The SoftRouter Architecture

T.V. Lakshman T. Nandagopal R. Ramjee K. Sabnani T. Woo
Bell Laboratories, Lucent Technologies
{lakshman, thyaga, ramjee, kks, woo}@lucent.com

ABSTRACT

In current routers, implementations of the control and forwarding functions are colocated and tightly integrated. In this paper, we present the SoftRouter architecture that separates the implementation of control plane functions from packet forwarding functions. In this architecture, all control plane functions are implemented on general purpose servers called the control elements (CE's) that could be multiple hops away from the forwarding elements (FEs). A network element (NE) or a router is formed using dynamic binding between the CEs and the FEs. We argue that this flexibility results in several benefits including increased reliability, increased scalability, increased security, ease of adding new functionality, and decreased cost.

1. INTRODUCTION

In current routers, implementations of the control and forwarding functions are intertwined deeply in many ways. The control processors implementing control plane functions are colocated with the line cards that implement forwarding functions and often share the same router backplane. Communication between the control processors and the forwarding line cards is not based on any standards-based mechanism, making it impossible to interchange control processors and forwarding elements from different suppliers. This also leads to a static binding between forwarding elements and line cards. A router typically has at most two controllers (live and stand-by) running control plane software. Each line card is statically bound to these two controllers. The two controllers, the line-cards to which they are statically bound, and the switch fabric together constitute the router.

In this paper, we propose a new control plane architecture called the SoftRouter architecture that separates the implementation of control plane functions from packet forwarding functions. In this architecture, all control plane functions are implemented on general purpose servers called the control elements (CEs) that could be multiple hops away from the line cards or forwarding elements (FEs). Thus, there is no need for a static association between the CEs and the FEs. Each FE, when it boots up, discovers a set of CEs that can control it. The FE dynamically binds itself to a "best" CE from the discovered set of CEs. In this architecture, a collection of FEs (along with their switch fabrics), together with their associated CEs, is called a Network Element (NE) and logically constitutes a router. We envisage a standardized interface between the CEs and the FEs similar to that being standardized in the IETF ForCES working group [12]. The SoftRouter architecture and the technical challenges introduced by this architecture are described in more detail in Sections 2 and 3, respectively.

Such an architecture has several benefits. By implementing the CE's on servers, the architecture permits easier scaling since the server capacity can be increased far more easily than increasing controller card capacities in routers. A server-based CE also facilitates use of stronger security mechanisms since such mechanisms have to be deployed at far fewer points in the network. A third aspect is the increased reliability possible both from a server-based implementation and from the architecture that permits each line card to have more than two possible controllers. A recent paper [4] argues that a server-based logically centralized implementation of BGP results in several benefits. By moving all control functionality out of the forwarding element, several other benefits, as described in Section 4, are now possible.

The proposed network evolution has similarities to the SoftSwitch based transformation of the voice network architecture that is currently taking place. The SoftSwitch architecture [11] was introduced to separate the voice transport path from the call control software. The SoftRouter architecture is aimed at providing an analogous migration in routed packet networks by separating the forwarding elements from the control elements. Similar to the SoftSwitch, the SoftRouter architecture reduces the complexity of adding new functionality into the network. We discuss this and other related work in Section 5 before presenting our conclusions in Section 6.

We now present an overview of the SoftRouter architecture.

2. ARCHITECTURE OVERVIEW

The SoftRouter architecture comprises of a number of different network entities and protocols between them. We describe them in separate subsections below.

2.1 Network Entities

A SoftRouter network can be described in two different views, namely, the *physical* view and the *routing* view.

In the physical view, a SoftRouter network is made up of *nodes* interconnected by *links*. There are two types of nodes: the *forwarding elements* (FEs) and the *control elements* (CEs). An FE is similar in construction to a traditional router; it may have multiple line cards (each in turn terminating multiple ports - physical or logical) and a backplane (switch fabric) that shuttles data traffic from one line card to another. Its key difference from a traditional router is the absence of sophisticated control logic (e.g., a routing process like OSPF) running locally. Instead the control logic is hosted at CEs, which are essentially general purpose server machines. A link connects any two elements (FEs/CEs). Typically, an FE has multiple incident links (so that data traffic can be routed from one link to another) and an CE is multi-homed to more than one FE (so that it is not disconnected from the network should its only link fail). In a nutshell, the physical view of a SoftRouter network is not that different from that of a traditional routed network,

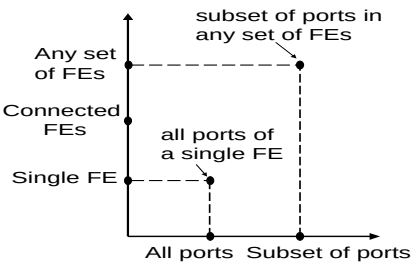

Figure 1: NE Possibilities

except with the addition of a few multi-homed servers (CEs).

The primary function of an FE is to "switch" data traffic between its links. The exact nature of the switching function can take different forms. We describe three possibilities here, among others: (1) Packet forwarding: this includes both layer 2 (MAC-based switching) and layer 3 (longest prefix match) forwarding. (2) Label switching: an example of this is MPLS forwarding. The data path forwarding function can include label swapping, pushing and popping. (3) Optical switching: the traffic in this case can be time-switched, wavelength-switched, or space-switched among the links. In each of these cases, the switching function is driven by a simple local table which is "computed" and "installed" by some CE on the network. In general, an FE can do more than switching. For example, an FE can perform security functions such as packet filtering and intrusion detection. The key requirement is that these functions should work off a local data structure whose management intelligence resides in some remote CE. For example, in the packet filtering case, the filtering logic consults only a local filter table, whose management (insertion and deletion of filtering rules) is performed in some remote CE.

The routing view of a network captures the topology of a network as seen by the routing control logic. To describe this view, we first need to define the concept of a *network element* (NE). At a high level, an NE is a logical grouping of network ports[1] and the respective CEs that manage those ports. By placing different restrictions on what network ports can be grouped together, the allowable NE varies. Specifically, there are two dimensions where restrictions can be placed (see Figure 1). The vertical dimension restricts the selection of FEs: *Single FE* means that the network ports in a NE must be from a single FE; *Connected FEs* means the ports can be selected from a set of FEs that are connected (i.e., there exists a physical path from one FE to another); and *Any FEs* means the ports can come from any set of FEs. The horizontal dimension specifies if an NE must include all ports of the FEs under consideration or if it can include only a subset. Clearly, any combination of the two dimensions provides a possible definition of the ports of an NE. We say that two NEs are *logically connected* if they contain ports belonging to FEs that are physically connected.

To illustrate these definitions better, we examine two specific cases, which represent the two extremes of the spectrum enabled by this definition.

[1] For obvious reasons, a port cannot belong to more than one NE.

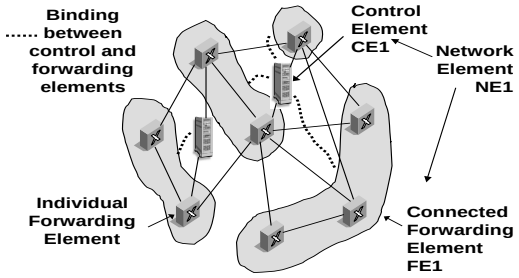

Figure 2: Example SoftRouter Network: All ports in Connected FEs

Subset of ports in any FEs: any ports from any FEs can be grouped as part of a single NE. The routing view can be significantly different from the physical view. This definition can be useful for VPN applications. Essentially, the different ports in an NE represent the different sites in a wide-area VPN. By logically collapsing them as part of a single NE in the routing view, this view provides a separation between the intra-VPN routing from the inter-VPN routing.

All ports in a single FE: each FE is part of one NE. The physical and the routing views become identical. With this definition, the only difference between a traditional routed network and a SoftRouter network is the offloading of the routing logic onto remote CEs.

Among all the possible definitions, the most interesting and practical case is the *All ports in Connected FEs* case. This represents the clustering of neighboring FEs into a single NE and corresponds to the typical case of several routers being connected back-to-back in a central office. From a routing perspective, this can provide significant simplification: (1) the reduced number of NEs in the routing view reduces the inter-NE routing complexity; and (2) a different (possibly less complex) routing protocol can be employed for intra-NE routing (more details are discussed in Section 3). For the balance of this paper, we focus mostly on this case.

In the SoftRouter model, the routing control of the NEs is disaggregated from the FEs and the control protocol runs on the CEs. A *binding* between an FE and a CE means that the CE is performing particular control functions on behalf of the FE. Because multiple protocols (e.g., IGP and EGP, or even multiple instances of a protocol) may be required for the operation of an FE, an FE may have multiple CE bindings.

An example SoftRouter network illustrating the different concepts we have introduced is shown in Figure 2. Note that a binding can exist between an FE and a CE even if they are not directly connected (e.g., CE1 and FE1 in Figure 2). It is worthy to point out that the SoftRouter architecture thus enables the creation of a separate (logically or physically) signaling network connecting all the CEs similar to the SS7 network in the PSTN.

2.2 Protocols

A number of different protocols are needed in the operation of a SoftRouter network. We describe the two most important ones here.

Discovery Protocol: In order for an FE to establish a

binding with a CE, it must first know about the existence of the CE and be able to reach it using some route. A discovery protocol finds out what CEs are available and lays out paths to them for the FEs. We describe this in greater details in Section 3.

FE/CE Control Protocol: Once a binding is established, the FEs and the CEs communicate using a control protocol. On the uplink (FE to CE) direction, this control protocol tunnels any control packets received in any of the ports in the FE (e.g., all OSPF protocol packets if CE is running OSPF for the FE) and provides link state information (e.g. link up/down signal) to the CE. On the downlink direction, the protocol carries configuration and control information (e.g., enable/disable a link, forwarding information base (FIB)). The key issues in the design of this protocol relate to the frequency, bandwidth and delay requirements for the FE/CE communication. In this paper, we assume that the ForCES [12] protocol would be sufficient for our purposes.

3. TECHNICAL CHALLENGES

In any given NE, the FEs are strongly connected (with the same underlying topology), and the CE can be many hops away from the corresponding forwarding set. This separation leads to many new scenarios and technical challenges that do not normally occur in existing networks.

In this section, we highlight some of these issues and present potential means of addressing them. We will use the term *integrated NE* to denote a router in an existing network, where the control element and the corresponding forwarding elements are part of a single physical device.

3.1 Bootstrapping

In an integrated NE, the configuration pertaining to that NE is obtained upon bootup, since the control card on the box is in direct contact with the forwarding engine through a *bus/interconnect*. As soon as the router (NE) comes up, the forwarding engine knows how to obtain the configuration information immediately from the control processor.

In our disaggregated model, upon bootup, the forwarding engine has to obtain its configuration information, which includes the IP addresses of its interfaces, from a remote control element that resides on a server. This poses a potential paradox: in order to discover a CE and send packets to it, the FE requires routing information; however, the routing information is supposed to come from the CE. One way to address this issue is to have a separate protocol that enables discovery of CEs by FEs and vice versa. The sole purpose of this protocol is to let FEs know of the available CEs in the network, and to ensure that a FE always has a path to reach its corresponding CE, if available.

The discovery protocol has to be extremely simple, since the FE is primarily a forwarding engine with minimal software required to update its FIB and to maintain association with its CE. Moreover, the FE may not even have its own layer 3 address. One possible solution is to assign a unique string as an identifier for an FE. While the control server knows its configuration upon bootup, for compatibility purposes it can also have a unique string identifier, in the same way as a FE. In an extreme case, one could even think of the string as a randomly chosen private IP address specific to the NE.

Any FE that is one hop away from a CE can discover the CE by listening to a status broadcast message from the CE. This FE can then propagate this information to its neighbors along with a source-route to the particular CE, with the source-route specified in terms of the randomized string identifiers. Thus, the reachability information for each CE can be propagated in a wave-like fashion to the entire NE. When a FE receives routes to a set of CEs, it can then choose one CE and use the advertised source-route to contact the CE with its local capability information. The CE will configure the FE with the relevant addressing and FIB database information that will enable the FE to begin packet forwarding.

It is important to note the assumptions in the above discovery protocol: (a) Any CE in a given NE is able to configure any FE within the NE; (b) The CE is always willing to handle the routing functionality on behalf of a given FE in the NE, regardless of system load and congestion level; (c) The CE is always available, in other words, the CE does not fail at all; (d) The source-route to a particular CE is available the entire time when the configuration of an FE by the CE is ongoing; (e) The CE trusts the FE and vice-versa; (f) Any discovery protocol packets can be overheard by nodes not within the NE, but are adjacent to it, without causing irreparable harm to the functioning of the network;

Relaxing each of these assumptions adds to the complexity of the discovery protocol, with more features and protocol messages needed to eliminate these assumptions. However, these are not fundamentally disabling assumptions, and they can be addressed while keeping the discovery protocol simple enough.

3.2 Routing and Forwarding

Since the FEs are assumed to be simple with minimal control on-board, the CE is responsible for maintaining the knowledge of the links' status between FEs within the same NE. In addition, the CE must also be able to integrate topology changes within the NE with external (inter-NE) route changes, and update the FIBs of individual FEs accordingly. This can be done by using a protocol similar to ForCES [12].

Due to the difference between the physical view and the routing view of the network as a result of this architecture, IP TTL and IP options behavior might deviate from traditional router design, which assumes geographical closeness of control and forwarding plane. In the SoftRouter architecture, the FEs of a given NE might be distributed over a large geographical area, which makes it expedient to decrement TTL and process IP options on a per FE (rather than a per NE) basis. For an NE with co-located FEs, one can similarly realize this behavior or imitate traditional router behavior.

Moreover, a NE might now have to select appropriate external routing metrics that suitably reflect the topology within the NE. The topology changes within the NE are caused not only by links going up or down, but also by FEs changing association to a different CE, and hence a different NE. Therefore, the hop count seen by the control plane differs from the one experienced by the data plane. The control plane sees hops on a per NE basis whereas the forwarding plane potentially sees hops on a per FE basis.

3.3 Protocol Partitioning and Optimization

In standard internet routing protocols, there are various messages that fulfil different functions. For example, OSPF

has *hello* packets to probe link status and various *LSA* messages to advertise link status to the rest of the network. In a SoftRouter architecture, these messages have to be differentiated according to their purpose. A FE can choose to either forward all received routing protocol packets to its CE, or it could handle a subset of the protocol packets by itself without forwarding it to the CE, thereby reducing response time and control traffic in the network.

Let us consider the OSPF routing protocol as an example. *HELLO* messages serve to discover the OSPF peering status of neighboring FEs. Instead of having the CEs originate the HELLO messages, we could allow the FEs to exchange these messages between themselves[2] and notify their respective CEs only when there is a change in the status of the FE peering status. On the other hand, LSA messages serve to advertise connectivity information to the rest of the network, and hence these will originate from the CE, and will be exchanged between CEs.

Within the SoftRouter architecture, routing protocols can be optimized in terms of message overhead. In a traditional router network running OSPF, for example, LSA messages from each FE are flooded over the entire network. In a SoftRouter network, LSAs are flooded only over the network of CEs. Given the potential difference in the magnitude of CEs and FEs, this reduces the number of OSPF messages sent over the network. We are investigating further optimizations for various other protocols enabled by the SoftRouter architecture.

3.4 Failures and Load Balancing

By separating the control elements, we provide more choices for CEs to control a given FE. This choice comes at the risk of vulnerability to loss of CE connectivity due to both intermediate link and node failures, in addition to failure of the CE itself. Aggregation of control elements aggravates the problem of CE failure since a large number of FEs will be left without a route controller.

We address this by using a standard server concept, that of backup CEs. Each FE will have a *primary CE*, and a *secondary CE set*. When the discovery protocol or the control protocol signals a loss of association with the primary CE, the FE can choose a candidate from the secondary CE set to become its primary CE. The re-association is done using the discovery protocol used for bootstrapping earlier.

Apart from CE failures, a CE can voluntarily relinquish control of a FE for load balancing purposes. In that scenario, the CE can ask the FE to find another CE to manage it, or the CE can guide the FE to a new CE. In the former, the new CE will have to restart the routing instance for the FE, while in the latter, the old CE can transfer the FE's state to the new CE, resulting in a hot fail-over. The ability to load balance and recover quickly from control failures is a salient feature of this architecture.

We would like to point out that when there is loss of association between a CE and a FE, the FE can continue to forward packets as long as there are no other topology changes in the NE, using its existing FIB. The discovery protocol will discover another CE in the meantime, thereby recovering from the loss of control.

The description of the failure recovery processes above are, in some sense, a generalization of the features being added to existing dual-processor routers by router vendors such as Cisco, and are referred variably as Stateful Switchover, Globally Resilient IP and NonStop Forwarding. The vendors are, however, limited to their choice of a single physical box as a NE, while our architecture allows us more flexibility in terms of the availability of CEs.

[2]at the cost of violating the strict semantics of the HELLO message

4. BENEFITS

There are five significant benefits to disaggregation:
1) Increased reliability: The reduced software in a forwarding element makes that element more robust. As regards the control element, protocol specific and independent mechanisms can be incorporated to enhance reliability;
2) Increased scalability: control elements can be implemented on general-purpose servers, and thus can be easily scaled up using well-established server scaling techniques;
3) Increased control plane security: fewer management points makes it easier to manage and provide a strong defense around the control elements, thus making the overall network more secure;
4) Ease of adding new functionality: adding new functionality is easier on a separate control server, executing on general purpose processors and operating systems;
5) Lower costs: SoftRouter decouples the innovation curve of the control and forwarding elements, allowing economies of scale to reduce cost.
We now discuss each of these benefits in more detail.

4.1 Reliability

The separation of control and forwarding elements in the SoftRouter architecture provides several reliability benefits. First, the reduced software in a forwarding element implies that it is easier to make that element robust. On the control plane server side, sophisticated reliability enhancing mechanisms such as automatic fail-over and the use of hot or cold standby's, as discussed in Section 3, can be incorporated. Further, a server can have much higher redundancy capabilities such as 1:N failover capability rather than a very limited 1:1 control blade failover capability of today's routers. Finally, the ability to host different protocols on different CEs imply that the failure of one CE does not render the corresponding FE useless (e.g. the failure of the CE hosting BGP would still allow the FE to process OSPF protocol messages). Apart from these generic benefits that result in higher availability, we detail protocol-specific optimizations with respect to two-key protocols, BGP and OSPF, below.

4.1.1 Increased BGP reliability

Figure 3(a) shows a typical deployment of BGP with Route Reflectors [2] in a large Autonomous System (AS) today. This deployment has two main drawbacks. Specifically, under certain conditions, the network can go into persistent route oscillation where a subset of routers may exchange routing information without ever reaching a stable routing state [1]. Another issue with the Route Reflector architecture is I-BGP reliability. While the failure of one I-BGP session will affect only two routers in the case of a full mesh I-BGP architecture, the same failure of a session between two Route Reflectors could partition the network, resulting in significantly lower reliability.

In the case of BGP deployment in the SoftRouter architecture (Figure 3(b)), the number of control elements that run BGP will typically be at least an order of magnitude smaller

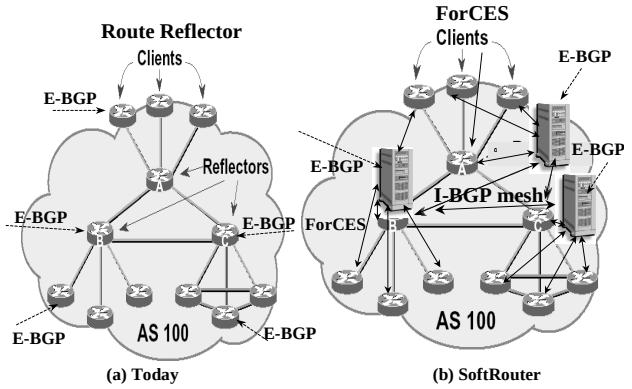

Figure 3: BGP deployment

than the number of routers. Thus, a full I-BGP mesh can easily be maintained among the control elements. The control elements would then download the appropriate forwarding tables to all the forwarding elements using the ForCES protocol [12]. Thus, the persistent route oscillation problem is trivially solved in the SoftRouter architecture (since there are no Route Reflectors), thereby increasing the availability of the network.

Further, in the SoftRouter architecture, the I-BGP mesh is between servers that can employ a higher degree of redundancy such as 1:N (N > 1) as compared to 1:1 redundancy on the control processors of the routers. Thus, BGP reliability in the SoftRouter architecture can be significantly improved over the current BGP deployments[3].

Note that the implementation of BGP in the SoftRouter architecture is logically similar to the router control platform (RCP) [4] proposal. The key differences are that a) unlike RCP, the IGP protocol is also executing in the server, b) we use the ForCES protocol to communicate with the forwarding elements rather than using IGP and IBGP as is the case in RCP (drastically simplifying the software on the forwarding element), and c) we use an IBGP mesh between the control element servers inside a single AS.

4.1.2 Faster OSPF convergence

OSPF convergence in the presence of failures is known to take tens of seconds in large networks today [6]. This delay can have a significant impact on the availability of the network, especially for critical services such as voice-over-IP. Compared to the 5 9's availability of the telephone network (translates to 3 minutes of downtime a year), today's data network can ill afford more than a couple of link failures before its availability falls below the 99.999% availability target. SoftRouter architecture allows for faster OSPF convergence through several complementary techniques:
(a) Order of magnitude fewer control elements for the same number of forwarding elements, resulting in a smaller OSPF control network and faster convergence[4]
(b) Control plane optimizations specific to the SoftRouter architecture (detailed earlier), allowing faster convergence
(c) Faster processors (see section 4.2) result in faster computations (shortest-path calculations)

[3]assuming that the CEs are multi-homed and there are multiple CEs in the network for fail-over.

[4]assuming that propagation delay from the FEs to the CEs is much smaller than the propagation delay of OSPF updates, which are governed by timers on the order of seconds.

Thus, OSPF convergence time can be significantly reduced from tens of seconds [6], resulting in a highly available data network.

4.2 Scalability

Some of the fundamental limitations in scaling routing protocols in existing architectures are control processor capacity and on-board memory. The decoupling provided by the SoftRouter architecture makes it easier to upgrade control hardware which is based on general-purpose servers; contrast this to the difficulty in obtaining an upgraded control processor card from a router manufacturer who needs to accommodate upgrades with other constraints such as power availability, slot availability, etc..

We now highlight this advantage through a specific example of requirements of a highly scalable Mobile IP home agent [8]. Mobile IP home agent will require increasing scalability as cellular carriers such as Verizon and Sprint introduce wireless data. There are two approaches to home agent scalability in the industry today: using only routers and using only general purpose processors. However, both of these approaches have limitations as discussed below.

Routers from major router vendors support several hundred thousand home agents but signaling scalability is limited due to limitations of the control processor to about hundred bindings/sec (or less than two updates per hour per user). This is a significant limitation as updates generated through both mobility as well as the periodic refresh mechanism can easily exceed two per hour per user. Mobile IP could also be implemented on a cluster of general purpose processors. Signaling scalability will not be an issue here. However, scaling the number of home agents becomes difficult since IPSec processing (for each agent) is CPU intensive and will not scale efficiently to several hundred thousand home agents without specialized hardware.

SoftRouter architecture admits a complementary combination of both of these approaches. It allows server-based signaling scalability while retaining hardware-based transport scalability. Thus, transport will still handled by FEs with hardware support for IPsec using regular router blades while signaling capacity can be easily scaled up using multiple server blades.

4.3 Security

The SoftRouter architecture enables the adoption of a multi-fence approach to security with each fence adding a layer of security. These include:
(a) Off-the-shelf versus special-purpose operating system: specialized operating systems are not as widely-tested for security holes;
(b) Multi-blade server platform versus one or two control blades in the router: overload due to malicious traffic can be distributed over a large number of processors and sophisticated compute-intensive intrusive detection mechanisms can be deployed;
(c) Fewer control elements: managing fewer elements is easier (e.g. changing security keys frequently) and it may be possible to place these few elements in a more secure environment (physically or logically firewalled) compared to the numerous forwarding elements;
(d) Separate signaling network: using a physically or logically separate signaling network for the Internet, similar to SS7, can limit attacks on control plane protocol messages.

4.4 New Functionality

The separation of control and forwarding elements and the use of a few general purpose servers to host the control processes enable easier introduction of new network-based functionalities such as quality of service support, traffic engineering, network-based virtual private network (VPN) support etc. We now discuss the benefits of the SoftRouter architecture in supporting network-based VPNs.

There has been a lot of recent activity in the IETF in defining network-based VPN services using BGP/MPLS [9]. In this application, a VPN server dynamically creates MPLS or IPSEC tunnels among the provider edge routers. While the VPN server would execute on the router control board in today's architecture, migrating the VPN server functionality into a control element in the SoftRouter architecture has several benefits: 1) VPN server upgrades can now be independently performed without impacting basic network operations such as forwarding; 2) Network-wide failover of VPN control servers can be performed without impacting existing or new VPN sessions; 3) Configuring BGP policies for the provider edge routers connected to the VPN customer sites can be done in a central location at the VPN server rather than at multiple routers (e.g. edge routers and route reflectors involved in the VPN); 4) MPLS tunnels can be engineered in a centralized manner to meet customer requirements; 5) Scalability for support of large number of VPNs can be easily handled using generic server scaling techniques.

4.5 Cost

In the disaggregated model, the forwarding element is mostly hardware-based and requires little management. This allows economies of scale to help reduce the cost of each forwarding element. The individual router control element is consolidated into a few dedicated control plane servers. These control elements will run on generic computing servers rather than in specialized control processor cards in routers as is the case today. This allows "complete sharing" of control processor resources resulting in better efficiency than the "complete partitioning" approach adopted today. Further, since the control plane servers are just another application for generic computing servers, the SoftRouter architecture can leverage the CPU price-performance curve of these server platforms. Finally, the reduced number of control plane elements means fewer boxes to manage, thus reducing operating expenses.

5. RELATED WORK

The SoftSwitch architecture was introduced in the late 1990's in the telecom world [11] to separate voice transport from call control. As mentioned earlier, the SoftRouter architecture attempts to provide an analogous migration in the routed packet network by separating the forwarding elements from the control elements.

The Internet Engineering Task Force (IETF) is working on standardizing a protocol between the control element and the forwarding element in the ForCES [12] working group. However, unlike the SoftRouter architecture, the focus is on a architecture where the control element is directly connected (single IP hop) to the forwarding element.

The case for separating some of the routing protocols (specifically, BGP) multiple hops away from the routers have been made by several researchers [4, 5]. While it is possible to migrate a few selected protocols out of the forwarding element, such an approach does not deliver the full benefits of the SoftRouter architecture.

One of the key benefits of the SoftRouter architecture is that it makes it easier to add new functionality into the network. Researchers have proposed other techniques such as Active Routers [10] or open programmable routers [7] to increase flexibility in deploying new protocols in the Internet. By separating the forwarding and control elements and hosting the control protocols on general purpose servers, a lot more resources are available for adding new software services in the SoftRouter architecture.

Finally, as the tussle between the service providers and the end users occurs in the Internet [3], an architecture like the SoftRouter may be necessary for allowing flexible solutions, that may be resource-intensive, to be deployable.

6. CONCLUSIONS

In this paper we presented the SoftRouter architecture that separated the implementation of control plane functions from packet forwarding functions. The forwarding elements were simple hardware devices with little intelligence and were controlled by control elements that might be multiple hops away. We then highlighted both the flexibility accommodated by the partitioned view as well as technical challenges unique to this architecture. Further, we argued that the SoftRouter architecture provides significant benefits including increased reliability, increased scalability, increased security, ease of adding new functionality and decreased cost. As data networks become integral to everyday life and as voice-over-IP services become increasingly deployed on data networks, these issues will become critical necessities. The SoftRouter architecture is well placed to meet these critical requirements.

7. REFERENCES

[1] A. Basu et. al., "Route Oscillations in I-BGP with Route Reflection," In Proc. of ACM SIGCOMM, Aug. 2002.

[2] T. Bates, R. Chandra and E. Chen, "BGP Route Reflection - An Alternative to Full Mesh IBGP," RFC 2796, April 2000.

[3] D. Clark, J. Wroclawski, K. Sollins, and R. Braden, "Tussle in Cyberspace: Defining Tomorrow's Internet," In *Proc. ACM SICOMM Conference*, August 2002.

[4] N. Feamster, H. Balakrishnan, J. Rexford, A. Shaikh, and J. van der Merwe, "The Case for Separating Routing from Routers," In *Proc. of FDNA workshop*, August 2004.

[5] R. Govindan, C. Alaettinoglu, K. Varadhan and D. Estrin, "Route servers for inter-domain routing," em Computer Networks and ISDN systems, Vol. 30, 1998, pp 1157-1174.

[6] M. Goyal, K. Ramakrishnan and W. Feng, "Achieving Faster Failure Detection in OSPF Networks," In *Proc. of ICC*, 2003.

[7] M. Handley, O. Hudson, and E. Kohler, "XORP: An open platform for network research," In *HotNets*, 2002.

[8] C. Perkins, "IP Mobility Support for IPv4," RFC3344, August 2002.

[9] E. Rosen and Y. Rekhter, "BGP/MPLS VPNs," RFC2547, March 1999.

[10] D. Wetherall, U. Legedza and J. Guttag, "Introducing New Internet Services: Why and How," IEEE Network Magazine, July/August 1998.

[11] S. Williams, "The softswitch advantage," *IEE Review* Volume: 48 , Issue: 4 , July 2002, Pages:25 - 29

[12] L. Yang et al., "Forwarding and Control Element Separation (ForCES) Framework," RFC 3746, 2004.