MPLS and the Evolving Internet Architecture

Tony Li, Procket Networks, Inc.

ABSTRACT The Internet architecture has evolved over time, adapting to the needs of its users and incorporating new technology as it has been developed. The introduction of MPLS as a part of the Internet forwarding architecture has immediate applications in traffic engineering and virtual private networks. In the longer term, MPLS may affect how traffic transits the Internet and the services that the Internet delivers.

Since the initial deployment of the original ARPAnet, the forerunner to the Internet, the architecture of the network has been changing. It has evolved in response to technological progress, experimentation, explosive growth, and new services. The most recent change to the Internet architecture is the addition of multiprotocol label switching (MPLS).

MPLS has an impact on both the mechanisms used to forward packets within the Internet and the routing protocols used to determine the path that packets should take while transiting the Internet. Because these changes are happening in the forwarding and routing functions of the network, they are altering the fundamental architecture of the Internet. These architectural changes are enabling new services based on the existing Internet infrastructure.

MPLS is being deployed because it has an immediate and direct benefit to the network. However, the long-term impact of MPLS is difficult to anticipate because of the innovations it enables. This is, in part, one of the indicators of an architectural change, as opposed to an evolutionary change, which would have clear, immediate, and foreseeable consequences.

Components of the Internet Architecture

The architecture of the Internet can be broken down into four basic levels:

- The physical level
- The link layer
- The network layer
- The transport layer

The International Organization for Standardization (ISO) seven-layer reference model includes these four layers, plus three more on top of the transport layer.

In this model, connection-oriented protocols such as ATM typically appear as an independent layer between the link layer and the network layer, and are sometimes known as *layer 2.5 protocols*. MPLS is another connection mechanism that appears in this layer, similar to asynchronous transfer mode (ATM) or frame relay, but without a dependence on the link layer.

In the Internet architecture, the network layer is composed of the Internet Protocol (IP) itself, the forwarding mechanisms, the routing mechanisms, and the low-level control functions found in the Internet Control Message Protocol (ICMP). IP defines a packet format that contains a destination address, where each address is unique within the entire network and contains sufficient information for the network layer to deliver the packet to the appropriate destination.

Within the network layer, the routing mechanisms are responsible for computing paths to each node in the Internet. Obvious requirements of the routing mechanisms are knowledge of the topology, knowledge of and compliance with the differing policies of various providers, and scalability. The results of the routing mechanisms are contained in a forwarding table that includes the next hop to take to reach each possible destination.

To ensure that the forwarding table remains scalable, each individual forwarding table entry must be used for a large number of destination addresses. Forwarding table entries for destinations close to the node will be used for a smaller number of destinations, while entries for distant destinations must be used for a larger number of destinations. This implies that there is a need for more specific routing knowledge in the immediate region, and more abstract information for remote destinations.

The forwarding mechanisms are responsible for taking the results of the routing component and switching the IP packets toward the final destination. The traditional forwarding architecture is based on hop-by-hop forwarding, in which each node in the network independently computes its own table for forwarding, and each packet is switched based on its destination address at each hop. One of the important implications of this architecture is that the routing component must provide consistent results at each node in the network; otherwise, packets can be switched incorrectly, resulting in grossly suboptimal paths or forwarding loops. Thus, providing routing that is largely consistent is also a strong requirement for the routing mechanisms.

Hop-by-hop destination-based forwarding also acts as a constraint on the services the network can provide. For example, it implies that packets with the same destination that traverse the same point in the network will follow the same path from that point. This makes it difficult for the architecture to support other services, such as providing paths that are specific to particular sources, to particular services at the destination, or to a particular class of service.

The Evolution of the Internet Routing Architecture

The mechanisms for providing routing functionality within the Internet have evolved over time. Prior to what we now know as the Internet, the ARPAnet was structured as a single flat link layer, based on packet switches known as *interface message processors* (IMPs). All IMPs were given a unique number, and each port on an IMP was also numbered, so a link layer address was simply an IMP number and a port number. To tie IP addressing to the link layer addressing, each IP address included the link layer address of the host. This meant that all routing functionality could be performed at the link layer, and routing protocols ran between the IMPs to perform path computation.

This sufficed while the ARPAnet (and replications of it) was a single link layer, but with the rise of other link layers, such as Ethernet, the situation became more challenging, requiring a protocol such as IP that could traverse different link layers. Passing packets between such link layers required a system that was attached to both of the link layers. Such a system was called a *gateway* but is now better known as a router.

THE CORE GATEWAY ARCHITECTURE

A number of gateways were established within the ARPAnet to interconnect with other IP networks. In this architecture, the gateways had complete knowledge of all connected IP networks, and they would route packets between the various networks. The gateways used the Gateway-to-Gateway Protocol [1] to exchange routing information with other gateways.

This evolved into a two-level hierarchical routing architecture, where certain gateways were designated as core gateways and acted as the central repositories and distribution mechanism for routing information. The remaining gateways in the network were known as exterior gateways. The Exterior Gateway Protocol (EGP) [2] relayed reachability information between the core and exterior gateways. EGP allowed exterior gateways to exchange traffic directly with other exterior gateways, while routing information followed a hub-and-spoke topology. This improved the scalability of the routing architecture. Each site (known as an autonomous system or a domain) was responsible for routing within the site, based on an Interior Gateway Protocol (IGP).

This early routing architecture is remarkable in several respects: it mandated a single set of centralized, authoritative gateways that were topologically constrained to exist only in the logical center of the network. Further, the topology was effectively restricted to a hub-and-spoke design because EGP only exchanged reachability information and did not include any mechanism to preclude loops in the information flow. If such a loop did form in the information flow, it could have easily created a forwarding loop, causing packets to go around in circles.

During this period, the most common IGP was the Routing Information Protocol [3] which was implemented in Berkeley Unix and quickly propagated to many domains. It sufficed because most organizations had relatively small networks, and the overhead of the protocol was not onerous.

THE CREATION OF NSFNET

With the creation of the NSFnet backbone and NSFnet regional networks, and the gradual transition of organizations off of the ARPAnet, the situation became much more chaotic. The NSFnet existed in parallel with the ARPAnet, and the core gateways were no longer the only useful source of routing information. Engineers pressed EGP and IGPs into service to provide interdomain routing and used manual filtering extensively as a check against incorrect information. This was an awkward solution that persisted until EGP was replaced by the Border Gateway Protocol (BGP) [4].

BGP had several advantages over EGP. It scaled better than EGP because it dispensed with periodic updates, relying instead on a TCP connection for reliable delivery. BGP also dispensed with the topological restrictions imposed by EGP, allowing domains to be interconnected in a flexible way, while continuing to ensure loop-free routing through a simple, explicit mechanism for loop detection. Further, BGP included policy mechanisms, allowing interdomain policy information to be expressed directly within the protocol, instead of relying on manual functions outside of the protocol. In a very short time, BGP became the de facto interdomain routing protocol for the Internet.

CLASSLESS INTERDOMAIN ROUTING

Shortly after the initial deployment of BGP, it became apparent that the routing architecture was under stress from another direction: the amount of data necessary to support the routing architecture was growing rapidly and showed no signs of slowing. This growth was a direct result of the address allocation mechanisms that were originally deployed for the ARPAnet, in which each site got a network number, and that network number needed to be in the routing tables in the logical core of the network. This implied that the routing tables grew with the number of organizations attached to the network, which was fine as long as the community remained limited. However, the increasing popularity of the Internet meant that this growth could not be sustained for long. To meet this problem, the routing and addressing architectures of the Internet were again changed. The new architecture is known as *classless interdomain routing* (CIDR). The major change of this architecture was to replace the original two-level hierarchical address with a generalized, multilevel hierarchical address and to generalize BGP so that it could carry the new, hierarchically assigned addresses.

ALTERNATIVE ARCHITECTURES

Architectural work after CIDR has mostly been on two alternative architectures known as Nimrod [5] and the unified routing architecture [6]. Nimrod is based on a hierarchical map-based routing paradigm, normal datagram forwarding, and aggregated, explicitly routed flows. MPLS label-switched paths (LSPs) are in some respects very similar to these aggregated, explicitly routed flows, and many of the techniques described in Nimrod, such as local repair of an aggregated flow, are directly applicable to an LSP.

Similarly, the unified routing architecture posits the use of BGP for normal best-effort traffic, with more specialized explicit paths for special services and applications. Initially, the unified architecture used a tunneling mechanism with an explicit path in every packet as the mechanism for supporting these special services. However, the overhead of carrying around the explicit path in each packet was deemed prohibitive. The obvious alternative is the encoding of the explicit path in state information that is propagated along the forwarding path. This is exactly the approach that has been taken by MPLS, as a direct result of the work done on the unified architecture.

Essentially, the Internet architecture has been evolving since the inception of the ARPAnet, and it appears that it will continue to evolve as our understanding of networking continues to grow. Because the problems of complexity and scalability are at their worst in the public Internet, it's likely that the Internet will continue to drive changes in its own architecture, both to address tactical problems and to aid in the ongoing deployment of new services across the Internet. All of this has happened, and will continue to happen in the midst of a fully functional, operational Internet.

CONNECTION-ORIENTED ARCHITECTURES

In parallel with the evolution of the Internet, other types of networks have continued to evolve. The most popular alternative to a datagram network is known as a connection-oriented network, because a logical connection must exist between endpoints in the network before data can be exchanged. The Internet, in contrast, is considered a datagram network because packets can be sent before any connection is created. Examples of connection-oriented link layers are ATM, frame relay, and X.25. These connection-oriented architectures are attractive because they require control information (i.e., state) at each network element along the connection, and this state can enable services that are simply intractable within a pure datagram network.

For example, certain quality of service features are much easier to support because all data traveling along a specific connection can be treated similarly, and no special analysis needs to be done on each data packet. More important, in a connection-oriented network, each connection can be delivered along a unique path through the network. For example, two packets at one point in the network, bound for the same destination but on different connections, would flow down two different paths to the destination. This contrasts with the hop-by-hop forwarding in a datagram network that forces packets for the same destination to follow the same path.

The ability to specify an explicit path for a particular connection is useful in traffic engineering the network. It allows the network engineer to shift traffic from congested links to uncongested links in the topology, thereby allowing the entire network to run more efficiently. The ability to set up explicit paths through the network is also useful as a policy tool. For example, a policy may require that certain traffic travel across particular transit carriers, based on either service or security requirements. Links can also be dedicated to a particular class of traffic, simplifying provisioning for complex classes of service.

Connection-oriented services also enable high-speed service restoration for loss-sensitive services. In the connection-oriented model, traffic can be redirected from a broken connection to other alternate connections to the same destination in a very rapid fashion, very close to the location of a failure. This is in contrast to the restoration times necessary in a datagram network, where the routing protocol must converge before service will be restored.

THE ROLE OF MPLS IN THE INTERNET ARCHITECTURE

Connection-oriented networks often serve as the infrastructure for datagram networks. Numerous IP over ATM networks demonstrate that this is a practical and useful approach. However, the separation between the connection-oriented layers and the datagram layer makes it difficult to harness the benefits of the connection-oriented network.

In MPLS, the approach is different. With MPLS a virtual connection is established between two points on a pure datagram network, and the connection in turn carries datagram traffic. The MPLS connection is the LSP. By using LSPs in a manner similar to a connection-oriented network, MPLS can provide many of the same advantages of a connection-oriented network while still retaining the underlying efficiency and operation of a datagram network. MPLS can be used to fully emulate a connection-oriented network if that is appropriate, or in a hybrid architecture where LSPs are used to deliver only connection-oriented services, but normal datagram mechanisms are used to deliver conventional datagram services.

The hybrid architecture is advantageous over full connection-oriented emulation because there is a small efficiency hit and management overhead inherent with connection emulation. By only incurring this overhead for services that require the connection-oriented services, the cost of running the network is minimized.

TRAFFIC ENGINEERING

The most immediate benefit of MPLS is the ability to perform traffic engineering. Traditionally, datagram networks have exhibited poor efficiency because the only mechanism for redirecting traffic has been to change the link metrics presented to the IGP. This mechanism is awkward because changing the metric for a link can potentially change the path of all packets traversing the link. MPLS provides better granularity when making this type of change because any particular LSP can be shifted from a congested path to an alternate path. This represents an efficiency improvement over the traditional operational methods for datagram networks because the network operator can run the network at much higher capacity under normal circumstances, secure in the knowledge that before congestion does occur, some of the traffic can easily be shifted away from the congestion point. Further, the operator can make use of global optimization algorithms that provide a mapping from the traffic demand to the physical links that could not otherwise be achieved using only local optimization. The net result is that the operator can achieve a much higher degree of link utilization throughout the network, thereby providing services for lower costs.

ROUTE PINNING

Another requirement that can be addressed by MPLS is the need for a specific and stable path through the network, also known as a route that has been *pinned*. Traditional Internet routing will compute a path to a destination and use it, sometimes in preference to a path that already existed and operated within requirements. This may not be appropriate for some applications, where certain traffic characteristics may be sensitive to a change in path. For example, some applications are highly sensitive to changes in latency, and an improvement in a path that decreases the latency of the connection may be as disruptive as an increase in latency.

One mechanism to address this need is to transport this application traffic on top of an LSP. Because the path of an LSP does not change from the time it is established until the time it is disconnected without explicit intervention, changes in the routing infrastructure will not make unwelcome changes to the forwarding path of the LSP.

VIRTUAL CIRCUIT EMULATION

Another application of MPLS is to emulate other connection-oriented networks. For example, an existing frame relay network could be migrated to an MPLS base by using an MPLS LSP to emulate a frame relay data link connection identifier (DLCI). This would require translation at two points in the network, but would allow a transparent change in the underlying service without changes in deployed edge equipment. More generally, because MPLS is a connection-oriented mechanism, it's possible to emulate any other unreliable connection-oriented service by emulating a virtual circuit with an LSP.

The advantages of this are clear. A single integrated datagram network can provide legacy services such as frame relay and ATM to end customers while using only a single infrastructure. Further, the network now operates with the traffic level that is the aggregate of each individual service, thereby allowing the network to take advantage of the economies of scale available at higher traffic levels.

VIRTUAL PRIVATE NETWORKS

One service currently delivered using a connection-oriented network is a virtual private network (VPN). Such networks are useful in providing the internal network to a distributed organizàtion. A typical example is the interconnection of several remote field offices with a corporate headquarters. Such a network may not have Internet access and has stringent privacy requirements on its traffic. This application is frequently addressed today using frame relay.

In an MPLS network, a VPN service could be delivered in a variety of ways. One way would be direct emulation of frame relay, as described above. Another approach would be to deliver the service using MPLS-aware subscriber equipment. Either approach allows a service provider to deliver this popular service in an integrated manner on the same infrastructure they use to provide Internet services.

Circuit Protection

An essential service in most networks is the ability to recover from circuit failures. In datagram networks, the routing architecture addresses this by having the routing protocols compute alternative paths. In circuit-based networks, this function is provided by the SONET/SDH infrastructure. In virtual-circuit-based networks, protection may be provided by both SONET and routing protocols within the connection-oriented architecture.

Fast Reroute

MPLS can be used in either of these ways. First, as a longer latency protection mechanism, replacement LSPs can be established around points of failure. Typically, this will have a restoration time similar to the time it would take for a routing protocol to converge. However, MPLS can also be used in a manner very similar to synchronous optical network (SONET), where no signaling is necessary to perform the protection switching of the LSP. This results in restoration times competitive with SONET. Such restoration times are not of interest to most datagram traffic, but for real-time services that cannot tolerate a brief outage, such as voice, these restoration times may be a necessity. This ability in a pure datagram network, without the expense of SONET equipment, is very attractive in some circumstances.

Hierarchical Forwarding

Perhaps the most profound change MPLS makes to the Internet architecture is not to the routing architecture, but to the forwarding architecture. As described above, the original hop-by-hop forwarding architecture has remained unchanged since the very early days of the architecture. To be sure, some connection-oriented link layers have been based on a different forwarding architecture, but these have been link-layer-specific and have not offered the possibility of a true end-to-end change in the overall forwarding architecture.

An interesting part of the MPLS forwarding architecture that may have significant impact is the ability to provide hierarchical forwarding. That is, the actual forwarding of an LSP may involve it being encapsulated within another LSP. This type of architectural feature is not wholly unheard of. ATM provides a two-level hierarchy with the notion of a virtual path and a virtual circuit. MPLS goes farther and generalizes the situation, allowing LSPs to be arbitrarily nested.

It remains to be seen how this feature will be actually used when deployed, but one obvious application for this is the ability to create wholly transparent and independently routed MPLS networks that provide transit service for any network layer protocol, including IP and MPLS itself. The benefit of this is that the transit provider need not carry global routing information, thus making the MPLS network more stable and scalable than a full-blown routed network.

In contrast to ATM, this same approach can be taken at many levels, and one MPLS network can be run over another, so the forwarding hierarchy can continue to scale to very large MPLS networks, each constructed from smaller MPLS networks. Such an architecture would not eliminate the top-level routed IP network, but would ensure that routing only occurred at nodes in the network where it was a necessity, such as at the customer entrance, and not at unnecessary points in the network.

An important point about this forwarding hierarchy is that there need not be a loss of granularity when merging LSPs into other transit LSPs. Because an LSP can encapsulate other LSPs without removing the original identification of the most granular LSP, many dissimilar LSPs can be encapsulated together, carried across a transit network, and then separated back into individual LSPs. This ability is important because the granularity is still present at the edges, where it is relevant, but is hidden at the transit core, where it could impede scalability. The utility of this is clearly visible in circuit-based networks, where SONET carriers are commonly used to encapsulate multiple circuits which are then delivered separately at the destination.

Conclusion

The deployment of MPLS in the Internet has some profound consequences at the architectural level. It changes the basic forwarding model, which has remained essentially unchanged since initial deployment of the ARPAnet. In turn, it also impacts the routing architecture, requiring that routing protocols perform new and more complex routing tasks. While the initial deployment of MPLS will primarily provide benefits for traffic engineering, the many applications of MPLS within carrier networks could provide many benefits in the years to come.

The exact ramifications of MPLS deployment are as yet unclear, but some of the immediate consequences are a more efficient transit core network, improved economy of scale, new connection-oriented services, and the ability for fast restoration of data traffic. Immediate applications are likely to be in intradomain traffic, where a single network administrator drives the MPLS deployment. Over time, however, interdomain MPLS deployment is likely to occur, where transit carriers provide MPLS service to local ISPs and to national ISPs. This allows for a beneficial separation of services: long haul bandwidth providers and their customers who provide long-haul Internet transit. The bandwidth provider is likely to be facilities-based and to bring raw MPLS bandwidth to market, where it is consumed by an engineering function that applies Internet routing to the available bandwidth. In large carriers, these functions are likely to coexist as separate groups and are analogous to the current separation of the transport group from the network group found in today's voice carriers.

The deployment of MPLS in the Internet is only possible because it is wholly transparent to the end user, and is beneficial because it changes the entire landscape for future developments in the Internet routing and forwarding architectures. The Internet architecture has been evolving rapidly for quite some time, and by providing a new basis for future architectural development, MPLS will in turn enable other new and interesting protocols and services. Some of these can be anticipated; others may only become more apparent over time.

References

[1] R. Hinden and A. Sheltzer, "The DARPA Internet Gateway," RFC 823, http://www.ietf.org/rfc/rfc0823.txt

[2] D. Mills, "Exterior Gateway Protocol Formal Specification," RFC 904, http://www.ietf.org/rfc/rfc0904.txt

[3] C. Hedrick, "Routing Information Protocol," RFC 1058, http://www.ietf.org/rfc/rfc1058.txt

[4] Y. Rehkter and T. Li, "A Border Gateway Protocol 4 (BGP-4)," RFC 1771, http://www.ietf.org/rfc/rfc1771.txt

[5] I. Castineyra, N. Chiappa, and M. Steenstrup, "The Nimrod Routing Architecture," RFC 1992, http://www.ietf.org/rfc/rfc1992.txt

[6] D. Estrin, Y. Rekhter, and S. Hotz, "A Unified Approach to Inter-Domain Routing," RFC 1322, http://www.ietf.org/rfc/rfc1322.txt

Biography

Tony Li (tli@procket.com) is a co-founder of Procket Networks. Previously, he was a Distinguished Engineer and project lead at Juniper Networks, where he led the design and development of MPLS. Prior to joining Juniper, Tony was a Technical Lead at Cisco Systems. He has contributed to multiple Internet backbone router architectures and works actively in Internet routing.