TEAM: A Traffic Engineering Automated Manager for DiffServ-Based MPLS Networks

Caterina Scoglio, Georgia Institute of Technology
Tricha Anjali, Illinois Institute of Technology
Jaudelice Cavalcante de Oliveira, Drexel University
Ian. F. Akyildiz, Georgia Institute of Technology
George Uhl, NASA Goddard Space Flight Center

ABSTRACT

In this article an automated manager called Traffic Engineering Automated Manager (TEAM) for DiffServ/MPLS networks is introduced, and its design and implementation details are discussed. TEAM is designed for complete automated management of an Internet domain. TEAM is an adaptive manager that provides the required quality of service to the users and reduces the congestion in the network. The former is achieved by reserving bandwidth resources for the requests and the latter by distributing the load efficiently. These goals are achieved by online measurements of the network state. TEAM is composed of a Traffic Engineering Tool (TET), which adaptively manages the bandwidth and routes in the network, a Measurement and Performance Evaluation Tool (MPET), which measures important parameters in the network and inputs them to the TET, and a Simulation Tool (ST), which may be used by TET to consolidate its decisions. These three tools work in synergy to achieve the desired network operation objectives. The experimental results demonstrate the efficiency of TEAM as a network manager in different and unpredictable traffic conditions at the expense of a limited increase in the computational complexity and costs.

INTRODUCTION

The combined use of the differentiated services (DiffServ) and multiprotocol label switching (MPLS) technologies is envisioned to provide guaranteed quality of service (QoS) for multimedia traffic in IP networks, while effectively using network resources [1]. The basic idea behind MPLS is to attach a short fixed-length label to packets at the ingress router of the MPLS domain. These edge routers are called label edge routers (LERs), while routers capable of forwarding both MPLS and IP packets are called label switching routers (LSRs). The packets are then routed based on the assigned label rather than the original packet header. The label assignments are based on the concept of a forwarding equivalent class (FEC). According to this concept, packets belonging to the same FEC are assigned the same label and generally traverse through the same path across the MPLS network. An FEC may consist of packets that have common ingress and egress nodes, or the same service class and same ingress/egress nodes, and so on. A path traversed by an FEC is called a label switched path (LSP). The Label Distribution Protocol (LDP) and an extension to the Resource Reservation Protocol (RSVP) are used to establish, maintain (refresh), and tear down LSPs. One of the most attractive applications of MPLS is traffic engineering (TE), since LSPs can be considered as virtual traffic trunks that carry flow aggregates [2]. However, MPLS by itself cannot provide service differentiation, which brings up the need to complement it with another technology capable of providing such a feature: DiffServ. By mapping the traffic from different DiffServ classes of service on separate LSPs, DiffServ-aware MPLS networks can meet engineering constraints specific to the given class on both shortest and non-shortest path. This TE strategy is called DiffServ-aware TE (DS-TE). In [3] the authors suggest how DiffServ behavior aggregates can be mapped onto LSPs. Such DiffServ-based MPLS networks should not be managed manually, since the network needs to respond promptly to changing traffic conditions. Therefore, automated managers are needed to simplify network management and to engineer traffic efficiently [4].

This work was supported by NASA Goddard and Swales Aerospace under contract no. S11201 (NAS5-01090) and NSF under award number 0219829.

0163-6804/04/$20.00 © 2004 IEEE

With the objective to study and research the issues mentioned above, an IP QoS testbed composed of Cisco routers was assembled in the Broadband and Wireless Networking Laboratory (BWN-Lab). The testbed is a high-speed top-of-the-line mix of highly capable routers and switches for testing DiffServ and MPLS functionalities. During experiences with the testbed in a joint project with NASA Goddard about QoS in IP networks, the need for an improved set of algorithms for network management was clear and consequently an integrated architecture for an automated network manager was realized. This led to the design and implementation of the Traffic Engineering Automated Manager (TEAM) tool. Individual problems addressed by TEAM may already have been considered, but an integrated solution did not exist in the research field. TEAM is developed as a centralized authority for managing a DiffServ/MPLS domain and is responsible for dynamic bandwidth and route management. Based on the network states, TEAM takes the appropriate decisions and reconfigures the network accordingly. TEAM is designed to provide a novel and unique architecture capable of managing large-scale MPLS/DiffServ domains.

Related Work

TEAM is an integration of a set of adaptive and efficient network management techniques. In particular, the following management issues are addressed:

- *Resource management*: New schemes were developed to dynamically set up and dimension LSPs, allocate their capacity based on traffic estimation, and preempt low-priority LSPs to accommodate new high-priority LSPs depending on the actual load on the network.
- *Route management*: A new routing scheme was developed to establish LSPs and forward packets on a state-dependent basis to meet the QoS requirements.

The integration of the above mentioned techniques results in a valuable resource for a network manager in order to provide QoS and better network resource utilization. As mentioned before, much effort has been concentrated in the literature on individual research topics that are parts of TEAM. For example, continuous tuning of the network based on online modeling, parameter search, and simulation capabilities of a simulation system was proposed. Another approach for automated and software-intensive configuration management of network inventory was given in [5]. An architecture for the design and implementation of active nodes to support different types of execution environment, policy-based driven network management, and a platform-independent approach to service specification and deployment was proposed in [6]. A QoS network management system based on Open Shortest Path First (OSPF) with traffic engineering extensions, MPLS for explicit routing of packets, and a QoS path provisioning algorithm for call admission control was proposed. We also mention design based routing and the Cisco MPLS Tunnel Builder as partial efforts toward network management. The design based routing is a routing algorithm where optimized paths computed offline are used to guide online LSP setups. The Cisco MPLS Tunnel Builder is a Web-based graphical application that simplifies visualization and configuration of MPLS tunnels in a network.

Few comprehensive TE managers have been proposed in literature, and furthermore they address only a subset of the issues covered by TEAM. For example, the Routing and Traffic Engineering Server (RATES) [7] is a software system developed at Bell Laboratories for MPLS TE and is built using a centralized paradigm. RATES communicates only with the source of the route and spawns off signaling from the source to the destination for route setup. RATES views this communication as a policy decision and therefore uses Common Open Policy Service (COPS) protocol. RATES uses a relational database as its information store. It implements the Minimum Interference Routing Algorithm (MIRA) [8] to route LSPs. It consists of the following major modules: explicit route computation, COPS server, network topology and state discovery, dispatcher, graphical user interface, open application programming interface, data repository, and a message bus connecting these modules. Summarizing, RATES is a well designed TE tool, but TE is only performed for the routing of bandwidth guaranteed LSPs.

Another state-dependent TE mechanism to distribute network load adaptively is suggested in [9]. MPLS Adaptive Traffic Engineering (MATE) assumes that several explicit LSPs have been established between an ingress and egress node in an MPLS domain using a standard protocol like RSVP-TE. The goal of the ingress node is to distribute the traffic across the LSPs. It is important to note that MATE is intended for traffic that does not require bandwidth reservation, with best effort traffic the most dominant type. Since the efficacy of any state-dependent TE scheme depends crucially on the traffic measurement process, MATE requires only the ingress and egress nodes to participate in the measurement process. Based on the authors' experience, available bandwidth was considered difficult to be measured, so packet delay and loss have been selected for measurement purposes. The network scenario for which MATE is suitable is when only a few ingress-egress pairs are considered. In fact, for a network with N nodes and x LSPs between each pair of nodes, the total number of LSPs is of the order of xN^2, which can be a large number. Furthermore, MATE is not designed for bandwidth guaranteed services.

Traffic Engineering for Quality of Service in the Internet at Large Scale (TEQUILA) [10] is a European collaborative research project looking at an integrated architecture and associated techniques for providing end-to-end QoS in a DiffServ-based Internet. In TEQUILA a framework for service level specification (SLS) has been produced, an integrated management and control architecture has been designed, and cur-

The Design Based Routing is a routing algorithm where optimized paths computed offline are used to guide online LSP setups. The Cisco MPLS Tunnel Builder is a Web-based graphical application that simplifies the visualization and configuration of MPLS tunnels in a network.

TEAM has a central server, the Traffic Engineering Tool, which is supported by two additional tools: the Simulation Tool and Measurement/ Performance Evaluation Tool.

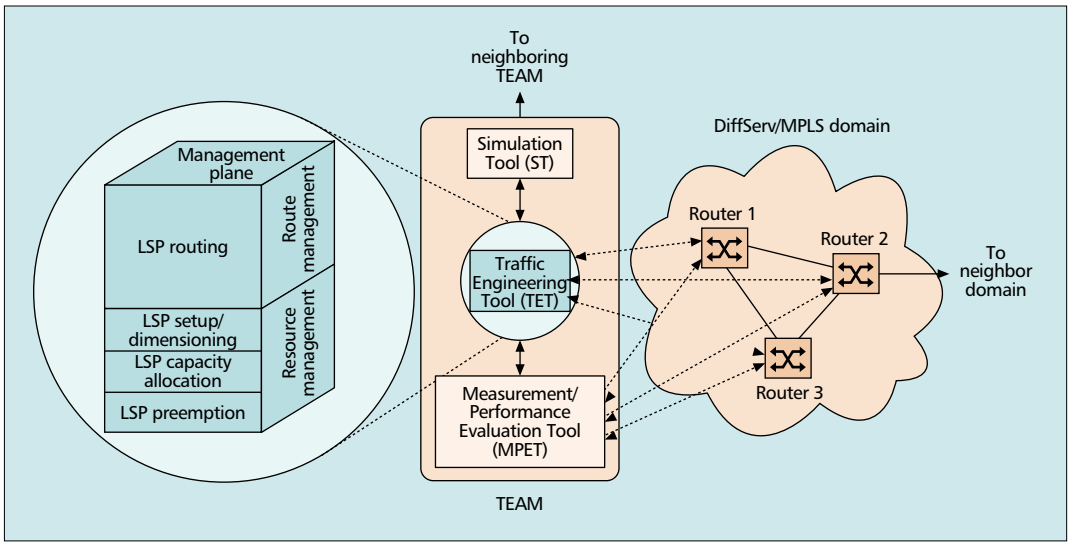

Figure 1. *TEAM framework and functionalities.*

rently MPLS and IP-based techniques are under investigation for TE. The TEQUILA architecture includes control, data, and management planes. The management plane aspects are related to the concept of a bandwidth broker (BB), and each autonomous system should deploy its own BB. The BB includes components for monitoring, traffic engineering, SLS management, and policy management. The TE subsystem is further decomposed into modules of traffic forecast, network dimensioning, dynamic route management, and dynamic resource management. The MPLS network dimensioning is based on the hose model, which is associated with one ingress and more than one egress node. The dynamic route management module considers:

- Setting up the forwarding parameters at the ingress node so that the incoming traffic is routed to LSPs according to the bandwidth determined by network dimensioning
- Modifying the routing according to feedback received from network monitoring
- Issuing an alarm to network dimensioning if the available capacity cannot be found to accommodate new connection requests

The dynamic resource module aims to ensure that the link capacity is appropriately distributed among the per-hop behaviors (PHBs) sharing a link by appropriately setting buffer and scheduling parameters. TEQUILA architecture is very interesting and shows a similar approach to MPLS network design and management to that of TEAM. However, to our knowledge the algorithms and techniques to be implemented in TEQUILA are not defined in detail at the moment, and their quantitative evaluation has not been carried out.

The use of MPLS for TE, QoS provisioning, and virtual private networks was decided at GlobalCrossing [11]. Approximately 200 routers participate in the MPLS system. Since a full meshed network would result in an MPLS system of about 40,000 LSPs, it was decided to deploy a hierarchical MPLS system of two layers of LSPs. To deploy an MPLS system for TE, the following procedure was proposed based on network operator experience:

- Statistics collected for traffic utilizing LSPs
- Deploy LSPs with bandwidth constraints
- Periodic update of LSP bandwidth
- Offline constraint-based routing

To provide QoS, MPLS is used in combination with the DiffServ architecture. It is desirable to use different LSPs for different classes. The effect is that the physical network is divided into multiple virtual networks, one per class. These networks can have different topologies and resources. The end effect is that premium traffic can use more resources. Many tools are needed for designing and managing these virtual networks. The use of MPLS for TE and QoS decided by an important Internet service provider (ISP) is confirmation that MPLS is a very promising technique even from a business point of view. The solution provided by TEAM is in line with the QoS architecture defined by GlobalCrossing. In [11] the authors demonstrated that MPLS TE has been effective in meeting the delay and jitter bounds required by applications.

In this article we present the architecture of TEAM, the adaptive network manager, with a brief description of the individual techniques. We present the framework of the TEAM tool. This is followed by the implementation details of TEAM and experimental results. We then conclude the article.

THE TEAM FRAMEWORK

The architecture of the TEAM is shown in Fig. 1. TEAM has a central server, the Traffic Engineering Tool (TET), which is supported by two additional tools: the Simulation Tool (ST) and Measurement/Performance Evaluation Tool (MPET). The TET and MPET interact with the routers and switches in the domain. The MPET provides a measure of the various parameters of the network and routers like available bandwidth, overall delay, jitter, queue lengths, and number of packets dropped in the

routers. This information is input to the TET. Based on this measured state, the TET performs resource and route management in the network. It decides the course of action, such as to create a new LSP or vary the capacity allocated to a given LSP, to preempt a low-priority LSP to accommodate a new one, or to establish a path for an LSP requiring a specified QoS. The TET also automatically implements the action, configuring the routers and switches in the domain accordingly. Whenever required, the TET can consolidate the decision using the ST. The ST simulates a network with the current state of the managed network and applies the decision of the TET to verify the achieved performance.

Traffic Engineering Tool

One of the most important components of TEAM is the TET, which is responsible for resource and route management, making the decisions related to network management tasks. The TET management tasks include resource management (LSP setup/dimensioning, LSP capacity allocation, LSP preemption) and route management (LSP routing), as shown in Fig. 1. The TET makes use of the other two TEAM tools, MPET and ST, in order to optimize the management of the network domain. To illustrate the interrelations of the listed problems for MPLS network management, consider the scenario where the network planning phase has provided an initial topology of the MPLS networks that will be adapted to the changing traffic demands. Possible events could be arrival of a request for LSP setup based on SLS agreements or arrival of a bandwidth request. The first event can be handled by combined use of the two methods in order: LSP routing and LSP preemption. The LSP routing aims to find the route on the physical network over which the LSP will be routed. LSP preemption decides if any existing LSPs can be preempted on the route to make way for the new LSP if there is not enough available bandwidth. The second event, arrival of bandwidth request triggers LSP setup and dimensioning, which may in turn trigger the LSP creation steps of routing and preemption.

Individual algorithms for each of the management tasks performed by the TET have been coded and tested through simulation and experiments in the BWN-Lab IP QoS testbed. In [12] simulation results for the LSP setup/dimensioning policy are given. Results for the preemption policy are given in [13]. In [14] the simulation results for performance of the LSP routing algorithm are evaluated. Next, we provide details of TEAM's managing actions performed by the TET, namely resource and route management.

LSP Setup/Dimensioning and Capacity Allocation — An important aspect in designing an MPLS network is to determine an initial topology and adapt it to the traffic load. A topology change in an MPLS network occurs when a new LSP is created between two nodes. LSP creation involves determining the route of the LSP and the according resource allocation to the path. The objective of our algorithm is to determine when an LSP should be created and how often it should be redimensioned.

We introduce a new decision policy in [12] that provides an online design for a MPLS network depending on current traffic load. The proposed policy is a traffic-driven approach and balances the signaling and switching costs. By increasing the number of LSPs in a network, the signaling costs increase while the switching costs decrease. In our policy, LSPs are set up or torn down depending on the actual traffic demand. Furthermore, since a given traffic load may change with rapid variations depending on time, the new policy also performs filtering in order to avoid oscillations that may occur with variable traffic. When a new bandwidth request $b_m(i,j)$ arrives between routers i and j in the MPLS network, the existence of a direct LSP between i and j is checked initially. For direct LSP between i and j, the available capacity $A(i,j)$ is then compared to the request $b_m(i,j)$. If $A(i,j) > b_m(i,j)$, the requested bandwidth is allocated on that LSP and the available capacity is reduced accordingly. Otherwise, the LSP capacity $C(i,j)$ can be increased subject to bandwidth constraints in order to satisfy the bandwidth request. If no direct LSP between i and j exists, we need to decide whether to set up a new LSP and its according capacity $C(i,j)$.

To make the decisions, we define an incremental cost function $W(s_m,a)$ associated with the system when a bandwidth request $b_m(i,j)$ arrives at time instant t_m, the network state is s_m, and action a is taken. It is the sum of three cost components: the bandwidth cost $W_b(s_m,a)$, switching cost $W_{sw}(s_m,a)$, and signaling cost $W_{sign}(s_m,a)$. The bandwidth cost $W_b(s_m,a)$ depends linearly on the required bandwidth and the number of hops $h(i,j)$ in the path over which the LSP is routed. The switching cost $W_{sw}(s_m,a)$ depends linearly on the number of switching operations in IP or MPLS mode and the switched bandwidth. The signaling and control cost $W_{sign}(s_m,a)$ is incurred when a new LSP is set up or redimensioned. We consider that this cost depends linearly on the number of hops $h(i,j)$ in the path over which the LSP is set up plus a constant component to take into account the notification of the new LSP setup to the network. In our assumptions redimensioning implies the same signaling cost as setting up a new LSP. The optimal LSP setup and teardown policy is derived in [12] by comparing the costs involved with the various decisions. The optimal policy is obtained by minimizing the cost function discounted over an infinite horizon. This optimization can be achieved by using Markov decision processes. The optimal policy has a threshold structure. However, it can be computationally expensive to precalculate and store the decision policy for each network state since the number of states grows exponentially with network size. So we also propose a suboptimal policy that is easier to implement and still maintains the threshold structure of the optimal policy.

LSP Preemption — When an LSP is created, a preemption attribute is assigned to it. The preemption attribute determines whether an LSP

A topology change in an MPLS network occurs when a new LSP is created between two nodes. LSP creation involves determining the route of the LSP and the according resource allocation to the path. The objective of our algorithm is to determine when an LSP should be created and how often it should be redimensioned.

A new LSP setup request has two important parameters: bandwidth and preemption priority. In order to minimize wastage, the set of LSPs to be preempted can be selected by optimizing an objective function that represents these two parameters, and the number of LSPs to be preempted.

with a certain priority attribute can preempt another LSP with a lower priority attribute from a given path, when there is a competition for available resources. The preempted LSP may then be rerouted. Preemption can be used to ensure that high-priority LSPs can always be routed through relatively favorable paths in a DiffServ environment. In the same context, preemption can be used to implement various prioritized access policies as well as restoration policies following fault events [2].

In a case in which preemption will occur, a preemption policy should be activated to find the preemptable LSPs with lower preemption priorities. Running preemption experiments using Cisco routers (7204VXR and 7505, OS v. 12.2.1), we could conclude that the preempted LSPs were always the ones with the lowest priority, even when the bandwidth allocated was much larger than that required for the new LSP. This policy would result in high bandwidth wastage for cases in which rerouting is not allowed. An LSP with a large bandwidth share might be preempted to give room to a higher-priority LSP that requires much lower bandwidth.

A new LSP setup request has two important parameters: bandwidth and preemption priority. In order to minimize wastage, the set of LSPs to be preempted can be selected by optimizing an objective function that represents these two parameters and the number of LSPs to be preempted. More specifically, the objective function could be any or a combination of the following:

- Preempt the connections that have the least priority (preemption priority)
- Preempt the least number of LSPs
- Preempt the least amount of bandwidth that still satisfies the request

After the preemption selection phase is finished, the selected LSPs must be torn down (and possibly rerouted), releasing the reserved bandwidth. The new LSP is established using the currently available bandwidth.

Our preemption policy [13] combines the three objectives described above in its objective function (weighted sum of the three criteria). The policy can be adjusted by the service provider in order to stress the desired criteria. No particular criterion order is enforced. Moreover, our preemption policy is complemented by an adaptive rate scheme. The resulting policy reduces the number of preempted LSPs by adjusting the rate of selected low-priority LSPs that can afford to have their rate reduced in order to accommodate a higher-priority request. This approach minimizes service disruption and rerouting decision and signaling.

When an LSP is to be set up on a path that does not have enough reservable bandwidth, first the algorithm checks whether there is enough preemptable bandwidth in order to make room for the new LSP. If the answer is yes, the weighted preemption policy selects a single LSP for preemption. This procedure is repeated until enough LSPs have been preempted so that enough free capacity is available for the new LSP.

Weighted preemption policy: Consider a request for a new LSP setup with bandwidth b and setup preemption priority p. When preemption is needed, due to lack of available resources, the preemptable LSPs will be chosen among the ones with lower preemption priority (higher numerical value) in order to fit $r = b - Abw$. The constant r represents the actual bandwidth that needs to be preempted (the requested, b, minus the available bandwidth on the link, Abw).

In order to represent a cost for the preemption priority of LSP l, an associated cost $y(l)$ inversely related to the preemption priority $p(l)$ is defined. The bandwidth of LSP l is represented by $b(l)$. To have the widest choice on the overall objective each service provider needs to achieve, a parameter $H(l)$ is defined for each LSP l. $H(l)$ is given as the sum of three components: $\alpha y(l)$, β and $\gamma(b(l) - r)^2$. $\alpha y(l)$ represents the cost of preempting LSP l, β represents the choice of minimizing the number of LSPs to be preempted in order to fit the request r, and $\gamma(b(l) - r)^2$ penalizes a choice of an LSP to be preempted that would result in high bandwidth wastage. Coefficients α, β, and γ are suitable weights that can be configured in order to stress the importance of each component in $H(l)$.

$H(l)$ is calculated for each LSP l. The LSPs to be preempted are chosen as the ones with smaller H that add enough bandwidth to accommodate r. If the value of H is equal for more than one LSP, these LSPs are chosen in increasing order of $b(l)$. More details on the algorithm are given in [13]. The algorithm's output contains the information about which LSPs are to be preempted and the amount of bandwidth preempted.

The decision to preempt an LSP may cause other preemptions in the network. This is called the *preemption cascading effect*, and different cascading levels may be achieved by the preemption of a single LSP. The cascading levels are defined in the following manner: when an LSP is preempted and rerouted without causing any further preemption, the cascading is said to be of level 0. However, when a preempted LSP is rerouted and, in order to be established in the new route, also causes the preemption of other LSPs, the cascading is said to be of level 1 and so on.

LSP Routing — Route management deals with deciding the routes for LSPs over a physical network and for bandwidth requests over an MPLS network. It is triggered by the arrival of either an LSP setup request or a bandwidth reservation request in MPLS networks.

In [14] we propose the new routing algorithm Stochastic Performance Comparison Routing Algorithm (SPeCRA). SPeCRA attempts to adaptively choose the best routing algorithm from a number of candidate algorithms, each of which may be suited to a different traffic mix.

SPeCRA behaves as a homogeneous Markov chain where the optimal routing scheme is a state of the chain that is visited at the steady state with a certain probability. Aiming to reduce the chance that we could leave the state due to estimate error, a noise filter was introduced in

SPeCRA. A state variable Q reduces the changes from "good" to "bad" routing schemes by acting as a threshold for the decision of switching between two routing schemes. The algorithm is detailed below.

Initial data: a set of possible routing schemes; probability matrix $R(a,b)$, which represents the probability of choosing b as a candidate routing scheme when the current routing scheme is a; an initial routing scheme and a control interval T. The state variable used for filtering purposes is initialized as $Q = 0$.

At every iteration a subset of routing schemes is chosen according to $R(a,b)$; all LSP setup requests arrived and ended during the current interval of time are recorded. and an estimate of the LSP setup rejection probability for each routing scheme is calculated. The routing scheme with smaller rejection probability is then selected as a candidate for comparison with the current routing scheme. The choice of switching routing scheme is made as follows:

If the rejection probability $p_B(\theta)$ of the current scheme θ subtracted by the state variable Q is larger than the rejection probability $p_B(\theta')$ of the candidate routing scheme θ′, the algorithm replaces the current routing scheme with the candidate. If the rejection probability of the current scheme subtracted by Q is smaller than the candidate scheme's rejection probability, Q is updated by adding to its current value, the probability of rejection of the candidate minus the probability of rejection of the current routing scheme divided by two. No routing scheme update is performed, and the current algorithm continues to run until a new decision is made at the next control interval time. Under very conservative assumptions, it is possible to prove that the estimate of the order between θ and θ′ in terms of LSP setup rejection probability is more robust than the estimate of the cardinal values of the two LSP setup rejection probabilities. In fact, if there are N independent estimates of $p_B(\theta)$ and $p_B(\theta')$ taken on N different and nonoverlapping intervals, the convergence rate of the estimated order to the real order is an exponential function of N and much larger than the convergence rate of the cardinal estimates, whose variance approaches 0 with $1/N$.

The simulation results presented in [14] show that adaptively choosing between many different fairly simple algorithms results in better performance than using a single more complicated computationally expensive algorithm.

Measurement and Performance Evaluation Tool

The MPET is used to measure the network state to be reported to the TET and also to check if the TET decisions that have been implemented have the intended effect on the network. Currently, we deem the available bandwidth as the most important state variable in the network that provides a sufficient glimpse of the network. Thus, the MPET implementation measures the available bandwidth of the network links reliably.

Our available bandwidth measurement approach [15] used in the MPET is based on the use of MRTG where each router in the domain is inquired through Simple Network Management Protocol (SNMP) to obtain information about the available bandwidth on each of its interfaces. The most accurate approach is to collect information from all possible sources at the highest possible frequency allowed by the management information base (MIB) update interval constraints. However, this approach can be very expensive in terms of signaling and data storage. Furthermore, it can be redundant to have so much information. We use a multistep linear predictor to predict the future values of link utilization based on past utilization values. We propose to dynamically adapt the length of the prediction interval and the number of past samples based on the prediction error. Let us denote the number of past measurements in prediction p and the number of future samples that can be reliably predicted h. We start with initial values p_0 and h_0, respectively, which are adapted to the prediction performance in an additive increase multiplicative decrease manner. We bound h by h_{min} on the lower side because small values of h imply frequent recomputation of the regression coefficients. Also, we limit p by p_{max} on the upper side because large values of p increase the computational cost of the regression. Once the estimates of the link utilization are obtained, we can use either of the following two methods to obtain the link available bandwidth. These methods aim to obtain a single representative value valid for the whole interval. The two methods provide different estimates based on the conservativeness requirements of the network operator. The more conservative of the two methods estimates the available bandwidth for the duration of h samples as the difference between the link capacity and the maximum estimated link utilization for the duration. The less conservative method estimates the available bandwidth as the difference between the link capacity and an effective bandwidth metric obtained for the duration of h samples.

Simulation Tool

The ST is a comprehensive C code that implements each of the policies in use by the TET. In order to help TEAM make optimal decisions, the TET may use the ST to consolidate the decisions made. The ST will simulate a network with the current state of the managed network and apply the decision of the TET to verify the achieved performance. The TET management tasks that can be simulated by ST include LSP setup/dimensioning, LSP preemption, LSP capacity allocation, LSP redimensioning, and LSP routing.

The TET implementation, described next, is such that the software can also work as the ST. This is possible because the software is written to respond in real time to the network events for medium-sized networks. Real-time responsiveness is essential for the TEAM software since network events have to be handled when they occur.

The TET implementation is such that the software can also work as the simulation tool. This is possible because the software is written to respond in real-time to the network events for medium sized networks.

TEAM is structured to be composed of two parts: the server and the client. The server can run at a high performance station in order to keep track of all the information of the network. The client connects to the server and sends commands using a user interface protocol.

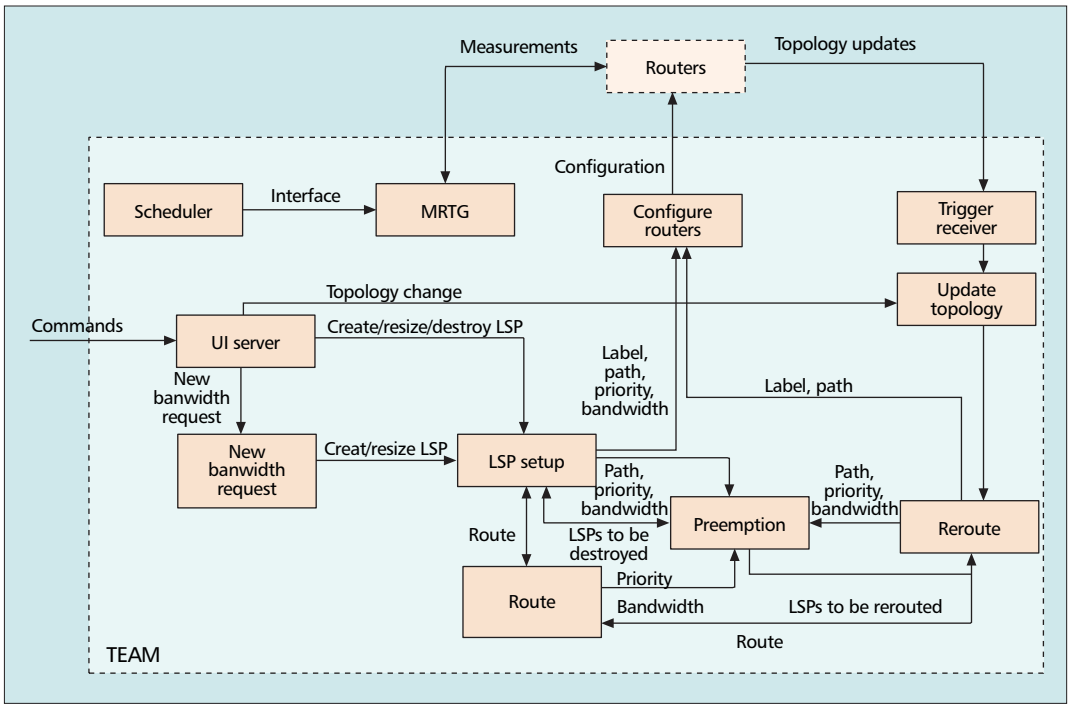

Figure 2. *TEAM top-level design.*

TEAM Implementation

We have a full-fledged next-generation Internet routers physical testbed in BWN-Lab at Georgia Tech, equipped with DiffServ capable routers and switches manufactured by Cisco. In particular, we have a Cisco 7500 router with a Gigabit Ethernet card and a layer 3 switch Catalyst 6500 with an enhanced Gigabit Ethernet card, among other routers and switches. These routers and switches are widely deployed in the backbones of current high-speed networks. All of our routers support MPLS and a variety of QoS technologies such as RSVP and DiffServ. Currently all devices are SNMP enabled, and different network measurement tools like MRTG and Netflow have been enabled and tested. During the analysis of MRTG, a new improved version of the tool was developed by our group, called MRTG++, which allows managers to monitor traffic at up to 10 s intervals rather than the original 5 min sampling of MRTG, providing more fine-grained detail about the state of the network. Our testbed is connected via an OC3 link to Abilene, the advanced backbone network of Internet2 society that supports development and deployment of new applications. We have performed end-to-end QoS performance experiments with NASA Goddard in Maryland. The objective of the experiments is to study the advantages and disadvantages of using DiffServ in a heterogeneous traffic environment. The traffic under study is generated from voice, video, and data sources. The testbed has been used as the platform to implement and test the operation of TEAM.

The implementation of TEAM must be able to:

- Receive a request for bandwidth reservation or LSP setup from the user
- Implement the proposed algorithms to obtain good performance for routing, LSP setup, and preemption
- Send commands and configure the testbed to create LSPs and route the traffic on the LSPs
- Reach a decision in a timely manner to handle a large domain with 200 routers and about 20,000 LSPs

The TEAM tool has been implemented to run on a computer with the Linux OS. It was successfully tested on RedHat 7.3, running kernel 2.4.18 on a Pentium III at 800 MHz with 256 Mbytes RAM and 512 Mbytes swap space. The first version requires a TFTP server to upload the configuration to the routers. SNMP is required to ensure communication between the program and the routers. TEAM uses the net-snmp library for the communication. Version 3 of SNMP is recommended to ensure secure transmission of passwords. In order to process bandwidth measurements, RRDTool and MRTG are required. Also, the GNU Scientific Library is required for matrix manipulation. The REA library is used for computing k-shortest paths. The program was successfully tested on a 40-node network and 20,000 LSPs on our Pentium III computer. The top-level design of TEAM is shown in Fig. 2.

Each LSP record takes about 100 bytes in addition to the path information. It takes 20 bytes for each hop in the path. The network topology information takes about 24 bytes/node and 40 bytes/link. The LSP setup decision process takes $O(PN\log N)$ time, where P is the average path length and N is the average number of LSPs in a link.

TEAM is structured to be composed of two parts: the server and the client. The server can run at a high-performance station in order to

keep track of all the information of the network. The client connects to the server and sends commands using a user interface protocol. Examples of commands include the creation and destruction of LSPs, and requesting the topology of the network.

The Server

The server can be executed in two modes. The first one is the TET mode, in which commands are received from the user and configurations are sent to the routers after a decision is made by the program. The second one is the ST mode, in which the MPLS domain is simulated in order to study the network behavior when a decision is applied.

When the server is run in TET mode, it stays in the background ready to receive commands from the client. It performs the following steps:

- Load the system-wide configuration file
- Obtain the network topology
- Obtain the initial LSP topology
- Prompt for user's command

At any time the program gives the option to print the current topology of the network, the LSP database, and the request database. The topology shows each node and all the links that originate from it. For each link the capacity and available bandwidth are shown. The LSP database lists all LSPs TEAM is maintaining. For each LSP the label, source, interface number, destination, priority, capacity, and path are shown. Finally, the request database shows similar output. It prints all the requests being served by TEAM at the moment. For each request, the identification, source, destination, priority, and bandwidth are shown. In addition, it shows the label of the LSP serving the request.

TEAM can send commands to the routers using SNMP or telnet. We adopted SNMPv3 in order to keep communications secure. Unfortunately, current MIBs are read-only and do not allow the tool to establish LSPs directly. TEAM instructs the router to retrieve a configuration file from a TFTP server and merge it into the current setting. Although the configuration is unprotected, passwords are never sent in clear text across the network.

In ST mode the server performs the same initial steps as in TET mode and then loads the command file, reading one line at a time to simulate the traffic. At the end of the simulation, the tool gives the option to show the topology, the LSP database, and the request database in the same way as before.

The Client

The client is the program used to send commands to the server. It can be written and implemented in any language as long as a specific user interface protocol is used. The protocol exports the basic functionality to control the MPLS domain.

The program presents a menu with each command for a choice of user operations. For example, in order to create an LSP, the program asks from which node the LSP is being originated, the destination, priority, and bandwidth. If the path is already defined, just type in each hop of the path. In order to facilitate selection of the path, the client shows valid choices for each hop in the path. When the LSP is created by the server, the client is notified.

Similar behavior occurs for the establishment of a request. The client asks for the source, destination, priority, and bandwidth, and TEAM creates the request. The client can also display the topology of the network.

Input Files

TEAM loads some information from a set of different files. We describe the format of these files here. All these files are located in the input directory, and their location can be modified in the configuration file. All fields are separated by a space.

The topology file contains the initial topology of the network. The initial LSP file contains the list of LSPs that will be in the database at the start of execution of the program. The command file is used to send commands to the system. Before each entry, a sequence number (or time component) should be included. This number is used to identify events in the output files. It is the time component of the simulation. The configuration file controls how the program should behave. It contains ON/OFF switches for each feature of the system.

In ST mode the server performs the same initial steps as in TET mode and then loads the command file, reading one line at a time, to simulate the traffic. At the end of the simulation, the tool gives the option to show the topology, the LSP database, and the request database in the same way as before.

Experimental Results

In this section we demonstrate the performance of TEAM and show its operation by combining each functionality mentioned before. Comparison of each TEAM functionality with current state-of-the-art equivalent techniques has been performed and can be found in our previous work. Since to the best of our knowledge there are no other comprehensive network managers such as TEAM with such a diverse set of functionalities, we compare TEAM to traditional Internet managers.

The following experimental results are obtained by simulating a network consisting of 40 nodes and 64 links, each with 600 Mb/s capacity (OC-48). This network topology is based on the backbone topology of a well-known Internet service provider. We chose to run the experiments on this simulated network to obtain a more realistic scenario where the operation of TEAM is visible. The traffic in the network consists of aggregated bandwidth requests between node pairs having two possible priorities. Priority level 0 is the lower priority, which can be preempted by higher-priority requests of level 1. We model these traffic requests with Poisson process arrivals and exponential durations. We divide the simulations into two broad traffic scenarios to represent significant conditions. These scenarios are characterized by different traffic loads in the network. We consider generalized medium and focused high traffic loads to bring out the contrast in traffic conditions, and observe the effects on network performance and the different actions taken by TEAM. We define the generalized medium traffic load as a traffic matrix with equal values as the elements. On the other hand, the focused high load scenario is represented by a matrix where elements corresponding to node pairs on the opposite extremes of

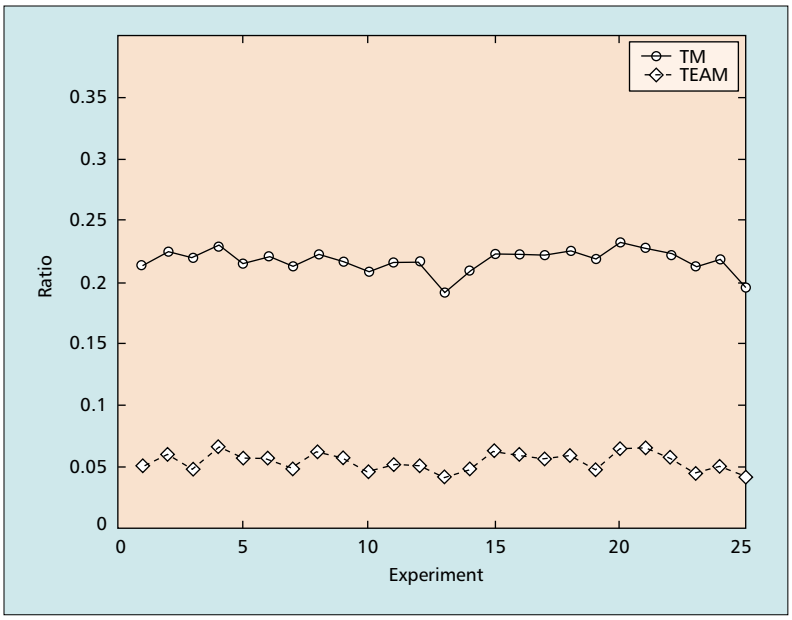

Figure 3. *Rejection ratio.*

the network have twice the value of other node pairs.

The routing algorithm employed by TEAM, SPeCRA, uses many well-known algorithms for performance comparison. This set can be modified depending on network requirements. In the following experiments we used shortest path, widest path, and maximum-utility-based routing algorithms.

To evaluate the performance of TEAM as a network manager, we consider both network performance and the complexity associated with TEAM. In particular, for performance we consider the rejection of requests, load distribution, cost of network measurements, and cost of providing service to the requests. The complexity is measured by the number of actions performed by TEAM and the level of the cascading effect of these actions. We compare these metrics for a network managed by TEAM and a network managed by a traditional manager (TM), such as the current Internet. We assume that in traditional network management, the MPLS network topology is static and the same as the physical network. In this case the shortest path routing algorithm is used for LSP establishment, there is no LSP preemption, and there are no online network measurements for adaptive network management.

The Generalized Medium Traffic Load

By running experiments with the generalized medium traffic load, we observed that the LSP setup and LSP routing (SPeCRA) techniques played a major role compared to LSP preemption. In Fig. 3 we show the rejection ratio for requests with and without TEAM. As we can see, the rejection is 75 percent lower when TEAM is managing the network. TEAM is able to achieve lower rejection due to more efficient load balancing than traditional network management.

Next, we demonstrate the efficiency of TEAM by comparing the performance with respect to the minimum and average available bandwidths for all the links in the network. In Fig. 4a we plot the minimum available bandwidth.

We see that in the absence of TEAM, the network links have lower minimum available bandwidth than when TEAM is active. This is attributed to the fact that the traffic load is evenly distributed in the network using TEAM. In Fig. 4b we show the average available bandwidth in the network. We see that the values when TEAM is employed are lower than those when TM is employed. This gives the false impression that the performance of TEAM in this case is worse than that of the TM. However, this is not correct, and it is still due to poor load balancing by the traditional network manager. In fact, when the load is not well distributed, few links in the network are overloaded and the rejection probability becomes higher. This observation is corroborated by the high rejection ratio reported in Fig. 3. Summarizing, the average available bandwidth in the network is lower using TEAM because TEAM is allowing more traffic to be carried.

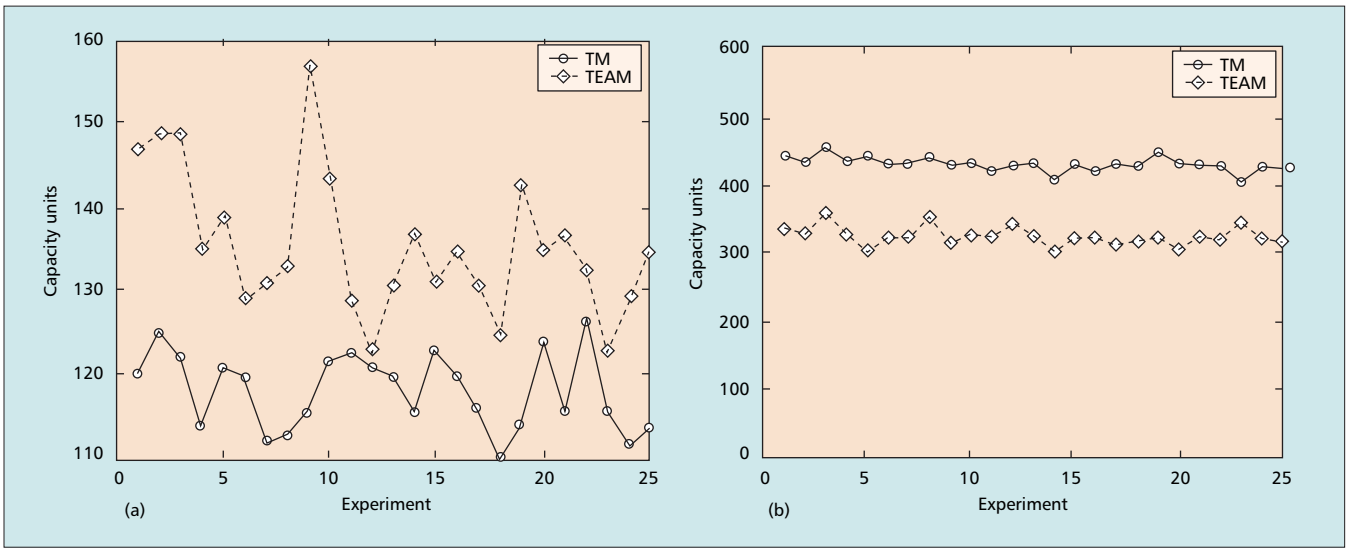

Figure 4. *Available bandwidth: a) minimum; b) average.*

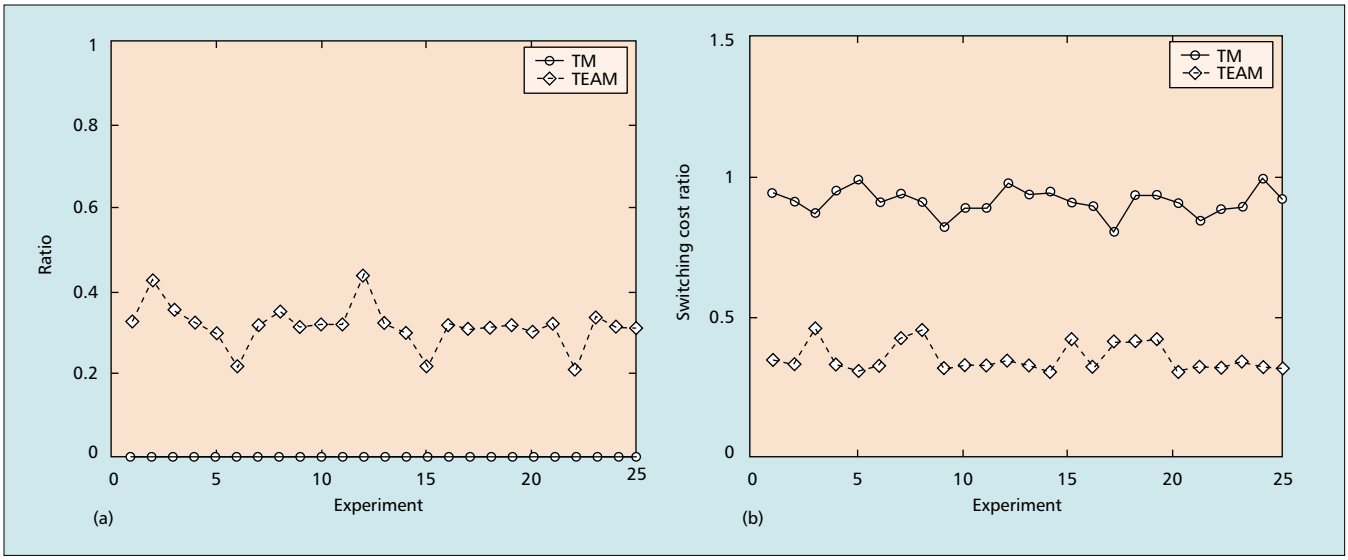

Figure 5. *Cost: a) network measurements; b) service provisioning.*

In Fig. 5a we plot the cost of performing network measurements. This cost is assumed to be linearly proportional to the number of available bandwidth measurements in the network. From Fig. 5a we see that around 30 percent of TEAM's actions (like LSP setup, routing, and preemption) required online measurement, compared to the TM where there is no need for network measurement since provisioning is based on service level agreements and nominal reservations. This null cost is depicted in the figure. Note that the measurement overhead has been limited to such low values by the filtering mechanisms in the individual TEAM techniques, and is offset by the lower rejection of requests and consequently higher revenue. In Fig. 5b we plot the normalized costs of providing service in the network . It is mainly representative of the traffic switching cost that can be performed in MPLS or IP mode. As is well known, it is less expensive to switch traffic in MPLS mode than in IP mode due to the simpler forwarding mechanism of MPLS routers. The more LSPs are created in the network, the lower the overall switching cost for the traffic. However, the lower switching cost has to be balanced with high signaling cost attributed to each LSP setup/redimension. Thus, TEAM provides an optimal number of LSPs in the network by balancing the switching and signaling costs. This optimal topology depends on the offered traffic, and in this generalized medium traffic scenario it is not as connected as a fully meshed topology. For this optimal topology, the switching cost is approximately 40 percent of that related to a static network topology. This static topology has the minimum number of LSPs as it corresponds to the physical topology.

Next, we consider the TEAM operational load; that is, the number of actions performed by TEAM to handle the incoming bandwidth requests. We see that 19 percent of the requests lead to activation of the LSP setup/redimensioning procedure, whereas only 0.5 percent of the requests were provisioned after preempting a pre-existing LSP. Most of the LSPs were routed using the shortest path routing algorithm because of the medium traffic load in the network. However, TEAM chooses other routing algorithms like widest path and maximum utility to achieve better load balancing when the shortest path route is overloaded.

For this traffic load, the cascading level is always 0 as all preempted LSPs are reestablished without causing any further preemptions. Thus, the cascading effects of preempting LSPs, which are undesirable, are absent in this medium traffic load scenario.

The Focused High Traffic Load

By running experiments with the focused high traffic load, we observed that the LSP preemption and LSP routing (SPeCRA) techniques played a more significant role than LSP setup and capacity allocation. In Fig. 6 we show the rejection ratio for requests with and without TEAM for priority levels 0 and 1. In the absence of TEAM, we observe around 30 and 15 percent rejection for low- and high-priority requests, respectively. When TEAM is deployed in the network, overall rejection is still 75 percent lower than without TEAM. However, the rejection of high-priority requests is reduced tenfold compared to a threefold decrease in low-priority rejection. This considerable decrease in high-priority rejection is due to the combined effect of load balancing and preemption introduced by TEAM. The rejection of low-priority traffic is also reduced, but not at the same scale because only load balancing is active, without preemption.

Since preemption plays a significant role in this traffic scenario, we present the effects of various preemption policies on network performance in terms of cascading. In high traffic load scenarios, when preemption is significant the cascading effects should be minimized because cascading is undesirable. We studied the cascading effects for the weighted preemption policy with different parameter choices. These results are compared to a random selection of LSPs for

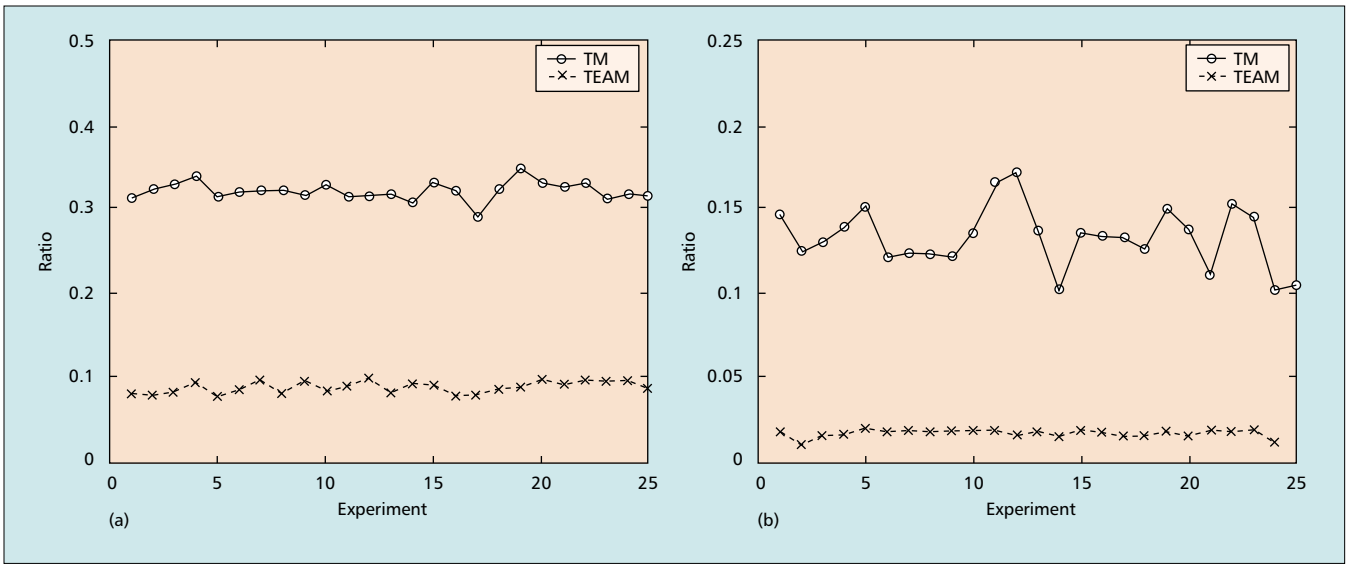

Figure 6. *Rejection ratio: a) priority 0; b) priority 1.*

preemption (RND), a commercial preemption policy (P), and the case without preemption (noTEAM).

The set of preemption policies we studied can be summarized as:

- PB: Weighted preemption policy with $\beta = 0$ (priority is the most important factor and selects the LSPs that minimize waste of bandwidth).
- BN: Weighted preemption policy with $\alpha = 0$ (bandwidth is the most important factor and selects the minimum number of LSPs to fit the new request).
- PN: Weighted preemption policy with $\gamma = 0$ (priority is the most important factor and selects the minimum number of LSPs to fit the new request).
- P: Only LSP priority is considered in the selection.
- RND: LSPs are randomly selected for preemption.
- noTEAM: The LSPs are not preempted.

When preemption policy PB was in use, fewer LSPs were preempted and it caused less cascading. This was expected because the algorithm will preempt LSPs with low priority, which will not propagate the preemption to other levels. Consequently, preempted LSPs were destroyed (not able to be rerouted). When BN was used, cascading was higher than the first case due to the fact that LSPs with higher priority could be preempted. The number of preempted LSPs was also higher as a consequence. However, the wasted bandwidth is smaller. Policy PN led to a smaller number of preempted LSPs. This policy preempts larger LSPs (higher wasted bandwidth) with low priority. Therefore, it makes room for more connections to be set up (improving the acceptance rate) and minimizes the number of preempted LSPs. Compared to PB, policy P resulted in a higher number of preempted LSPs and a higher rate of LSPs destroyed due to preemption. The cascading level was the same. However, the wasted bandwidth was much higher. In the RND (random) policy, the cascading effect was a lot stronger due to the preemption of LSPs with higher priority, which could then preempt other LSPs. The wasted bandwidth is also much higher. When preemption was not allowed (noTEAM), the cascading effects are obviously not present. The results show that when preemption is based on priority, cascading is not critical since the preempted LSPs will not be able to propagate preemption much further. When bandwidth is considered, fewer LSPs are preempted in each link and the wasted bandwidth is low. The policy PB seems to combine all these features, yielding the best results.

Next, we consider the TEAM operational load in this focused high traffic scenario. The PB preemption policy was implemented. We see that 35 percent of the requests led to activation of the LSP setup/redimensioning procedure, and 10 percent of the requests caused preemption of pre-existing LSPs. In this scenario SPeCRA chooses the widest path and maximum utility routing algorithms more often than the shortest path algorithm to achieve the desired load balancing.

Conclusions

Efficient and automated management of MPLS/DiffServ networks is an open issue. The goals of a network manager are QoS provisioning, efficient resource usage, and reduced risk of congestions in the network. These objectives should be achieved under variable and unpredictable traffic conditions which are characteristic of the current Internet. Toward this end, we have proposed and implemented TEAM, a novel network manager. TEAM performs efficient resource and route management in the network to achieve the desired objectives by using online measurements of network state and reacting instantly to network changes. We have shown the implementation details of TEAM along with experimental results to show how network performance is improved using TEAM, at the expense of limited increases in computational and control efforts.

ACKNOWLEDGMENTS

The authors would like to thank Leonardo Chen for implementation of the TEAM software. Special thanks for Jeff Smith and Agatino Sciuto from NASA GSFC for their support.

REFERENCES

[1] D. Awduche *et al.*, "Overview and Principles of Internet Traffic Engineering," IETF RFC 3272, May 2002.
[2] D. O. Awduche *et al.*, "Requirements for Traffic Engineering over MPLS," IETF RFC 2702, Sept. 1999.
[3] F. Le Faucheur *et al.*, "Multi-Protocol Label Switching (MPLS) Support of Differentiated Services," IETF RFC 3270, May 2002.
[4] R. Callon, "Predictions for the Core of the Network," *IEEE Internet Comp.*, vol. 4, no. 1, Jan./Feb. 2000, pp. 60–61.
[5] Y. Yemini, A. V. Konstantinou, and D. Florissi, "NESTOR: An Architecture for Network Self-Management and Organization," *IEEE JSAC*, vol. 18, no. 5, May 2000, pp. 758–66.
[6] C. Tsarouchis *et al.*, "A Policy-Based Management Architecture for Active and Programmable Networks," *IEEE Network*, vol. 17, May/June 2003, pp. 22–28.
[7] P. Aukia *et al.*, "RATES: A Server for MPLS Traffic Engineering," *IEEE Network*, vol. 14, Mar./Apr. 2000, pp. 34–41.
[8] K. Kar, M. Kodialam, and T. V. Lakshman, "Minimum Interference Routing of Bandwidth Guaranteed Tunnels with MPLS Traffic Engineering Application," *IEEE JSAC*, vol. 18, no.12, Dec. 2000, pp. 2566–79.
[9] A. Elwalid *et al.*, "MATE: MPLS Adaptive Traffic Engineering," *Proc. IEEE INFOCOM '01*, Anchorage, AK, Apr. 2001, pp. 1300–09.
[10] E. Mykoniati *et al.*, "Admission Control for Providing QoS in IP DiffServ Networks: The TEQUILA Approach," *IEEE Commun. Mag.*, vol. 41, no. 1, Jan. 2003, pp. 38–46.
[11] X. Xiao *et al.*, "A Practical Approach for Providing QoS in the Internet Backbone," *IEEE Commun. Mag.*, vol. 40, Dec. 2002, pp. 56–62.
[12] T. Anjali *et al.*, "Optimal Policy for LSP Setup in MPLS Networks," *Comp. Networks*, vol. 39, no. 2, June 2002, pp. 165–83.
[13] J. C. de Oliveira *et al.*, "A New Preemption Policy for DiffServ-Aware Traffic Engineering toMinimize Rerouting," *Proc. IEEE INFOCOM '02*, New York, NY, June 2002, pp. 695–704.
[14] J. C. de Oliveira, F. Martinelli, and C. Scoglio, "SPeCRA: A Stochastic Performance Comparison Routing Algorithm for LSP Setup in MPLS Networks," *Proc. IEEE GLOBECOM '02*, Taipei, Taiwan, Nov. 2002, pp. 2190–94.
[15] T. Anjali *et al.*, "ABEst: An Available Bandwidth Estimator within an Autonomous System," *Proc. IEEE GLOBECOM '02*, Taipei, Taiwan, Nov. 2002, pp. 2360–64.

BIOGRAPHIES

CATERINA SCOGLIO (caterina@ece.gatech.edu) received a Dr. Ing. degree in electronics engineering from the University of Rome "La Sapienza," Italy, summa cum laude, in May 1987. She received a post-graduate degree in mathematical theory and methods for system analysis and control from the University of Rome "La Sapienza" , in November 1988. From June 1987 to June 2000 she was with Fondazione Ugo Bordoni, Rome, where she was a research scientist in the TLC Network Department, Network Planning Group. Since September 2000 she has been with the Broadband and Wireless Networking Laboratory of Georgia Tech as a research engineer. Her research interests include optimal design and management of multiservice networks.

TRICHA ANJALI (tricha@ece.iit.edu) received her (integrated) M.Tech. in electrical engineering from the Indian Institute of Technology, Bombay, in 1998, and her Ph.D. degree in electrical and computer engineering from Georgia Tech in May 2004. Currently, she is an assistant professor in the Electrical and Computer Engineering Department at Illinois Institute of Technology. Her research interests are in QoS issues in the next-generation Internet.

JAUDELICE CAVALCANTE DE OLIVEIRA (jau@ece.drexel.edu) received her B.S.E.E. degree from Universidade Federal do Ceara, Ceara, Brazil, in December 1995. She received her M.S.E.E. degree from Universidade Estadual de Campinas, Sao Paulo, Brazil, in February 1998, and her Ph.D. degree in electrical and computer engineering from Georgia Tech in May 2003. She joined Drexel University as an assistant professor in July 2003. Her research interests include QoS provisioning in the future Internet, traffic engineering strategies for MPLS networks, and the design of solutions for managing heterogeneous and large computer networks.

IAN F. AKYILDIZ [F'95] (ian@ece.gatech.edu) is the Ken Byers Distinguished Chair Professor with the School of Electrical and Computer Engineering, Georgia Tech, and director of the Broadband and Wireless Networking Laboratory. He is Editor-in-Chief of *Computer Networks* (Elsevier) and *Ad Hoc Networks Journal* (Elsevier). He is an ACM Fellow (1996). He received the ACM Outstanding Distinguished Lecturer Award for 1994, the 1997 IEEE Leonard G. Abraham Prize award (IEEE Communications Society), and the 2002 IEEE Harry M. Goode Memorial award (IEEE Computer Society) with the citation "for significant and pioneering contributions to advanced architectures and protocols for wireless and satellite networking." He also received the 2003 IEEE Best Tutorial paper award (IEEE Communications Society) and 2003 ACM SIGMOBILE award for his significant contributions to mobile computing and wireless networking. His current research interests are in wireless networks, sensor networks, and Interplanetary Internet.

GEORGE UHL (uhl@gsfc.nasa.gov) is lead engineer at NASA's Earth Science Data and Information System (ESDIS) Network Prototyping Laboratory. He directs network research and protoyping activities for ESDIS. Current areas of research include network QoS and end-to-end performance improvement.

Efficient and automated management of MPLS/DiffServ networks is a vast open issue. The goals of a manager are QoS provisioning, efficient resource usage, and reduced risk of congestion in the network. These objectives should be achieved in variable and unpredictable traffic conditions.