an AS, or to the eBGP sessions or to both. A likely deployment scenario would involve first applying the RCP to the iBGP sessions and then

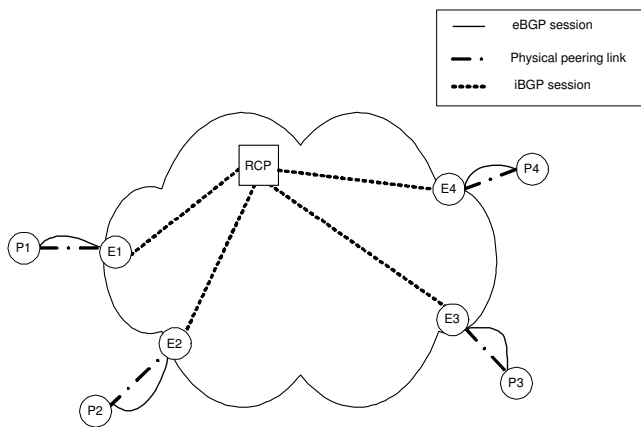


Figure 2: RCP with iBGP

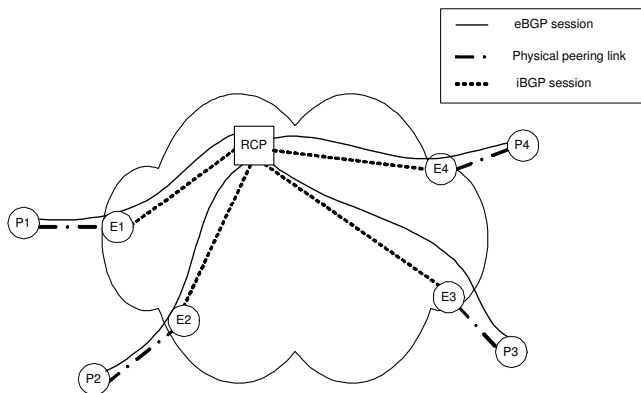


Figure 3: RCP with iBGP and eBGP

extending the deployment to also include some or all eBGP sessions. In the following discussion we will describe this architecture and deployment scenario.

**RCP with iBGP:** Figure 2 shows the iBGP sessions when an RCP is deployed<sup>1</sup>. Rather than having iBGP sessions with all other border routers, the border routers (E1 to E4) maintain an iBGP session only with the RCP. The border routers do not require any other changes and simply export routes to the RCP following normal BGP processing rules. That is, each border router will export all selected routes to the RCP. In this simple scenario the main function of the RCP is to select the routes for each border router that the border router would have selected itself had there been a full iBGP mesh. Since part of the “normal” route selection process involves the AS topology information (i.e., information gathered from the IGP), one of the auxiliary inputs to the RCP is such AS topology information. Monitoring the topology information in an AS is typically done anyway (e.g. for performance monitoring and root cause analysis) [11].

**RCP with iBGP and eBGP:** Using the RCP to replace the iBGP mesh has many benefits [3] but the routes that are exposed to the RCP are still “filtered”. That is, for each destination prefix a border router has to select one route and export that to the RCP. Allowing the RCP to “see” all available routes (not just the routes selected by border routers based on its own local information) would remove

<sup>1</sup>In this Section we show the RCP as a single box, but as indicated in the introduction, the logical RCP is made up of a number of RCSes.

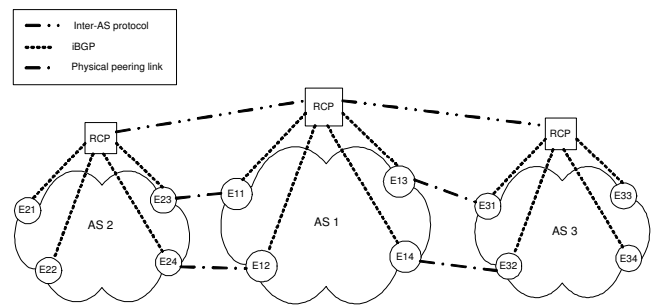


Figure 4: RCP for inter-AS route exchange

this restriction. This can be achieved in a number of ways. One way is to have eBGP sessions with border routers in neighboring ASes terminate on the RCP rather than on the border routers in the RCP’s AS. This scenario is depicted in Figure 3, where border routers (P1 to P4) in neighboring ASes now have eBGP sessions directly with the RCP. As before the RCP maintains iBGP sessions with the edge routers (E1 to E4) in its own AS to exchange routing information with them. Again no changes (other than small configuration changes) are required on either the edge routers in the local AS or the border routers in the neighboring ASes. Note that the effect of this deployment scenario is that all routing decisions are effectively made in the logically centralized RCP. In this case the iBGP speakers effectively make no local selection of routes. That is, the iBGP speakers will still go through the “normal” route selection process but the RCP will ensure that they only “see” (and therefore select) the routes that the RCP deemed appropriate for them to select. In this case the function of the RCP extends to exchanging appropriate routes with eBGP speakers. That is, not only does the RCP have to make routing decisions for its edge routers (the iBGP speakers) but it also has to communicate such selections to the eBGP speakers (associated with the appropriate edge router through the physical peering link), because the edge routers no longer maintain eBGP sessions with the border routers in neighboring ASes.

**RCP for inter-AS route exchange:** Finally, a scenario is feasible where RCPs in different ASes exchange routing information with each other, thereby eliminating the need for eBGP. This scenario is depicted in Figure 4. In this case the RCP will still use iBGP to communicate with the routers in the AS in which it is operating. However, there is no need for RCPs to use BGP to communicate with each other. Instead the inter-AS communication might be based on a new protocol.

We refer to the RCP architecture as “logically centralized” to emphasize that in terms of reasoning and functionality we want to think of the RCP as being centralized, while in practice the realization has to be somewhat distributed to avoid a single point of failure. In Section 2.1 we consider what logically centralized means in practice and address several practical issues with this approach.

Our approach does not require new functionality to be deployed on routers and the RCP can be realized by *configuration only* changes to the routers. Backward compatibility is principally achieved by using the BGP protocol unchanged to communicate with routers. Routers have to maintain these BGP sessions with the RCP rather than with each other and therefore all RCP-clients also require IP reachability to the RCP. Both of these can be achieved with fairly simple configuration changes on the routers. We address these issues in Section 2.2 below.

## 2.1 Logically Centralized

While a centralized architecture simplifies reasoning it is clear that the RCP can not be realized as a physically centralized component. In Section 2.1.1 we consider the options available for implementing the RCP architecture in a robust manner.

A centralized architecture (even a logically centralized architecture), has different protocol dynamics from that of a fully distributed architecture. For example, in our architecture link state changes will have to percolate to the RCP infrastructure via IGP before the RCP can take action and change routes for affected routers. Similarly, in our centralized architecture the BGP sessions (including eBGP sessions) will typically have to go over a number of intermediate routers to reach the RCP. This might have an impact on the stability of these BGP sessions. In Sections 2.1.2 and 2.1.3 we consider respectively the timeliness and the stability of our approach.

### 2.1.1 Robustness

Redundancy requirements mandate that the RCP architecture is not implemented as a physically centralized entity that would result in a single point of failure. From the router point of view this can easily be met by having each router maintain  $n$  BGP sessions to different RCPs (or RCSes making up the RCP). As long as these RCSes make decisions based on the same input data, their results will be consistent and routers will receive the same routes from all RCSes. In practice, however, it is not possible to guarantee that all RCSes are using the same input data and this inconsistencies can easily result in the creation of routing loops. This can be explained by the simple example depicted in Figure 5.

As shown in Figure 5, assume that we have two RCSes (RCS1 and RCS2) that make up the RCP, two iBGP speaking routers (E3 and E4) and two eBGP speaking routers (P3 and P4). Further assume that the network is in steady state and that P4 is used to reach a certain prefix, a fact that both RCSes agree on since they use the same inputs. Now suppose that RCS1 loses its BGP sessions to both E3 and P4 because of a networking issue in a different part of the network. That is, the routers P4 and E3 are still operational and maintain their BGP sessions to RCS2. For simplicity, also assume that RCS2 loses its BGP session with E4 as shown. Suppose that the prefix in question is also reachable via P3. Based on its inputs RCS1 will therefore decide to reroute all traffic to this prefix via P3 and will inform E4 about this fact. RCS2 in the mean time has no reason to do the same and (still) tells E3 to reach the prefix via P4 and a routing loop has thus been created between E3 and E4.

Although somewhat contrived the example illustrates that the RCSes must maintain a consistent view of the available routes to ensure that all routers receive consistent, loop-free paths. We are currently studying the types of inconsistencies that can result from various combinations of partitions. Our preliminary results suggest that even if a network is partitioned, RCSes in separate partitions cannot create a forwarding loop. This result follows from the fact that network partitions are caused by partitions of the IGP (e.g., OSPF) topology, and RCSes rely on the IGP to exchange routes with each other and with BGP routers. Thus, a protocol that elects an RCS for each partition guarantees correct, loop-free forwarding.

### 2.1.2 Timeliness

An important consideration is how quickly the RCP architecture can converge compared to the conventional BGP architecture. A major aspect of convergence depends on how quickly BGP updates can propagate to the RCP. Furthermore, in the event of an internal change, the RCP has to depend upon the IGP to learn about the

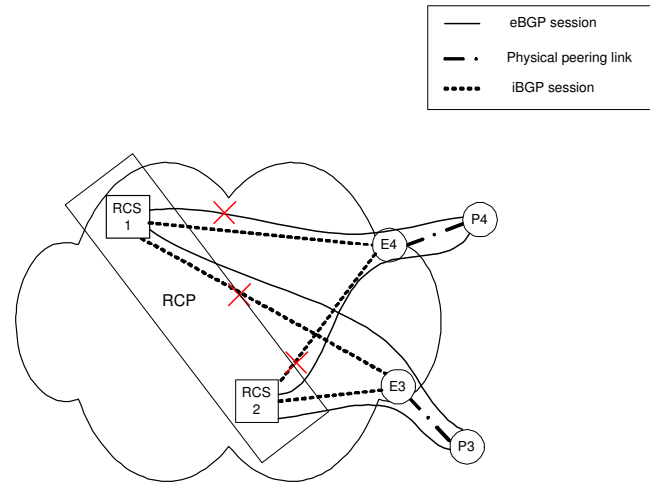


Figure 5: Inconsistent RCSes leading to a routing loop

change before it can select appropriate BGP routes. We discuss update propagation and IGP change propagation here. As for update propagation, we need to consider two factors while comparing the RCP architecture with the conventional BGP architecture. First, the RCP is likely to have multi-hop BGP sessions with most of the routers. However, the number of hops should not be very high since the diameter of even large ASes (belonging to tier-1 ISPs) should be in the order of tens of routers. Second, we need to consider the impact network congestion might have on these multi-hop BGP sessions. Although the conventional BGP architecture is likely to have very few multi-hop sessions, it will typically have a lot more sessions than the RCP architecture. Therefore, we believe that congestion is likely to have similar impact on both the architectures. Based on these two arguments, we believe that the update propagation time should not increase significantly with the RCP architecture. In fact, where iBGP hierarchy is used, moving away from it might improve update propagation since updates do not have to go through BGP processing, timers such as MRAI (Min-Route Advertisement Interval) and/or path exploration at every router along the path through the hierarchy.

Now let us consider the IGP change propagation in the RCP architecture. Note that even with the conventional BGP architecture routers have to learn about IGP changes before they can make BGP decisions. With link-state IGPs such as OSPF and IS-IS, the time to learn of an internal change would depend on the locations of the RCP and the change. If strategically placed, the RCP on an average should be able to learn of an internal change at the same time — if not before — the router(s) whose BGP decisions depend on the change.

### 2.1.3 Stability

The stability of the BGP sessions in the face of network conditions such as failures and congestion will have a significant impact on the reliability of the architecture as a whole. As was the case with timeliness of update propagation, the concern here is whether having multi-hop BGP sessions can have an impact on their stability or not. The stability of BGP sessions with RCP should not be any worse than the stability of the sessions in a full-mesh iBGP topology since BGP sessions in a full-mesh iBGP topology are likely to be multi-hop as well. Even with an iBGP hierarchy the BGP sessions can be multi-hop in theory, though in practice they are most likely to traverse a few hops at most since router-reflectors

are placed close to their clients to avoid deflections [7]. Though the BGP sessions to the RCP are likely to have more hops than the average number of hops with the iBGP hierarchy, this should not decrease the stability of these sessions. The reason is that the number of extra hops is not likely to be significantly higher and all the extra hops will still be within a single AS. To substantiate this, we looked at the session resets in a Tier-1 IP backbone from January 2004 through March 2004. In particular, we compared the number of resets for a few multi-hop sessions against a few single-hop sessions. We found that the multi-hop sessions were as stable as the single-hop sessions. We also looked at all iBGP session resets within the network, and found that only about 13% of these happened outside the maintenance window. This should alleviate any concerns about the stability of the iBGP sessions in the face of network failures.

## 2.2 Backward Compatibility

A very important consideration in deploying any new networking technology is how that technology would impact and interact with currently installed networks and network elements. Specifically, any new technology that requires new functionality to be installed in existing network elements, or require new functionality to be deployed everywhere before it becomes useful is problematic.

The RCP avoids these problems by being fully backwards compatible with existing router functionality. That means that *only* configuration changes are required on existing routers to enable RCP functionality. Further, the RCP approach allows for incremental and parallel deployment strategies that would not require any network down time. We consider the configuration changes and deployment strategies for each of the different scenarios discussed in the previous section.

The first condition that has to exist to enable the RCP approach is that the IP address(es) of the RCP need to be reachable from within the AS in which it is operating. This can be achieved by advertising the RCP IP address(es) through the IGP (e.g., through OSPF).

### 2.2.1 RCP doing only iBGP

This option refers to Figure 2. In this case, the only configuration change that is ultimately needed is that existing iBGP configuration should be replaced with an iBGP session to the RCP. Following a parallel deployment strategy, however, the new iBGP session to the RCP can be configured while the existing iBGP sessions remain intact. During this phase of deployment the RCP can be configured in “read-only” mode whereby it receives BGP updates from the router in question but does not export any routes to the router. Once all routers in the network have been configured in this way, the RCP can be configured in “read-write” mode (possible on a per router basis), so that the routers will also receive routes from the RCP. Finally, the configurations for the old iBGP sessions can be removed.

In an iBGP only deployment of an RCP, routers will accept routes from the RCP within the constraints of the current BGP decision process. Specifically, all other things being equal, a router will prefer an eBGP route to a specific destination over an iBGP route to the same destination received from the RCP. The RCP can force a router to accept routes sent by the RCP by setting route attributes, such as local preference, to make its routes more attractive than any BGP route learned from another source. However, doing so will cause the router in question to withdraw any route sent to the RCP to the same destination with a lower local preference, thus reducing the RCPs “visibility” into the full set of routes available to the destination. Having the RCP completely override the router decision

process is, however, the desired behavior for some RCP applications.

### 2.2.2 RCP doing iBGP and eBGP multi-hop

This option refers to Figure 3 where the RCP maintains both iBGP to routers in its AS and eBGP sessions to peer routers in other ASes. As far as the iBGP sessions are concerned the same considerations apply as for the iBGP only case.

Since the eBGP session is with a router in a neighboring AS, this scenario requires configuration changes to the router in the neighbor AS. With reference to Figures 2 and 3, we are assuming for this discussion that routers P1 and E1 in Figure 2 need to be changed to the setup for the same routers in Figure 3.

First the neighbor router (P1) needs to be configured with an eBGP session to the RCP. This session has to be multi-hop enabled as the RCP will be more than one router away from P1. Further, the IP address of RCP needs to be reachable from P1. The simplest way to achieve this is to configure P1 with a static route to the IP address of the RCP and associate that with the interface(s) that connect P1 and E1.

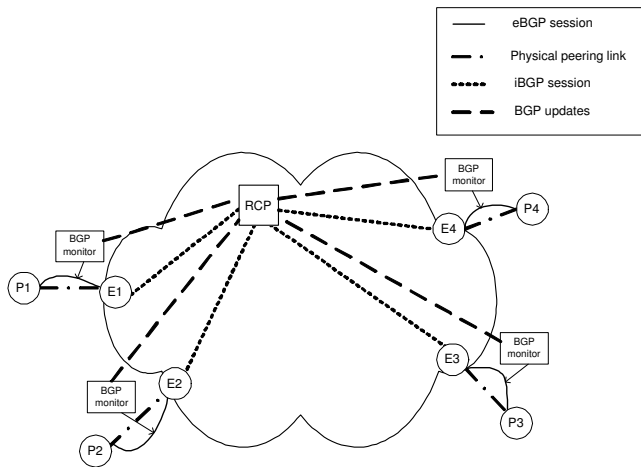
In the reverse direction, both RCP and E1 need to know how to reach P1. The simplest way to achieve this is to configure E1 with a static route to P1 that is associated with the interface that connects P1 and E1. E1 then redistributes this static route into the IGP (OSPF) which in turn allows RCP to reach P1. Alternatively, the RCP could be configured with a static route that associates P1’s end of the eBGP session with the loopback address of E1; if E1’s loopback address is announced in the IGP, the RCP would know to forward packets destined to P1 toward the router E1.

Another way to deal with the reachability problem is to use a “bootstrap” eBGP session between E1 and P1. This bootstrap session could advertise the IP prefix for the addresses of the RCSes. These bootstrap routes could be advertised with the “no advertise” community, or the neighboring AS could configure filters so that the routes for the RCSes are not re-advertised among routers in the neighboring AS. As before E1 can redistribute reachability to P1 into IGP to allow the RCP to reach P1.

The above configuration options would allow P1 and RCP to establish an eBGP session and forward the BGP messages as IP packets through E1. RCP then sends selected routes to P1 with E1 as the next-hop IP address. This should in general not be a problem, i.e., P1 should be willing to accept these routes unless it uses the configuration option that automatically over-writes the next-hop attribute with the peer IP address, e.g. the “set ip next-hop peer-address” command in Cisco IOS. This option should therefore not be set on the P1 router.

The configuration as described above forces the eBGP session between P1 and RCP to go through E1. This is beneficial as it has the desired side effect that the eBGP session will be torn down, and routes previously received from P1 withdrawn, when the link(s) between P1 and E1 go down. This behavior therefore mimics the current eBGP behavior (e.g., between P1 and E1 in Figure 2), where the physical link health between P1 and E1 is tied to the eBGP session it carries.

Note that it is possible to use other configurations where the eBGP session between P1 and RCP is *not* tied to the physical link(s) between P1 and E1. That would imply that it might be possible for the eBGP session between P1 and RCP to remain operational even when the physical peering link between P1 and E1 is down. Clearly in this case another link-health mechanism would have to be in place to inform the RCP that the peering link is down and that routes learned from P1 should no longer be used. (For example,



**Figure 6: Alternative RCP with eBGP realization**

the address block of the link(s) could be advertised in the IGP, or announced via a separate iBGP session to a health monitor, to inform the RCP of changes in the up/down status of the link(s). In principle the RCP could still maintain the eBGP relationship with P1 so that it has all the routes available for use immediately after the peering link is restored.

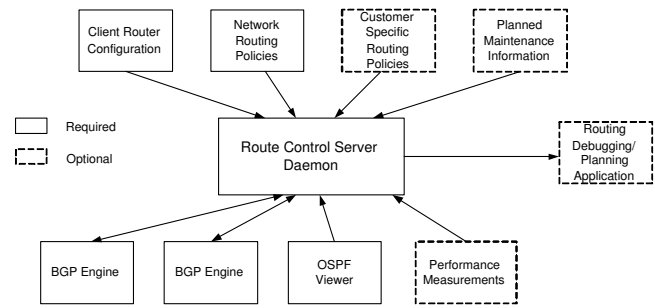
As in the “iBGP-only” scenario, a parallel deployment strategy can be followed. In particular, P1’s eBGP session to the RCP can be configured in parallel with its existing eBGP session to E1. P1’s session to E1 can then be torn down at a later point in time. Also, setting up eBGP sessions can be done one peer router at a time as that will have no impact on any other routers.

### 2.2.3 RCP doing iBGP and receiving eBGP routes

The previous case where the RCP maintains eBGP sessions with routers in neighboring ASes is architecturally cleanest but does require changes to the configuration of routers in other ASes. Figure 6 depicts an alternative solution which does not require any changes to routers in neighboring ASes. This allows an AS to provide the same functionality as the previous scenario without changes in the neighboring ASes.

In this case the RCP maintains iBGP sessions with the routers in its AS as before. These edge routers still maintain their eBGP peering relationship with the routers in neighboring ASes, e.g. P1 and E1 in Figure 6. To give the RCP access to all of the eBGP routes, some form of BGP monitor is deployed to snoop on this eBGP session. Various options exist for how this snooping can be realized. For example, router vendors might provide such functionality as a software module on their routers which essentially sends a copy of all eBGP-learned routes to the RCP. An alternative would be to deploy a physical monitoring box in close proximity to the router so that it can snoop on the eBGP session and send all updates to the RCP. (For example, a packet monitor on the P1-E1 links could track all TCP packets using the BGP port number. This monitor could then relay these raw packets, or the reconstructed BGP messages, to the RCP.)

A significant difference in this case is that the edge router in the RCP network (i.e., E1) still partakes in the BGP route selection process based on the routes that it receives from its eBGP peers. And it will dutifully send such selected routes on to the RCP as per normal iBGP procedures (unless this is prevented through configuration on E1). Based on its own route selection process, the RCP might how-



**Figure 7: Route Control Server Architecture**

ever decide that the router (E1) should have selected a different route. Using iBGP, the RCP can inform the router that it should use this other route and the RCP will set the next-hop attribute of the route to P1 (or to whatever other egress point, such as E2, that E1 should use as the next-hop). The RCP can use a higher local-pref route attribute for the route that it wants E1 to select. This way it can force E1 to pick the RCP-selected route over the route that E1 would have picked on its own.

Another way to address this problem would be to configure the edge router in the RCP network (E1) to filter out all routes it receives from the peer router (P1). The RCP will receive all the routes advertised by P1 via the BGP monitoring function and will instruct E1 about the routes it should use via iBGP as before.

## 3. Route Control Server Architecture

Having dealt with the way a RCP would be deployed within a network we now zoom in to consider the internal architecture of an individual Route Control Server (RCS). Figure 7 depicts at a logical level the different functional components and inputs that make up an RCS. At the center of the system is the RCS daemon. The different components of the RCS can be implemented on a single physical device, or each component might be implemented on a different physical device. The RCS daemon interacts with one or more BGP engines that are responsible for maintaining BGP sessions to iBGP and/or eBGP speakers. The BGP engines are responsible for managing the details of the BGP protocol in terms of message exchanges, timer management etc., but all routes received by the BGP engines are passed on to the RCS daemon who is responsible for route selection. Routes selected by the RCS daemon for each BGP neighbor are passed back to the BGP engine(s) which then exchange the routes with the appropriate BGP neighbor using the BGP protocol.

As discussed above the RCP/RCS needs topology information for the AS in which it is operating. This can be provided by a component that monitors the IGP routing protocol in the AS. For example, as shown in Figure 7, an OSPF viewer keeps track of the link-state advertisements (LSAs) exchanged by the OSPF protocol in the AS and provides this topology information to the RCS daemon.

The RCS daemon also needs various types of other inputs that are not obtained from the current running network. Examples of such inputs to the RCS daemon are shown at the top of Figure 7. First the RCS needs to be configured with information about the BGP neighbors (i.e., client routers) that it needs to communicate with. This would include the neighbor’s IP address and whether the neighbor is an iBGP or an eBGP speaker. The RCS is also configured with the desired routing policies for the network as a whole. Examples include which routes are being accepted from

which neighbor, what routes are being advertised to each neighbor and what routes are never accepted into the system.

Given this set of components the RCP is able to provide the same functionality as the current BGP routing architecture, but with much simplified per-router configuration and without many of the problems associated with scaling of iBGP in an AS. However, the RCP also enables new functionality and applications. That is exemplified by the remaining “optional” components shown in Figure 7.

First, the RCP can make use of current traffic conditions (e.g. load or other performance metrics) as an input to its route selection process. Providing this information to the RCS daemon is the function of the performance measurement component in Figure 7.

Since the RCP is free to use any appropriate algorithm to perform route selection, customer specific routing policies can be introduced in the AS. For example, a customer’s voice-over-IP (VoIP) traffic may be routed via a path that is different from the default shortest path because it has more desirable characteristics for that application than the default path.

Another application of the RCP would be to allow planned maintenance of the network to be taken into account to make sure that no network disruption would be caused by such maintenance. For example, if a network link needs to be taken down for maintenance, the RCP can in a controlled manner move traffic off the affected link before maintenance is started and in such a way that no other links are overloaded in the process. Because of the direct control that the RCP is exercising over the routes that are selected in the network, the RCP can ensure that there will be no unexpected side effects during such a procedure.

Finally, the right most component in Figure 7 shows another type of application that is enabled by the RCP. Because the RCP has access to all routes that are available in the AS in one (logically) centralized place, it becomes much easier to perform debugging of routing problems or to analyze the routing system as a whole for planning purposes.

### 3.1 RCS Daemon Processing

A simplified view of the internal structure of the RCS daemon is shown in Figure 8. Starting from the left, routes advertised by BGP speaking neighbors are received by the RCS daemon (from the BGP engines as described above). The RCS daemon first applies per neighbor filters to get rid of any routes that should either not be allowed at all or that are not accepted from the neighbor in question. All accepted routes go into a repository, the Routing Information Base In (RIB-In), where all routes are stored whether or not they will be selected for use in the network or not. This allows the RCS to have alternative routes readily available should other routes be withdrawn. Keeping a repository of all routes also enables applications such as the debugging/planning application described above which needs as much information as possible.

For each received route the RCS decides whether that route should be used from the point of view of each iBGP speaker in the AS. As part of this process IGP topology information is used as shown in the figure. If a route is selected on behalf of an iBGP speaker, per-neighbor filters can be applied to the route (e.g. to modify attributes of the route if needed) and the route is then placed in a per-neighbor routing table that the RCS maintains, the RIB-Out. These routes are then communicated to a BGP engine which proceeds to exchange the route with the iBGP speaker.

The process described above holds for the case where the RCS only maintains iBGP sessions and the case where it maintains iBGP and eBGP sessions. In the latter case, however, the RCS also needs to communicate appropriate selections to the eBGP speakers “asso-

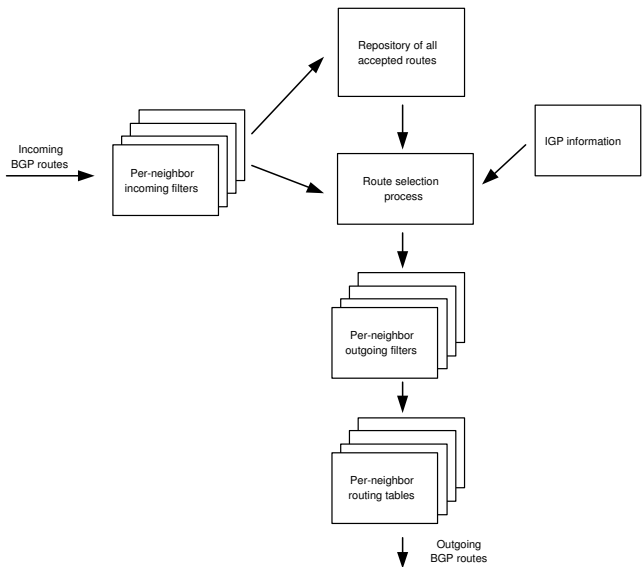


Figure 8: RCP Daemon Functionality

ciated” with iBGP speakers. For example, as shown in Figure 3, the border router in the AS in which the RCS is operating, say for example E1, no longer maintains an eBGP session with its “logical” eBGP peer, P1. Therefore, when the RCS selects a route for use by the iBGP speaker, E1, it should also decide whether that route should be exported to the eBGP “peer” P1.

### 3.2 RCS Daemon Scalability

In this Section we consider the scalability of a single instance of an RCS. We have indicated in Section 2.1.1 that the RCP architecture as a whole requires a distributed realization. As part of a distributed implementation the workload can potentially also be partitioned and distributed. The simplest solution would however be to have  $N$  identical RCSes. In what follows we therefore consider the scalability of a single RCS assuming that it needs to scale to realize the RCP on its own. The scalability of such an approach would clearly be the most demanding in terms of resources required by an RCS.

While our emphasis is on the scalability of a single RCS, we note that moving BGP decision making onto the RCP has a number of desirable scalability effects on the network as a whole:

- The RCSes can be deployed on general purpose servers for which the technology curve in terms of memory and processing power is much more aggressive than for routers. Similarly, scaling resources is easier for general purpose servers than it is for routers, either directly by adding resources to a server or through clustering techniques.
- The RCP can control how much resources are used on routers. Specifically, the RCP can ensure that no router receives more routes than what it’s physical memory is capable of, thereby preventing the catastrophic consequences of exceeding the physical memory requirements [2].
- Similarly, by moving the processing involved with BGP decision making into the RCP the routers themselves are not required to perform this processing, or perform very limited processing. This means that concerns about the impact of high router CPU load on data forwarding is eliminated [5].

For a single RCS, the resources of concern include the number of concurrent connections that has to be maintained, the amount of memory needed, the bandwidth requirements to get route updates to and from the RCS and the processing requirements to be able to make routing decisions in a timely fashion. Each of these are considered in more detail below.

First, the RCS would need to maintain a BGP session with each of its router clients. This would require the BGP Engine to maintain a large number (potentially thousands) of TCP connections in a network with many routers. Maintaining that many TCP connections might require changing the default parameters of the networking stack on many operating systems, but this problem has been solved in the context of Web servers which have similar requirements [9].

To estimate memory requirements, let's assume an implementation where for each client router the RCS maintains all routes received from that router in the Routing Information Base In (RIB-In). Similarly assume that the RCS maintains for each client router all routes sent to that router in the RIB-Out. In the most straight forward implementation the RCS maintains these routes separately for each client. In this case the memory requirements for each RCS-client would therefore be the sum of the RIB-In, the RIB-Out and the state required for maintaining the BGP connection. The latter will be insignificant compared to the RIBs and will be ignored for estimating memory requirements.

The number of routes in each RIB will vary greatly depending on routing policies and the role of the router (i.e. iBGP speaking routers, versus eBGP speaking routers). Further, the type of eBGP router (i.e. peering router vs customer router) would further have an impact on the number of routes. For example, customers might receive default routes only, partial routing information or full routing information from their provider. Also, since ASes typically do not provide transit services to its peers, routes received from peers are exported to customers but not to other peers.

To aid our discussion let's assume an imaginary ISP with the following characteristics in terms of number of sessions, number of routes imported from (RIB-In) and number of routes exported to (RIB-Out):

- 100 eBGP peering sessions, with 30K RIB-In and 10K RIB-Out routes.
- 300 eBGP customer sessions, with 1000 RIB-In and 130K RIB-Out routes, i.e., they export to the provider their internal routes and import full routing tables from the provider.
- 300 eBGP customer sessions, with 1000 RIB-In and 30K RIB-Out routes, i.e., they export to the provider their internal routes and import partial routing tables from the provider.
- 300 eBGP customer sessions, with 1000 RIB-In and 1 RIB-Out route, i.e., they export to the provider their internal routes and import from the provider default routes only.
- 200 iBGP sessions to internal routers, with 100 RIB-In and 130K RIB-Out, i.e., they export to the RCP statically configured routes and import from the RCP a full routing table.

Assume further that each BGP route stored in a RIB requires 200 Bytes of memory<sup>2</sup>. The total memory requirements for this RCS instance would then be 200 Bytes times 3.92 million RIB-In routes, plus 200 Bytes times 75.003 million RIB-Out routes, or 15.8 GB worth of memory. While not insignificant, these numbers

<sup>2</sup>This is the approximate amount of memory required per route by the Quagga open source BGP implementation [12]

are quite feasible with off-the-shelf server platforms. More importantly, however, the numbers illustrate that memory requirements for an RCS will be fairly significant and care has to be taken in an implementation to store routing information in an efficient manner. On the plus side though, the memory requirements of the RCS can be met by general purpose workstation memory as opposed to high performance memory that would be required for a router-based implementation.

In terms of bandwidth requirements for sending and receiving updates, the worst case scenario would involve an RCS losing all of its BGP sessions at the same time and then having to re-advertise all routes. This might happen for example if an RCS were naively deployed using a single connection to the network it is controlling. Again, such a deployment is clearly undesirable and a more realistic setting would be for the RCS to be deployed with redundant connectivity to the network. Never the less, we consider this worst case scenario to develop an understanding of the bandwidth requirements, again assuming our imaginary ISP setup described above.

Assume that each route update would translate into approximately 40 bytes of data transferred over the wire<sup>3</sup>. The total amount of RIB-In data for the described ISP would be 156.8 million bytes or 149.6 MB. Similarly the RIB-Out data would be 3000.12 million bytes or 2861.2 MB. Transferring this amount of data over a 155 Mbps OC3 link would take approximately 8 seconds for the RIB-In data and 164 seconds for the RIB-Out data. Even this worst case scenario therefore would not appear to present a significant demand on the system in terms of bandwidth requirements.

The inherent CPU demands of route selection is not very significant. Actual CPU demands largely depends the efficiency of the actual data structures used in an implementation. The different tradeoffs of an efficient RCS implementation is the subject of ongoing work.

## 4. Conclusions

In this paper we presented an overview of the Route Control Platform (RCP) architecture. RCP allows the route decision process for inter-domain routing to be removed from routers into a logically-centralized control function. This separation of functionality is desirable both for the flexibility that it enables in the routing decision process, as well as the simplification that results in the router configuration. While logically centralized the RCP by necessity has to be distributed for robustness reasons. The physical server that make up the RCP are called Routes Control Servers (RCSes). We presented the RCP architecture and discussed issues with its realization. We showed that the RCP is fully backwards compatible with the existing routing infrastructure, requiring only configuration changes in routers. We discussed the functionality that is required in a RCS and considered the scalability requirements for a typical medium sized ISP.

## 5. References

- [1] BATES, T., CHANDRA, R., AND CHEN, E. BGP Route Reflection - An Alternative to Full Mesh IBGP. Request For Comments 2796, April 2000.
- [2] CHANG, D.-F., GOVINDAN, R., AND HEIDEMANN, J. An empirical study of router response to large BGP routing table load. In *Proceedings of the ACM SIGCOMM Internet Measurement Workshop* (November 2002), pp. 203–208.
- [3] FEAMSTER, N., BALAKRISHNAN, H., REXFORD, J., SHAIKH, A., AND VAN DER MERWE, J. The Case for Separating Routing from Routers. FDNA workshop SIGCOMM'04, August 2004.

<sup>3</sup>Analysis of BGP traffic from a Tier-1 IP network over a three month period showed an average per-prefix size of 36 bytes.

- [4] FEAMSTER, N., BORKENHAGEN, J., AND REXFORD, J. Guidelines for interdomain traffic engineering. In *ACM Computer Communication Review* (October 2003).
- [5] FELDMANN, A., KONG, H., MAENNEL, O., AND TUDOR, A. Measuring bgp pass-through times. Passive and Active Measurement Workshop (PAM), April 2004.
- [6] GOODELL, G., AIELLO, W., GRIFFIN, T., IOANNIDIS, J., MCDANIEL, P., AND RUBIN, A. Working around bgp: An incremental approach to improving security and accuracy of interdomain routing. In *Proc. Network and Distributed Systems Security 2003, Internet Society* (February 2003).
- [7] GRIFFIN, T. G., AND WILFONG, G. On the correctness of IBGP configuration. In *Proc. ACM SIGCOMM* (August 2002).
- [8] MAHAJAN, R., WETHERALL, D., AND ANDERSON, T. Understanding BGP misconfiguration. In *Proc. ACM SIGCOMM* (August 2002).
- [9] PAI, V. S., DRUSCHEL, P., AND ZWAENEPOEL, W. Flash: An efficient and portable web server. In *Proceedings of the USENIX Annual Technical Conference* (1999).
- [10] REKHTER, Y., LI, T., AND HARES, S. A Border Gateway Protocol 4 (BGP-4). Internet Draft draft-ietf-idr-bgp4-22.txt, work in progress, October 2003.
- [11] SHAIKH, A., AND GREENBERG, A. OSPF Monitoring: Architecture, Design, and Deployment Experience. In *Proc. First Symposium on Networked Systems Design and Implementation (NSDI)* (San Francisco, CA, March 2004).
- [12] WWW.QUAGGA.NET. Quagga Routing Suit.  
<http://www.nanog.org/mtg-0302/ppt/van.pdf>.