

# Comparison of Traffic Engineering Techniques in Connectionless and Connection-Oriented Networks

**This article will look at approaches to optimising the utilisation of network capacity, as demonstrated by the traffic engineering techniques used in IP networks and by older connection-oriented network architectures. The relative simplicity of the older approach should make it easier to identify the principles behind traffic engineering making the newer techniques easier to understand.**

**In modern IP-based networks, path selection is based on methods which have their roots in the connection-oriented approach. The basic ideas have been re-engineered to make use of traditional IP constructs such as link-state routing protocols and RSVP, which, through the use of more sophisticated algorithms, offer the prospect of near optimal traffic routing. Such methods are capable of using a more sophisticated approach to quality of service than was previously possible.**

**differentiated traffic engineering provides more insight into how paths may be determined**

## Introduction

There has been widespread interest in traffic conditioning in data networks ranging from the now well-established differentiated services (DiffServ) model<sup>1, 2</sup>, to more sophisticated models, which often utilise traffic-engineering-based approaches<sup>3</sup>. These optimise network loading in a way that is not possible with networks that direct traffic along paths solely determined by traditional routing protocols. This has led to a range of solutions that use pre-configured tunnels or dynamic routing algorithms based on estimated or measured network loadings.

This article will look at the principles behind a traffic engineering approach to the design and configuration of IP networks and compare this with the approach taken in older connection-oriented network architectures and designs. Examination of specific cases suggests that the latter has significantly influenced the former and

should be simpler than the modern network constructs. This should make it easier to assess the principles behind traffic engineering and make it possible to transfer this learning to the modern approaches, making them easier to understand. This will also highlight the differences between allocating traffic paths based on layer 2 link loadings rather than layer 3 routing protocols and help illustrate how modern technology has moved the debate on.

The basic ideas behind traffic engineering within modern networks will be considered and some of the approaches summarised, including that taken in modern multi-protocol label switching (MPLS) networks based on IETF standards for traffic engineering. Some of the basic tenets of a now obsolete network architecture known as Tymnet<sup>4, 5</sup> will then be considered. This approach dynamically allocated network paths to switched virtual circuits based on existing traffic loadings and available bandwidth. More recent MPLS traffic engineering developments covering quality of service, recovery from network component failure, and the complexities imposed by the hierarchical nature of link-state routing protocols, will also be examined. Further approaches, such as differentiated traffic engineering, which is derived from research taking place outside the standards bodies, are also explored – these provide more insight into how paths may be determined. Finally the article will examine how some of the ideas embodied in the Tymnet approach have found their way into more modern thinking and how new ideas have added to this thinking.

This article will show that, in meeting the need to optimise the use of bandwidth in modern IP-based networks, path selection based on methods which have their roots firmly planted in traditional connection-oriented networks have been used. The basic ideas of path creation and selection have been re-engineered to make use of traditional IP constructs like link-state routing and the resource reservation protocol (RSVP). Through use of more sophisticated algorithms, the prospect of

The Author: Ed Smith is with BT.

more optimal traffic routing can be realised. Such methods are capable of using a more sophisticated approach to resource utilisation and quality of service than was previously possible, although some observers argue strongly for simplification.

## Traffic Engineering in IP Networks

### Overview

Traffic engineering is the mechanism by which traffic traversing a meshed or partially meshed network is distributed to even out network loading and minimise costs. Its primary driver is the avoidance of points of congestion across the network, so that service level agreements can be met with maximum reliability at minimum cost.

This comes as a standard feature of layer 2 networks, which are engineered to route virtual circuits as efficiently as possible, allowing the traffic load to be more evenly spread across network trunks. This situation is not mirrored with layer 3 based networks, where traffic routing is governed by standard routing protocols, predominantly interior gateway protocols (IGPs) such as open shortest path first (OSPF) and intermediate system to intermediate system (IS-IS). These protocols direct traffic across the shortest available path irrespective of network loadings. This often leads to some trunk links being heavily loaded while others are only partially utilised.

Prior to the widespread adoption of MPLS<sup>6, 7</sup> this was not a problem, since most IP networks were built as virtual private networks across a layer 2 infrastructure, such as frame relay and ATM. The loadings on these networks could be managed by the careful routing of private virtual circuits across the infrastructure. However, the adoption of MPLS has in most cases removed the layer 2 structures which underpinned those traffic engineering capabilities of the network infrastructure.

As a result traffic engineering capabilities have been devised using the label switching capabilities defined by the IETF and documented in RFC2702<sup>3</sup>. Spraggs<sup>8</sup> describes the design goals of MPLS traffic engineering as being to:

- maximise the utilisation of network resources;
- allow MPLS to replicate the traffic engineering functions of layer 2 networks such as frame relay and ATM;
- integrate traffic engineering into layer 3;

- route traffic flows based on the resources available in the network;
- use either static routing or constraint based routing (CBR) to select the shortest path that meets the resource requirements of the traffic flow within constraints of available resources – a flow may have constraints based on bandwidth or media requirements and its priority against other flows;
- gracefully recover from link or node failures that change the topology of the backbone by adapting to a new set of constraints.

There are six main components of a traffic engineering solution. The first of these is the tunnel or traffic trunk attributes that describe the network requirements for that trunk or tunnel. These include the required bandwidth, operator route prioritisation parameters, the ability to pre-empt (that is displace) existing tunnels or to be pre-empted by new tunnels, response to failure and re-optimisation, and the means to explicitly include or exclude specific links from path evaluation.

The second component is the network resource attributes, which are inputs into the CBR calculations used to set up the traffic engineered path and constrain the placement of tunnels through the network. The first of these attributes is the available bandwidth, i.e. the amount of bandwidth available to set up new tunnels. The second is the resource class, which is checked against the policy set for the tunnel by the network administrator. This will determine whether the link in question should be included in the path calculation.

The distribution of network resource attributes is the third component. Resource attributes are the router's view of the capability of its attached links and allow the headend router to make tunnel routing decisions. This information needs to be distributed across the network, which is normally done using extensions to the IGP. This carries attribute information such as available bandwidth, as well as routing costs, using extensions to the basic link-state IGP, in order that each router has a full view of the network topology. The initial approach is restricted to a single OSPF or IS-IS area.

Path selection is the process performed on the headend router that determines the explicit path to be used for packet forwarding. This is effected by either manual configuration or by computation at the headend using CBR, building the shortest path starting from the destination, working towards the tunnel headend. It takes into account the current capabilities of

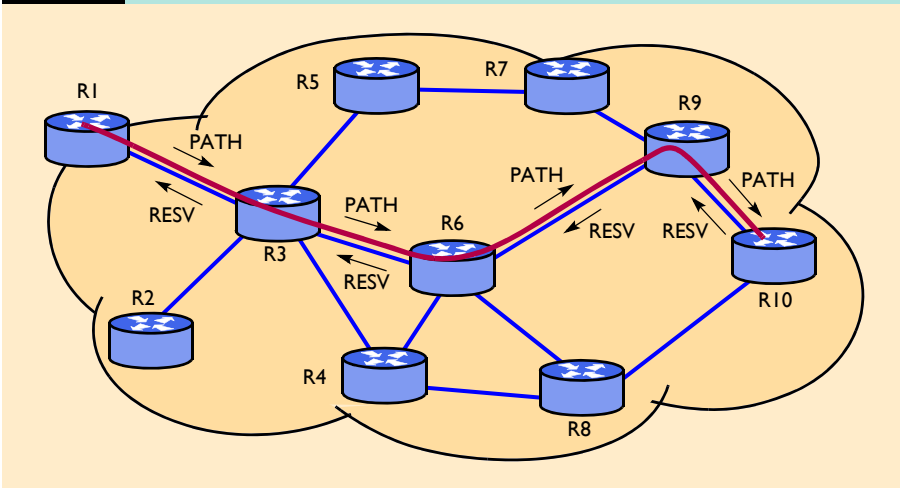
the network and has two stages, the first of which is the pruning stage. All routes that are explicitly excluded by the tunnel policy or available bandwidth are removed from the calculation. The path selection decision is primarily based on the path with the highest available bandwidth and then on the one with the fewest hops.

The traffic engineering set-up and maintenance mechanisms form the fifth component and are commonly achieved by using RSVP (also known as RSVP-TE)<sup>9, 10</sup> as a signalling protocol. The output of the path computation process is an explicit route expressed as a sequence of router IP addresses, which will form the route taken by the tunnel. This is held within the RSVP 'path' packet and is known as an explicit route object (ERO). Admission control is performed on all intermediate routers upon receipt of an RSVP 'path' message, which flows from the head to the tail of the tunnel. Each intermediate router performs admission control on the tunnel set-up request, using the bandwidth and tunnel priority information in the 'path' message to validate that there are sufficient resources to honour the request. If there are sufficient resources, the router will create 'path state' and transmit the 'path' message to the next router in the explicit route object after first removing its own address from the ERO. If insufficient resources are available the router creates an RSVP 'patherr' message, indicating that there has been an admission control failure and sends this back to the headend router (see Figure 1).

Once the 'path' message has reached the tail of the route, the final router creates a path state and sends a 'resv' message, containing an MPLS label, back along the path towards the headend router (see Figure 2). This effectively turns path state into a reservation at each router, with each router changing the MPLS label into the label it wishes the next router in the link towards the headend to use to send data to it. In the event of a reservation failure or if a link's bandwidth 'up' or 'down' threshold has been violated, due to tunnel set-up or tear down, then the affected router floods this information via the routing protocol.

The final component is the means of routing the traffic on to the traffic engineered tunnels once they are set up. To the routing process, a traffic-engineered tunnel looks like a unidirectional pipe, on to which only the headend router can put traffic. The headend router sees the tunnel as a logical interface and can map traffic on to it using either a static route or through a route dynamically set up by the IGP. The IGP shortest path algorithm, which computes the shortest path to a destination,

**Figure 1 Tunnel signalling**



has been modified to favour traffic engineered tunnels over other IP routes.

**Error recovery**

The traffic engineering approach also provides the capability for re-optimisation and restoration, the former being the process whereby some tunnels are re-routed to improve overall network efficiency and the latter providing link and path protection.

Re-optimisation follows a ‘make-before-break’ philosophy and typically occurs following the re-establishment of the primary path. The headend router establishes a new label switched path (LSP) for the traffic tunnel and once this is established the traffic is switched to the new path and the old one is torn down.

Restoration occurs when the headend detects that the label switched path that carries the tunnel has failed. The headend then follows the actions prescribed by the resilience attribute of the tunnel, which is specified by the network manager based on the trade-off between speed of restoration and the level of bandwidth tied up in reservations for stand-by paths. The

recovery action begins when the headend detects the failure, usually following notification by either RSVP or the IGP.

Link protection is an alternative, which provides faster restoration and is controlled by the two routers on either end of a link. When they detect a failure they switch the link to a preconfigured back-up link.

**Traffic Conditioning in the Connection-Oriented World**

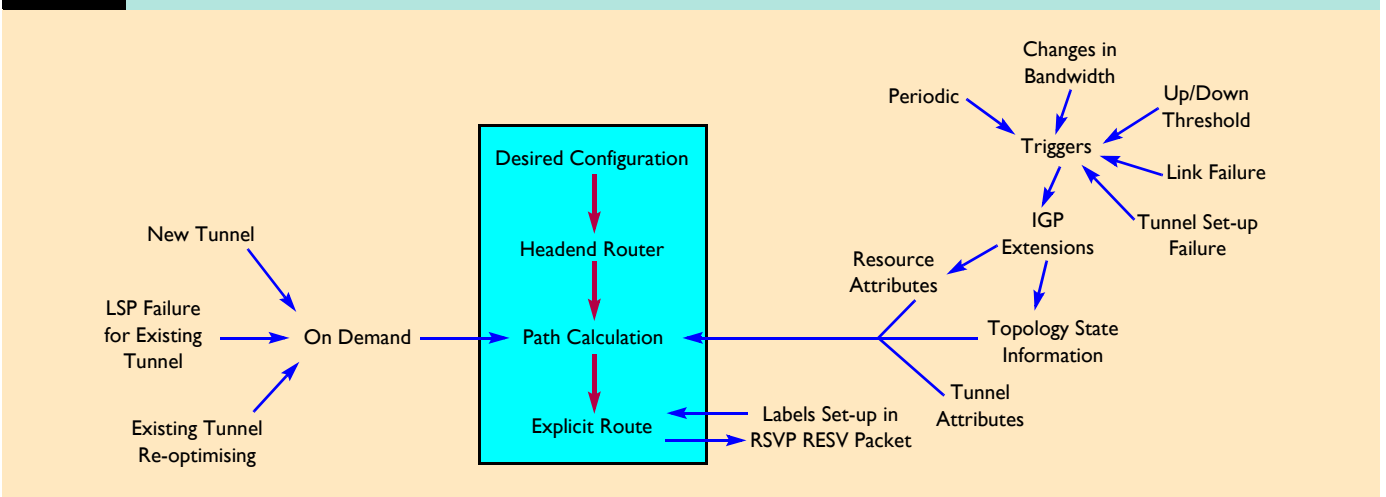
Having examined the details of traffic engineering in the connectionless world, it is appropriate to go and look at the use of similar ideas in a much earlier technology developed at a time when the majority of commercially available public data networks were connection oriented. One such approach is the now defunct Tymnet architecture, which worked by dynamically allocating network paths to switched virtual circuits based on existing traffic loadings and available bandwidth. By examining this

architecture it will be possible to pull out common themes, illustrating some key networking principles and learning from the differences between the two approaches.

The Tymnet network, which formed the basis of the Concert Packet Network<sup>11</sup>, came into being in the USA in 1971 and at that time consisted of 30 nodes. By 1995 it had grown in size to over 5000 nodes and 10 000 links, handling over a million sessions and 30 000 million bytes of user traffic per day. Each route and session was established by the Tymnet supervisor, which had visibility of all nodes and links in the network. Each virtual circuit was established using an algorithm that was primarily based on least hops, but also took link speeds, network load, session type (interactive or batch) and link type (terrestrial or satellite) into account. The allocated route was retained until either the user logged off or a node or link failure effected the re-routing of the session. The supervisor was backed up by four other stand-by supervisors, one of which would take over in the event that the primary supervisor failed. The active supervisor was the only location in the network with global network knowledge.

Traffic, network resource utilisation and network performance statistics were collected and used as the basis for capacity planning. The growth of the network was planned using this information and analysis of the observed growth patterns, with additional capacity being added as key planning thresholds were exceeded. Network delay statistics were used to assess whether any performance problems were down to a basic lack of capacity or inadequate connectivity out of a switching site. This led to either increasing capacity or redesigning local node clusters. Network planning tools were built incorporating the supervisor least-hop routing algorithm, to

**Figure 2 MPLS traffic engineering tunnel set-up overview**



allow more accurate traffic modelling. Capacity plans were also adjusted to take into account sales and marketing forecasts. The network was highly resilient with the backbone reaching an availability of 99.99%, while access links typically had an availability of 99.7%.

The Tymnet supervisor maintained a database of the current network topology and optimised the use of network resources. It collected and stored accounting information supplied to it by the nodes in the network. Tymnet nodes were connected together by trunk circuits running a proprietary protocol known as Tym 2, using logic defined within a software module in the node (also called an engine) known as Node Code. Within the Node Code there was a routine known as Leprechaun logic, which was used for building virtual circuits. This code maintained virtual circuits to the active and stand-by supervisors, known as a command circuit (see Figure 3).

The supervisor held network topology tables in memory, which described nodes and hosts. Nodes were identified using their node number, together with the type, speed and status of the links connecting them to other nodes. There were four types of link information – whether the link was up or down, shut, overloaded or unable to accommodate additional circuits passing through it. Information held by the host covered the host number (its unique identifier), the nodes to which it was connected, and the current host status.

There were five supervisors on the Tymnet network, only one of which could control the network at one time. This was said to be in the ‘awake’ state and ensured

that the other four remained ‘asleep’ by sending them ‘sleeping pills’ at regular intervals. If the currently active supervisor crashed, then it stopped sending out sleeping pills and the ‘sleeping’ supervisor with the lowest ‘drowsiness factor’ took over control of the network within two minutes, with no operator intervention. This then sent out sleeping pills to all the other supervisors attached to the network. Each supervisor was located at a different location and had circuits established to the other network utilities at all times. All command circuits were rebuilt every time a new supervisor took over.

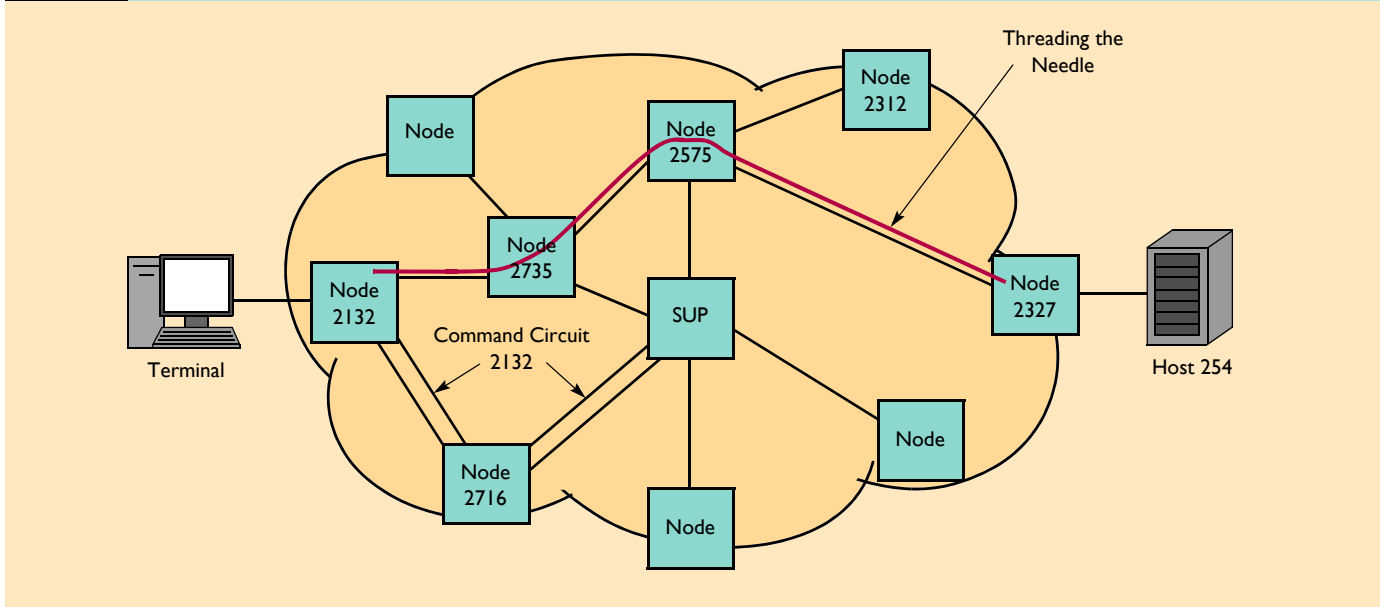
When one host requested a connection to another host, it sent a request to the supervisor using the command circuit. The supervisor then created a source-to-destination virtual circuit path based on the most economical use of network resources, thus avoiding congestion and balancing the load across the network. The objective was to keep routes as short as possible, to provide routes with adequate bandwidth, and to minimise delays across the network. This was done by calculating a cost index for each link, which was a function of the bandwidth, current traffic load and type of link. The cost index was increased for slow or overloaded links and became infinite for failed or shut links.

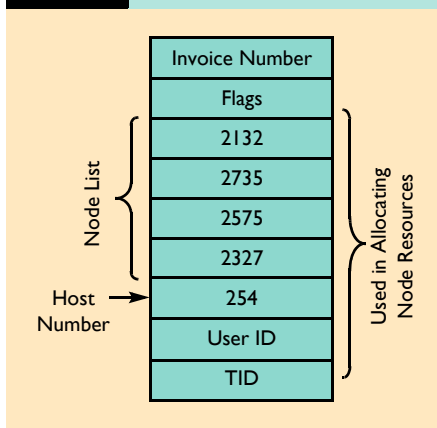
While Tym 2 was used for communications within the network, communications to the outside world were through more common protocols such as synchronous data link control (SDLC) or X.25. This code was run in a program partition known as a ‘host slot’ on the Tymnet node. When a virtual circuit was set up, the host slot made

a request to the node code, which notified the supervisor of the request via the command circuit, providing the address of the desired destination host and the security credentials of the source host. On successfully validating the request to establish the connection, the supervisor built the ‘needle’, which was made up of the invoice number (used for billing information), the destination node and host number, the origin node number, flags and TID (circuit type information used in node resource allocation), and a sequential list of nodes forming the optimal path to the destination node (see Figure 4). This was returned to the requesting node using the command circuit.

The needle was used by the source node, which first removed its node number from the node list and then built the next portion of the virtual circuit to the second node on the list. It then sent the needle to the next node on the virtual circuit’s path, which on receipt removed its node number from the node list and built the next part of the virtual circuit. This process was then repeated until the needle reached the destination node, by which time the circuit connection was made to the host slot, and the needle node list, after removal of the destination node number, was empty. The destination node then passed the needle stub back to the supervisor via its command circuit. The supervisor interpreted this as a signal that the circuit had been successfully built and that it could be billed. This process, of building a path across the network, was known as ‘threading the needle’ and the virtual circuit set up as a result was bidirectional.

**Figure 3** Setting up a Tym 2 virtual circuit



**Figure 4** Format of the Tym 2 needle

In the event that the originating node detected that a virtual circuit had gone down, it would inform the supervisor which, provided the end nodes were rebuild capable, would initiate a circuit rebuild process, that followed the same steps. The rebuild session maintained the same session invoice number and the user was unaware that the error had occurred. When a virtual circuit was cleared down, the clear down process was initiated from the destination end using a process known as 'circuit tear down'.

When a circuit was broken, the source of the break issued messages, known as 'soft zappers', which travelled the remainder of the circuit and caused the network resources allocated to it to be released. The end of a session was signalled by the host issuing a 'hard zapper', which also caused the resources to be released. If, however, one end of the circuit wished to flush unwanted data out of the virtual circuit without tearing it down, then this could be achieved by sending a packet known as a 'gobbler'.

When a link became heavily loaded on the network, the supervisor was informed and updated its view of the network accordingly.

The Tymnet network did not support what would be recognised today as a QoS capability with its associated sophisticated queuing and scheduling mechanisms. It did, however, provide traffic conditioning of a kind, by exercising frame-level flow control through a sliding window mechanism at the virtual-circuit level, using a capability known as 'back pressure'. This technique has much in common with the token bucket mechanisms employed in more modern networks. When a circuit was first established it was allocated a throughput rate known as a 'gouging level', which was the maximum throughput that could be attained. Each end of the circuit was allocated a counter, set to half the gouging level, which was expressed in terms of

**path computation makes use of revised bandwidth segmentation rules to route different tunnels by different paths across the network – thus a tunnel for carrying best effort traffic and one for carrying high priority voice traffic can be independently routed**

characters per second. Every time a character was sent, this counter was decremented until it reached zero, when no further characters could be sent. The receiving end would send a reset packet to the sender every half second, which would reset the counter to the gouging level, allowing data transmission to be resumed on that virtual circuit. This effectively stopped one virtual circuit consuming a disproportionate amount of the link's resources.

## Developments in Traffic Engineering

The section on traffic engineering in IP networking looked principally at the basic capability – however, the requirements of modern networks go beyond these basic mechanisms. The principal extensions are:

- differentiated services across MPLS traffic engineered tunnels;
- rapid re-routing around network failures;
- issues of network scale inherent in link-state IGP deployment.

These are briefly reviewed below to show the increased levels of sophistication that have to be handled by modern networks, which were not such key issues for the Tymnet technology, as discussed later in the article.

### DiffServ-aware traffic engineering

DiffServ-TE (DS-TE) makes traffic engineering DiffServ aware and takes into account separate engineering constraints for each class of service<sup>12</sup>. It allows different traffic classes to take different routes through the network. Unlike DiffServ, DS-TE is a control plane feature, i.e. its effect is achieved through signalling and not through data plane processes such as queuing. This approach is currently being standardised in the IETF as CoS-aware Traffic Engineering and relies on further extensions to IGP routing protocols to transport additional bandwidth allocation information.

This capability requires the bandwidth on the link to be segmented between different tunnels, while maintaining the

existing pre-emption rules. To enable this to happen, further modifications have been made to the constraint shortest path first (CSPF) algorithm, the IGP, RSVP and the bandwidth allocation process.

Path computation makes use of revised bandwidth segmentation rules to route different tunnels by different paths across the network. Thus a tunnel for carrying best effort traffic and one for carrying high priority voice traffic can be independently routed. Once the tunnel is set up, the headend router can allocate the traffic to a particular tunnel, based on static routing or dynamic routing, using the tunnel virtual interface in conjunction with well-known traffic classification techniques.

In a best-effort-based traffic-engineered network, the process is to find the appropriate route for traffic and set up the aggregate tunnel. In a traffic-engineered, DiffServ environment, the approach is to set up a tunnel for each class. For example, to achieve voice trunking in a multiservice network, DiffServ-aware MPLS-TE can be used to provide explicit admission control of EF traffic/voice trunks using EF-aware constraint-based routing. In combination with DiffServ-aware MPLS, traffic engineering provides rigorous QoS for voice services without relying on over-engineering.

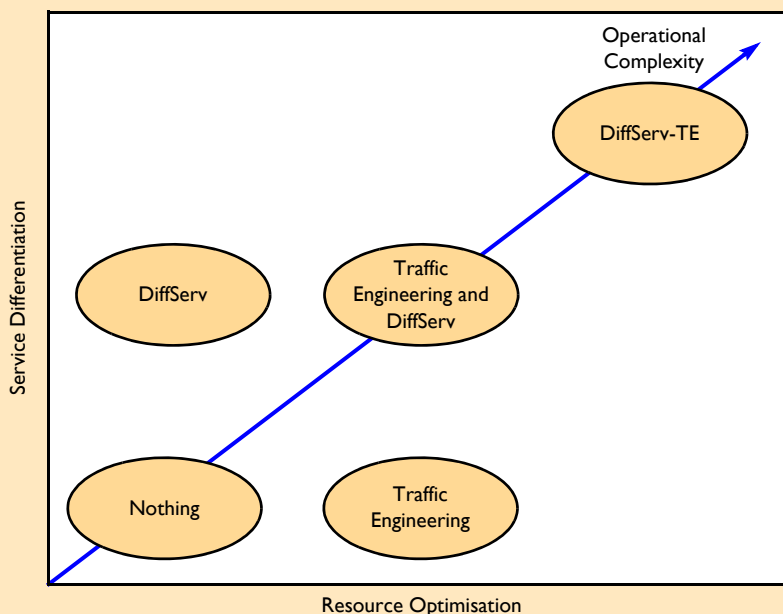
Two path calculation metrics can be used to validate a link – bandwidth or propagation delay. Each link can be configured with a traffic engineering metric that is different from the IGP metric. Each tunnel can be configured so that path calculation is carried out using the IGP metric or traffic engineering metric for that tunnel (typically data will use the former and voice the latter).

Figure 5 illustrates how DS-TE fits in with other network engineering capabilities in an MPLS network, providing improvements in service differentiation and resource optimisation at the expense of operational complexity. DS-TE is currently being standardised by the IETF.

### Re-routing strategies

Multiple approaches to routing strategies exist. The standard approach to protection and restoration in MPLS networks is

**Figure 5** Positioning DS-TE with respect to other MPLS techniques



achieved using the IGP, which protects against both link and node failure and gives convergence times of the order of 30 to 40 sec. This can be dramatically improved by a number of enhancements, such as:

- fast fault detection;
- fast SPF and LSA propagation triggering;
- priority flooding;
- the incremental Dijkstra algorithm;
- load balancing.

This can get convergence down to 1–3 sec, which may be acceptable for most applications, but some, such as voice trunking, need a more aggressive target – typically 50 ms<sup>12</sup>.

There are two basic approaches to implementing resilience in an MPLS infrastructure – protection and restoration. Protection uses a pre-established back-up path, while restoration dynamically calculates a new path; consequently protection is faster, but needs spare resources and provides stronger guarantees. These techniques may be combined in a single routing strategy.

The selection of the appropriate mechanism is influenced by the scope of recovery which can be local, at the link or node level, or performed by the node immediately upstream.

Alternatively, it can be global, with the repair originated by the headend, needing both restoration and protection. In this case the headend reacts directly to network signalling and consequently this is slower than local repair.

There are three techniques which can be used to provide accelerated convergence of ‘broken’ MPLS tunnels.

- **MPLS TE re-routing**  
This is the default, and relies extensively on RSVP and IGP signalling, with the headend of the tunnel initiating the rebuilding of tunnels. This technique has a restoration time of typically several seconds.
- **MPLS TE path protection**  
This is a global repair mechanism, using pre-allocated back-up routes. Consequently the rate-determining step for restoration is fault detection, again via IGP and RSVP mechanisms, with the re-routing process taking negligible time. It should be noted that the back-up route calculation is computationally intensive, but is done outside the recovery process. This approach yields a recovery time of hundreds of milliseconds.
- **MPLS TE fast re-route**  
This is a local protection mechanism, offering protection against the failure of a single node or link. This relies on either direct detection of a link failure or the loss of hello messages between adjacent nodes. The hello mechanism is implemented as an extension of RSVP. This process works using a statically configured tunnel to bypass either the node or link to be protected. Although the tunnels are set up through network

configuration, their use is enabled through label propagation at tunnel set-up time, using further extensions to RSVP. When failure is detected, re-routing is not dependent on signalling, but is effected by the node which detects the failure, adjusting its label swapping mechanisms to reflect the use of an alternate tunnel. The re-routing is, however, signalled to the headend and this may instigate route optimisation. Fast re-route is configured on a per-node basis and gives re-routing times of the order of 50 ms.

MPLS TE fast re-route may be used in specific parts of the network where very fast convergence is required. Compared to other protection schemes, such as SDH, no back-up bandwidth is wasted. It is possible to mix IP routing restoration and MPLS TE fast re-route protection, with its back-up tunnel path computation and provisioning.

Complexity is driven by the parameters which take into account the degree of optimisation required. The level of complexity will also determine whether the back-up tunnel computation is done off-line or on-line (distributed).

In summary, MPLS TE re-route (global restoration) is the default, but is relatively slow to recover from failure. Path protection (global protection) is used if just a few protected LSPs are required and sub-second recovery is not needed. MPLS fast re-route (local protection) is the most efficient protection scheme, providing 50 ms recovery in the case of link and node protection.

### MPLS multi-area traffic engineering

Traffic engineering relies heavily on the use of link-state routing protocols to transport relevant information across the network. These are hierarchical in nature, dividing the network up into several regions often known as areas, in order to make the maintenance of routing information easier. Each area will usually have access to detailed topology information about itself and to summary information about the other areas. Two or more areas intersect at an area border router (ABR), which maintains detailed topology information about all intersecting areas. Consequently

**MPLS TE fast re-route may be used in specific parts of the network where very fast convergence is required – compared to other protection schemes, such as SDH, no back-up bandwidth is wasted**

the headend router does not have a full view of the entire network, which poses particular problems for the generation of traffic engineered tunnels.

There are various approaches for inter-area path computation and set up<sup>13, 14</sup>, one of which is known as the systematic approach. Here the headend inserts a marker in its 'path' message that gives the ABR an 'indicative' route to the tunnel destination router. The ABR then refines this marker and updates the 'path' message with its view of how to reach the next ABR in the path to the tunnel destination. This second ABR will then complete the path detail required to construct the tunnel. This approach involves a significant element of trial and error in evaluating the path environments available through different ABRs.

An alternative approach is outsourced path computation, either using a centralised path computation server (PCS) or by distributed path computation, with the headend and tail end ABRs sharing the PCS role. The central path computation server, having a view of the traffic engineering information covering all of the domain's resources and a topology database for the whole network, computes an end-to-end 'optimal' path. Any label switching router (LSR) (also called a PCC – path computation client) requiring the computation of an inter-area TE LSP sends a request to the path computation server (PCS) and in turn gets a computed path (see Figure 6). The PCS provides the PCC with an explicit route object, which is a sequential list of the

router IP addresses that will comprise the required tunnel. Using this information the headend router instigates building the path using RSVP-TE. In many ways this resembles the functionality offered by the Tymnet supervisor.

In distributed path computation, the request is sent from the headend router to the headend ABR, which in turn sends the request to the tail end ABRs through which the tail end router may be reached. Each tail end ABR returns its most optimal path to the tail end router. The headend ABR then evaluates the best end-to-end LSP, communicates this to the headend router, and the inter-area TE LSP is then signalled using normal RSVP TE operation.

More work is needed in this area to consider the issues of TE LSP path survivability in the field of traffic engineering across multiple areas.

### Differentiated Traffic Engineering for QoS Provisioning

Differentiated traffic engineering<sup>15</sup> (DTE) is a means of providing quality of service by maintaining core link utilisation below a threshold value, while eliminating complex packet processing tasks such as policing, shaping and scheduling in the core of the network. The traffic engineering component of this architecture handles congestion control and load balancing, by exploiting different paths for different service classes.

It could be argued that this approach has much in common with DS-TE (Tabatabaee et al<sup>15</sup> focus much of their discussion on a comparison with DiffServ itself); however, path evaluation and the allocation of classes of service to specific paths is handled using a centralised DTE controller device. This device, it is argued<sup>15</sup>, bears some similarities with the bandwidth broker described in RFC2638<sup>16</sup> and certainly appears to replicate some of the functionality of the Tymnet supervisor.

The bandwidth broker<sup>16</sup> was proposed to promote dynamic bandwidth allocation across networks, including multiple ISPs. It had two responsibilities:

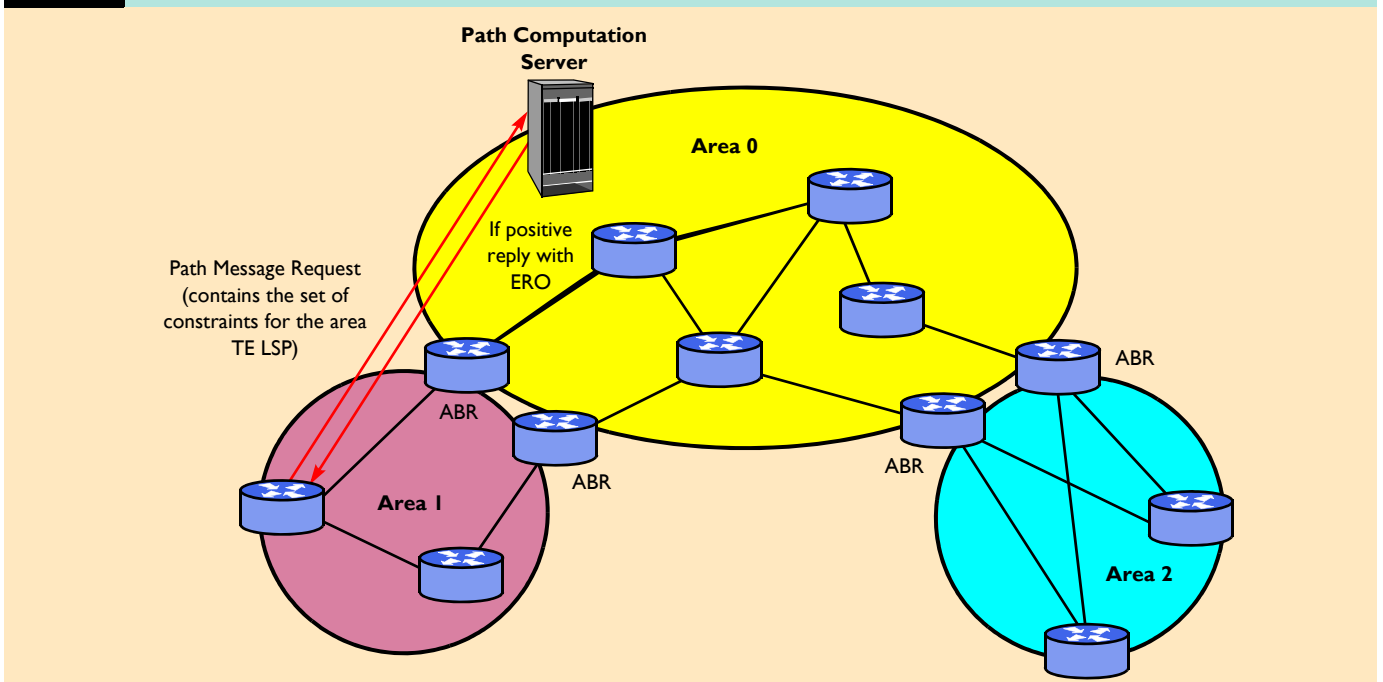
- to allocate its region's marked traffic capacity, setting up the leaf routers within its local domain accordingly;
- to manage the messages sent across boundaries to an adjacent region's bandwidth broker.

The bandwidth broker had a policy database and authenticated users requesting a bandwidth allocation verifying that there was sufficient bandwidth to meet the request.

The DTE controller monitors link utilisation across the network and derives the nominal traffic demand. It uses this information to provide the network path based on the following components.

- Management and monitoring unit  
This is responsible for requesting new outputs from three traffic engineering algorithms, passing them the input data and gathering their output. Two of these

**Figure 6** Role of external path computation server in setting up traffic-engineered tunnel in a multi-area IGP environment



algorithms (the path selection and routing algorithm (PSR) and the path to class assignment (PCA) algorithm) are also components of the DTE controller.

- PSR algorithm

This is responsible for inserting and removing paths between input and output pairs. The PSR updates paths on a slow time-scale of the order of many hours.

- PCA algorithm

This assigns paths to different classes of service. Much of the description of differentiated traffic engineering is concerned with the algorithm used by the PCA to minimise the network resources required to provide the desired levels of service.

The third traffic engineering component of the architecture is load distribution, which is not a feature of the DTE controller, but is a function carried out by the ingress routers, making use of short-term on-line traffic demand variables to work with traffic conditioning and admission control to control on-line load distribution between the available paths. The ingress edge routers specify paths for packets based on their destination and service class, performing traffic conditioning and packet buffering for packets going out on the same link in a single first in, first out (FIFO) queue.

The DTE controller monitors the system on a continuous basis and may intervene should the performance deteriorate unacceptably.

The basic architecture allocates responsibility for traffic conditioning, on-line traffic engineering, classification, labelling, link utilisation monitoring and reporting to the ingress router. The core of the network is responsible for managing link utilisation, link monitoring, label switching and reporting, with the egress router shaping the aggregated traffic that exits the network according to the configuration given by the DTE controller. The goal is to avoid over-provisioning through a simple regime, which ensures that only total link utilisation is monitored. This provides flexibility, diminishes the dependence on rigorous network planning, and reduces the need for frequent and expensive upgrades. Alternative schemes, it is argued, depend on configurations which increase in number as the size of the network increases and which are therefore more costly to administer.

The model is based on standard traffic engineering principles, but ensures that the utilisation of all links in the network comply with the requirements of the most stringent

traffic class. It utilises a multi-path-source-based routing model, allocating paths to different classes of service and over-provisioning paths that carry sensitive data. In an ideal DTE network a limited set of paths are over-provisioned and this is sufficient to remove the burden of QoS enforcement inside the network. This approach has not been implemented by a major router manufacturer, so the effectiveness of the method has to be evaluated through modelling and simulation. This shows<sup>15</sup> that the dependency of performance on traffic demand is not symmetric and is found to be more sensitive when the demand deviates above the nominal value. It is clear that path states should not be updated too frequently and that it is better to be more conservative in assessing the nominal traffic demand.

## Analysis

This article has looked at the principles behind the optimised utilisation of network capacity as implemented by the traffic engineering techniques used in IP networks and as implemented on a specific packet-based connection-oriented network known as Tymnet. As the issue of improving resource utilisation is addressed, the connectionless network begins in places to resemble the connection-oriented world. There would seem to be minimal difference between a switched virtual circuit and an MPLS traffic engineered tunnel.

Key similarities between these approaches appear to be:

- separation of the processing of user data packets (the data plane) and signalling packets (the control plane);
- use of a source-based routing approach for establishing data paths;
- the selection of routes based on circuit requirements and available network resources;
- active monitoring of available network resources, either centrally, as in the case of Tymnet, or in a more dispersed manner using IGP signalling in the case of traffic engineering;
- a level of traffic conditioning, which in Tymnet is really no more sophisticated than traffic pacing, but uses a richer set of techniques in the IP world.

It is interesting to note that although the initial portion of signalling takes place going from the headend of the tunnel/switched virtual circuit (SVC), in the case of the RSVP signalled traffic engineering solution, the label signalling which actually implements the tunnel is set up in the reverse direction.

It is important to note that while a label switched path is a simplex channel (similarly, an RSVP reservation is also simplex in nature), a Tymnet virtual circuit is a full duplex communications channel. Nevertheless, there are similarities between them. For the purposes of forwarding traffic, a label switching router contains a table known as the label forwarding information base, which relates an incoming label to the corresponding outgoing label, outgoing interface and outgoing link-level information. Consequently the label for the incoming packet acts as an index into this table, allowing the LSR to swap this label with the desired outgoing label and to identify the appropriate outgoing interface<sup>6</sup>.

For a Tym 2 packet, individual streams on a link are identified by a channel number, which is specific to that link. On entry into the node, this is translated into an absolute channel (all channels available to the node will have a unique absolute channel number within that node). This can be used to index a table known as the Permuter table, which holds pointers to internal buffers being used for that channel and the absolute channel number of the outgoing channel. Using a reversal of the logic used to translate the incoming link and relative channel number into the absolute channel number, the relative channel number and link for the outgoing packet can be identified from the outgoing absolute channel number. This has similarities with label switching, with the incoming and outgoing packets containing different relative channel numbers, rather than different MPLS labels.

Traditionally the IP world has preferred to avoid centralised routing control, preferring to spread the decision-making process across the network nodes with status information being transferred using a traditional IGP protocol, and tunnel signalling being transported using RSVP extensions. It is clear that a centralised approach has been considered. This is shown first by the proposed use of a bandwidth broker<sup>16</sup> to perform network

**the DTE controller monitors the system on a continuous basis and may intervene should the performance deteriorate unacceptably**

**Table 1** Comparison of the functions of the Tymnet supervisor and the DTE controller

Function	Tymnet supervisor	DTE controller
Participates in call admission control	Yes	Yes
Configures traffic conditioners	No	Yes
Monitors link utilisation	Yes	Yes
Source-based multipath routing	No – there is one source-based path per virtual circuit	Yes
Ingress edge routers specify paths for packets based on destination and service class	Partially – the path selection will be based on the session type requested at call set-up.	Yes
Resilient structures	Yes – multiple stand-by supervisors	Not specified
Path to class assignment	Determined at call set-up time by session type	Yes

resource allocation and authentication across multiple network domains, and then by the suggested use of an external path computation server in some approaches to managing the set-up of traffic engineered tunnels across multiple IGP areas<sup>13, 14</sup>. A third variant on the external-server-based approach is the DTE controller proposed by Tabatabaee et al<sup>15</sup>. A comparison of this device and the Tymnet supervisor is listed in Table 1.

Neither of these models proposes the use of sophisticated traffic conditioning in the core. It is also worth noting that the Tymnet supervisor is the only one of the 'network controller' devices described that has an explicit resilience regime defined.

The Tymnet model is clearly very simple and provides a mechanism which is very similar to that deployed in the emerging traffic engineering standards from the IETF. The Tymnet architecture operates using centralised path calculation and provides a simpler, more straightforward implementation than is possible in the IP world. The increase in complexity with IP is exacerbated by the need for IP to accommodate the new demands imposed by the changing paradigms in computing and networking technology. Examples of these changes are:

- the increasing dominance of IP technology;
- dramatic lowering of the price of bandwidth, with ever-increasing link speeds and traffic demands;
- the enabling power of Moore's law has meant that the increased computational power and memory available to routing devices at minimal increase in cost has greatly increased their packet switching power and ability to handle complex data plane activities such as QoS;
- the greater demands placed on network performance, with the need to process delay-sensitive traffic, such as voice, and to provide for applications like voice trunking that requires rapid recovery from network component failure;

- the requirement for service discrimination available through developments like DiffServ.

This has meant that developments which were never envisaged for the Tymnet environment, such as DiffServ TE and fast re-routing, are now required and the network has become more versatile, but at the same time more complex.

## Conclusions

In meeting the need to optimise the use of bandwidth in modern IP-based networks, path selection, based on methods which have their roots firmly planted in traditional connection-oriented networks, has been used.

The basic ideas of path selection and notification have been re-engineered to make use of traditional IP constructs like link-state routing protocols and RSVP and through use of more sophisticated algorithms offer the prospect of more optimal traffic routing. Such approaches are capable of using a more sophisticated approach to quality of service than was previously possible, although some observers argue strongly for simplification.

This article has looked at both a connection-oriented and connectionless approach, which, sharing the same cost reduction goals, have adopted very similar techniques to network cost optimisation. (It could be argued that in doing so the connectionless approach has adopted many of the characteristics of a connection-oriented approach, with the differences between a tunnel and a virtual circuit being difficult to discern.) The Tymnet approach appears to provide a credible stepping stone to understanding the principles used within traffic engineering, both in its standard form and more advanced forms, where routing can be carried out on a per-class basis and fast recovery from errors can be implemented.

## References

- 1 Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z. and Weiss, W. An Architecture for Differentiated Services. IETF RFC2475, December 1998.
- 2 Carter, S. F. Quality of Service in BT's MPLS-VPN Platform. *BT Technology J*, April 2005, **23**(2), pp 61–72.
- 3 Awduche, D., Malcolm, J., Agogbua, N., O'Dell, M. and McManus, J. Requirements for Traffic Engineering Over MPLS. IETF RFC2702, September 1999.
- 4 Tymnet – <http://en.wikipedia.org/wiki/Tymnet>
- 5 Detailed Description of Tymnet – <http://www.cap-lore.com/Tymnet/ETH.html>
- 6 MPLS/Tag Switching – [http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito\\_doc/mpls\\_tsw.pdf](http://www.cisco.com/univercd/cc/td/doc/cisintwk/ito_doc/mpls_tsw.pdf)
- 7 Rosen, E., Viswanathan, A. and Callin, R. Multiprotocol Label Switching Architecture. IETF RFC3031, January 2001.
- 8 Spraggs, S. Traffic Engineering. *BT Technology J*, July 2000, **18**(3), pp137–150.
- 9 Braden, R., Clark, D. and Shenker, S. Integrated Services in the Internet Architecture. IETF RFC1633, June 1994.
- 10 Braden, R., Zhang, L., Berson, S., Herzog, S. and Jamin, S. Resource Reservation protocol (RSVP) – Version 1 Functional specification. IETF RFC2205, September 1997.
- 11 Hwang, G. and Seah, P. Planning the Concert Packet Services Network, *Journal of the Institution of British Telecommunications Engineers*, 1994, **13**(3), pp 227–232.

- I2 Advanced Topics in MPLS TE Deployment – [http://www.cisco.com/warp/public/cc/pd/iosw/prodlit/mwglp\\_wp.pdf](http://www.cisco.com/warp/public/cc/pd/iosw/prodlit/mwglp_wp.pdf)
  
- I3 Draft-ash-multi-area-te-compare-00.txt – <http://www3.ietf.org/proceedings/02mar/I-D/draft-ash-multi-area-te-compare-00.txt>
  
- I4 Draft-Kompella-mpls-multiarea-te-04.txt – <http://tools.ietf.org/html/draft-kompella-mpls-multiarea-te-04>
  
- I5 Tabatabaee, V., Bhattacharjee, R., La Mark, R. J. and Shayman, M. A. Differentiated Traffic Engineering for QoS Provisioning. Infocom 2005 (24th Annual Joint Conference of the IEE Computer and Communications Society), *Proceedings IEEE*, March 2005, Vol 4, pp 2349–2359.
  
- I6 Nichols, K., Jacobson, V. and Zhang, L. A Two Bit Differentiated Service Architecture for the Internet. IETF RFC2638, July 1999.

### Biography



**Ed Smith**  
BT

Ed Smith is a consulting engineer within BT Global Services, working with major clients across a wide range of technologies. He has been with BT since 1982 after developing his IT skills with United Biscuits, handling real time process control systems, and Elida Gibbs, developing transaction processing middleware and data communications solutions. He holds BSc and PhD degrees from the University of Leicester and a Post Graduate Certificate in Commercial Management from De Montefort University. He is a Fellow of the British Computer Society, a Chartered Information Technology Practitioner and a Chartered Engineer.

[edward.a.smith@bt.com](mailto:edward.a.smith@bt.com)