

Deploying MPLS & DiffServ

Thomas Telkamp

Director, Data Architecture & Technology
Global Crossing Telecommunications, Inc.
telkamp@gblx.net

MPLS and DiffServ technologies are getting a lot of attention these days because of their ability to support Quality of Service (QoS) and Service Level Agreements (SLA) on IP networks. This article will address some of the issues related to the deployment of MPLS and DiffServ in an IP backbone network, and evaluate the effectiveness of meeting the requirements of applications using the network.

Network Requirements

Before we can talk about network technologies we need to understand what kind of applications will make use of the network, and what requirements we need to meet. A lot of existing IP networks are mainly carrying Internet traffic. Traditionally this has always been dealt with as 'Best Effort' traffic, meaning that there are no guarantees on packet loss, delay or jitter in the network. Because an increasing amount of people depend on the Internet for their work or private life, the demand for high quality Internet services has increased. Internet Service Providers (ISP) are now providing SLA's to their customers, specifying what quality they intend to offer in terms of availability, packet loss and other parameters. This is the first kind of change in requirements for the network from the plain 'Best Effort' service.

By deployment of Virtual Private Networks (VPN) on an IP backbone, the network operator can offer services it controls end-to-end. In the Internet this is normally not possible because most traffic originates or terminates on a different ISP's network, and the quality is not fully controlled by one ISP alone. Companies can use these VPNs to connect their offices, data centers, or even to facilitate transactions between them and other companies (IP Extranet). Many of the applications running on this service are business critical to these companies (e.g. financial transactions and trading), and require a very high availability. The network operator will not only need to offer a SLA meeting the requirements to these companies, but also design/engineer the network to provide the needed quality.

Voice over IP (VoIP) and video applications generate even higher requirements for the network. They do not only need a high availability and the lowest delay, but also require a low and bounded jitter (variance in delay). The interactive

nature of these applications results in a direct service quality impact if any of the network requirements are not met.

Of course these different quality offerings also allow the network operator to price the services differently from each other. This way the operator can make more money off its value added services, and thereby increasing the revenue and margins.

Network Philosophy

Now we know the requirements we can establish some general rules for the network architecture. It may sound very obvious (and it is!) but good network design is the key to providing good service. In a network that has a redundant architecture and sufficient capacity, all the services/applications will be of high quality. A lot of services are billed on the actual usage, and therefore more transported packets means more revenue for the operator. The network should not have any bottlenecks in normal condition, in other words it is overprovisioned. In more detail this means that with the use of Traffic Engineering the network can handle all traffic even if the most critical link fails.

The key to QoS is to maximize the amount of time the network runs in this optimal condition. Factors contributing to this are redundancy (dual equipment), stable routing design, regular configuration audits, mechanisms to deal with Denial of Service (DoS) attacks, etc. In order to provide QoS, it is better to spend resources and effort on these factors, than on the various mechanisms described later that deal with failure scenarios. The deployment of too complex and too many features will make the network unreliable and unstable.

Unfortunately networks don't always behave the way we would like. Links fail, routers crash and configuration mistakes happen. In these kind of situations the performance of the network will not be optimal, and this is the point we need to start differentiating between the various services/applications.

The main issue now is how many 'levels' or 'classes' of service are needed. The following two criteria can be used to validate the number of classes:

- 1) is there a targeted application for each class?
- 2) can end users distinguish between classes?

An example of classes based on the requirements described above is:

- 1) Best Effort Service
 - Application: Internet
- 2) Assured Service
 - Application: trading and non-interactive audio and video
- 3) Real-time Service

- Application: interactive voice

It's obvious in this case that each class has a clear application, but the answer to the second question really depends on the network's architecture and implementation. The next few paragraphs will describe how to distinguish these classes, by using MPLS and DiffServ.

MPLS

Multi Protocol Label Switching (MPLS) is a label swapping technology to forward data packets in a network. A short, fixed-length label is attached to every packet entering the network, and that label will then define the path a packet is following through the network. This path is predefined, and can be set up matching certain constraints, such as a minimum amount of bandwidth, or maximum delay. In 'traditional' IP networks every router makes an independent decision on the forwarding of a packet (normally based on the shortest path), with MPLS the first router in the network decides on the entire path through the network. The deployment of MPLS enables an operator to use Traffic Engineering, Fast Reroute on its backbone, and also to use MPLS to build VPNs.

The first major application of MPLS was Traffic Engineering (TE). In traditional IP networks packets follow the shortest path from the source to their destination. This can cause congestion at certain places in the network because of an uneven traffic distribution. With MPLS Traffic Engineering predefined paths are being set up between the routers in the network, and each path has a requirement on how much bandwidth it needs. The network will now configure itself, using a constraints based routing protocol, to match all these requirements. Traffic might not always follow the shortest path anymore, but it will have a path where the bandwidth it needs is available.

Even on an overprovisioned network TE can be useful. Although in theory sufficient bandwidth should be available, there can always occur hotspots in the network, created by major outages, explosive traffic growth (in combination with slow capacity deployment) or uneven traffic distributions. With the use of TE traffic can be rerouted around these hotspots.

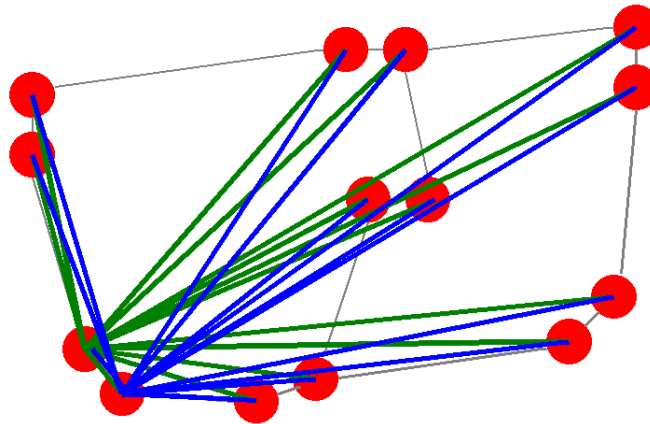


Figure 1 shows an example on how to perform TE in an IP backbone. The red dots are core routers, and every site has two of those for redundancy. From every core router a Label Switched Path (LSP), an MPLS 'tunnel', is set up to every other core router in the network (the picture shows this only for one site). The required bandwidth constraint is now configured on each LSP (based on collected statistics) and the network will route those LSPs over paths where the required bandwidth is available. The bandwidth values need to be adjusted daily or weekly from now on to keep the network optimized.

The above scenario has been deployed in Global Crossing's worldwide IP backbone since 2Q 1999, with currently over active 8000 LSPs. The network has been congestion free because of the use of Traffic Engineering.

Another, more recent, application of MPLS is Fast Reroute (FRR). When links or nodes fail, it takes time for the network to distribute the changed topology information to all the other nodes, who then re-converge to route around the failure. This typically takes in the order of several seconds in a best case scenario. Certain applications however (e.g. voice or trading) require a faster convergence than several seconds. FRR is the tool to achieve this.

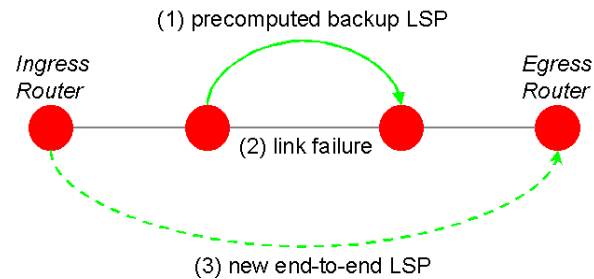


Figure 2 shows the working of FRR. For each protected link in the network, the routers on each side of that link precompute and set up a backup LSP to the router on the other side of the link. This LSP will be set up over a redundant path (1). If the link fails, the router will detect the failure and can immediately switch traffic to the local backup LSP (2). In the meantime it will signal back to the head-end router to compute a new end-to-end LSP, and traffic will flow over the local backup LSP until this new LSP is ready (3). It is important to note that the backup LSPs do not make bandwidth reservations like the primary LSPs do. This means that during the FRR period (a few seconds) the network might experience some congestion. DiffServ will be used to deal with that.

FRR is able to restore connectivity in the millisecond range when a failure occurs (in practice <100ms). The key to this is that the backup LSP already has been established and that only the router detecting the failure needs to take action. It does not require signaling between routers as is needed in networks without FRR. This brings SDH/SONET-like restoration times to IP networks. Although FRR is the only mechanism available to achieve these kind of restoration times, work is going on to improve the convergence of non-MPLS networks as well. By tuning the routing protocols (like IS-IS and OSPF) it seems possible to achieve convergence times under or around 1 second.

The deployment of MPLS is not trivial. It requires a 'seamless' network, built as a single Autonomous System (AS) and with a single internal routing protocol (IGP) without multiple areas or levels. This is often not the case in existing networks, and can slowdown the deployment of MPLS. Although interoperability between different router vendors is normally not a problem, having more than one vendor might slow down the deployment of new MPLS features (e.g. FRR) because both

of them need to fully support the standard. Another important issue is the education and training of the operations staff. MPLS means a lot of new complexity in the network, and the network operators will need to understand the technology.

Differentiated Services

We already discussed the use of classes to differentiate between various services on the same network. Although MPLS is deployed to prevent congestion, congestion could still happen because of major failures or during a FRR. In these periods we need to make sure that the critical services receive the quality they need, for example the delay and jitter guarantees for VoIP traffic. Queuing and scheduling mechanisms in the routers are used to achieve this goal.

Normally all traffic shares the same queue on an outgoing interface, and when more traffic is sent than the interface can service, the queue will fill up and eventually drop packets (e.g. when sending 200Mbps of IP traffic over an STM-1 155Mbps interface). This will cause delay, jitter and packet loss. If we use our example of 3 classes (Best Effort, Assured and Real-time), then we need to setup 3 parallel queues and service them according to the defined priorities.

Two basic queuing/scheduling mechanisms are available: Weighted Round Robin (WRR) and Strict Priority. The WRR mechanism will allocate a certain amount of bandwidth to each queue, to guarantee that each service will get its fair share (as defined) of the link bandwidth. The Strict Priority queue however will be serviced whenever there is a packet in this queue, meaning that traffic in this class has absolute priority over any other traffic. This can be used for the Real-time service, as it guarantees low delay and jitter.

Although the theory is not very complex, the configuration of the queue parameters proves to be very hard. The question is how to translate the service/application requirements to actual queue configurations. Experimentation will be needed to find the right values and fine-tune the configuration. Also keep in mind that this will only be effective in case of congestion.

In the classes defined before we use the Strict Priority queue for Real-time traffic, and WRR for Assured and Best Effort. If we expect the remaining traffic after serving the Strict Priority queue to be 20% of the Assured class and 80% Best Effort, then we can configure the WRR queue with 90% of the bandwidth for Assured traffic and 10% for Best Effort. In this setup the Assured service will be able to use up to 90% of the available bandwidth (excluding Real-time traffic) to make sure it won't drop any packets (as we expect only 20% traffic, based on earlier collected statistics). This 90/10 split is a rule of thumb based on experience, other configurations are possible as well.

When deploying DiffServ it is also very important to set up monitoring and statistics collection to deal with the classes defined on the network.

Conclusion

We have seen how to use MPLS for Traffic Engineering and Fast Reroute, and how to use DiffServ to deal with congestion. Now we can answer the second question to see if the classes definition is right: can end users distinguish between classes?

For the Real-time service we use FRR and Strict Priority queuing to meet the availability, delay and jitter guarantees. For the Assured traffic we allocate most of the available bandwidth left, so its availability requirement (packet loss) is met. The rest of the bandwidth is available for Best Effort traffic, like Internet. MPLS TE will optimize the network and increase the quality of all services.

This shows that the 3 defined classes are a good basis for differentiation in requirements and pricing.

The operational aspects of MPLS and DiffServ deployment should not be overlooked. Development, education and training will take a lot of resources, and it might not be worthwhile in every situation. It all depends on what is required of the network, and that will be different for every operator. Also keep in mind that too much complexity will create an unreliable and unstable network, so it is preferred to deploy the simplest solution that meets the requirements.