Network Prediction in a Policy-Based IP Network

Mandis Beigi and
Dinesh Verma
IBM T.J. Watson Research Center
30 SawmillRiver Road
Hawthorne, NY 10532
<mandis,dverma>@us.ibm.com

Abstract - In a Quality of Service (QoS) enabled network, the Internet Service Providers (ISPs) need to define the network requirements for the customers by defining a set of policies. Every time a new customer gets added or removed or when a customer's resource requirements change, the administrator needs to modify the policies installed on the routers/servers in the network. However, these modifications will affect the traffic flowing in the network and in turn might take away some of the existing customers' resources. Therefore, there is a need to predict how changing the policies will affect the performance of the existing traffic flows in the network. In this paper, we present a mechanism for predicting whether adding, removing or changing a policy will degrade the performance of the traffic flows belonging to the previous customers. The network administrator can use this information to better decide whether such modifications to the policies are desired.

I. INTRODUCTION

The fast growing use of the Internet and the limited amount of bandwidth and other network resources and services have required a need for providing Quality of Service (QoS). There have been two approaches for providing QoS, which then became standards in the IETF. One approach is called IntServ/RSVP, which uses a signaling mechanism for reserving the required bandwidth. This work was then followed by a newer approach called DiffServ, which solved the problem of scaling with RSVP. In either approach, the service classes are assigned to the users by the Internet Service Providers (ISPs) and corporate administrators by defining a set of policies [2]. A policy is defined for a user (or user group) accessing an application (or application group) running on a server (or group of servers). The policy maps a traffic aggregate to a class of service (COS). In a policy, the traffic is characterized by the 5-tuples in the IP header (i.e. source and destination IP address or a range of IP addresses, source and destination port numbers or a range of port numbers and the protocol). The user can define an IP address range by specifying a netmask or a prefix. A table of COS mappings defines how each class of service is to be implemented in the network. The COS mapping table defines limits on the amount of resources that each class of service should enforce, e.g. limits on the network bandwidth, or bounds on the total number of connections belonging to that class should be permitted at a server. A policy can have a life span, which defines the period when the policy is active. A life span has a beginning time and an ending time. In addition to the life span, the policy may be further restricted to be active only on specific days or dates by specifying a validity time field.

In a typical enterprise, the number of routers and servers runs into hundreds and thousands. Independent configuration of all these devices is tedious and can easily lead to missed devices and inconsistent configuration. Therefore, in order to easily manage the network configuration, a configuration management tool must be used. In a QoS enabled network, this tool allows the administrator to configure the policies from a central location [1]. One advantage of having a central management tool is that it allows the network administrator to generate and distribute consistent configurations for all routers/servers throughout the network. It also allows the manager to verify the validity of the defined policies using the knowledge of the network topology and the information about each node [1]. However, when adding, removing or changing policies in a network care must be taken not to degrade the performance of the traffic matching the existing policies. Therefore, there is a need for predicting the behavior of the network before these policies get modified. There has been some work on network traffic prediction by modeling techniques [3, 4, 5]. In [3] the prediction method uses archived quantitative performance data to create a quantitative/qualitative dynamic system representation. The representation captures the qualitative states of the network, qualitative input events and transitions among the states resulting from these events. The model shows the current system behavior and is used to predict future possible behaviors. [4] discusses a graphical predictor tool using an object-oriented simulation analysis. In [5] the network predictor discussed is built for QoS networks and uses a combination of network component models and probabilistic analysis. This predictor continuously monitors quality of service, compares it with the expected or desired values and projects future values. These methods, however, use past data and performance measurements to predict the network behavior in the future and they assume that the network configuration and the policies are unchanged. So far, there has been no work on prediction of the behavior of the traffic in a QoS enabled network when policies get modified. In the following sections of this paper, we present a method for designing such a network prediction tool and discuss some of the issues. We will also show some results from a simulation using the network simulation (NS2) tool with the

DiffServ extension and finally present some conclusions and future work in this area.

II. NETWORK PREDICTION METHOD

When defining the policies in a policy-based network, the administrator needs to make sure they cover all possible traffic flows. This means that there must be a policy defined for every traffic flow or traffic aggregate in the network. Each policy must have a priority associated with it, which is used for ordering of the policies. The policy with the lowest priority is the default policy, which maps all traffic not covered by the previous policies onto a default class of service, which normally is best effort. A traffic flow gets the service class associated with the first matching policy (i.e. with the highest priority) in the list.

Modification of a policy is one of three operations: addition, change and deletion of the policy. Since a policy exists for every possible traffic flow in a network, the addition and the deletion operations as well as the change operation, result in changing of a service class from one level to a new level. When the network manager adds a new policy, the prediction tool first checks whether the new policy requires more or less resources than the old policy. Assuming that a service class is defined as a traffic rate, a more resource requirement translates to a requirement for a higher traffic rate. If the new policy specifies a requirement for fewer resources, the prediction tool can safely predict that this policy will not hurt the performance of any other traffic flows that share the same resources. On the other hand, if the new policy requires more resources, the prediction tool performs more involved steps. It first determines whether the traffic matching the new policy shares resources with any other traffic in the network. To do so, it finds the access router pair(s) (i.e. the ingress and the egress routers) for the source/destination IP addresses specified in the new policy. It is assumed that the manager is aware of the network topology and can retrieve the access routers by performing a table lookup. Since in a policy, the source and the destination IP addresses can be specified as a range of addresses, there can be multiple ingress/egress access routers. The next step is to determine all the possible paths from the ingress to the egress access router(s) (i.e. find out all the intermediate nodes) by performing a 'traceroute' from the ingress access router(s) to the egress access router(s). Figure 1 shows an example of having multiple paths between a pair of source/destination address ranges. For example, let's assume that the new policy is defined as having premium service for all the traffic originating from subnet 1.1.0.0 (Site 1) and destined to subnet 1.2.0.0 (Site 2). After a table lookup, the predictor finds out that subnet 1.1.0.0 is located behind the access routers called ingress1 and ingress2 and subnet 1.2.0.0 is located behind the access routers called egress1 and egress2. Therefore, there are four possible paths connecting these two sites as listed below.

Path1: ingress1-router1-router2-router3-router4-egress1
Path2: ingress1-router1-router2-router3-router6-egress2
Path3: ingress2-router5-router2-router3-router4-egress1
Path4: ingress2-router5-router2-router3-router6-egress2

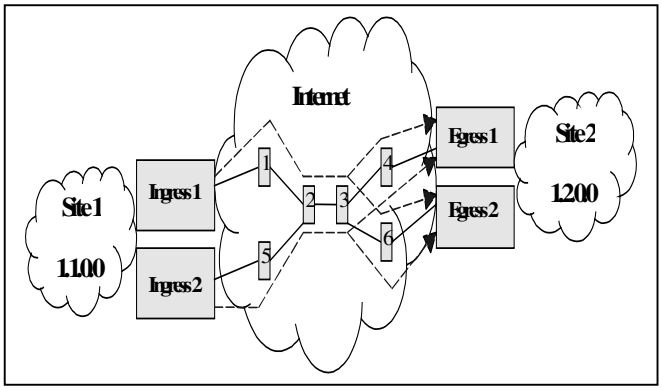

Figure 1

The predictor then measures the traffic utilization (ignoring the best effort traffic) on every link along these paths from the ingress router to the egress router. Assuming that all the link bandwidths in the network are known, the amount of free bandwidth on every link along these paths is calculated. All traffic measurements used by the predictor tool are retrieved from the network manager (i.e. a monitoring tool) which calculates a running average over a long period of time. The next step is to find out all other policies' traffic that will be affected by the new policy. This involves determining all the possible paths in the network that share any single link with the paths previously found and then finding the corresponding policies. Since there are too many such paths present in a large network, this operation is not efficient and it is almost impossible to perform. To simplify this operation, the predictor first calculates the extra bandwidth required by the new policy by subtracting its old bandwidth requirement from the new bandwidth requirement. It then compares this difference with the available bandwidth on every link along the path. It then only considers the links along this path whose available bandwidth is less than this difference. These links are so called the congested links. The predictor then finds all the existing policies that define traffic flows, which pass through these congested links. To do this, the network predictor dumps the shortest path graph of the network produced by the OSPF routing protocol. This graph is split into two separate trees, with the roots being the nodes on each side of the congested link and the leaves of the trees being the access routers in the network (see Figure 2). The predictor traverses both trees and finds all the access routers in each tree. It determines all the access routers on one side and all the access routers on the other side of the congested link. Then, it goes through all the installed policies and finds all

the policies that have an ingress access router in the ingress section and an egress access router in the egress section. The predictor ignores the best effort traffic in all measurements since it does not need to predict the affect of the new policy on this kind of traffic. Here, we assume that the network administrator is only concerned about the traffic belonging to the customers who pay for the level of service they have requested. It is fair to assume that the bottlenecks are the link speeds in the network and not the processing power and the internal resources of each router as is true in today's networks.

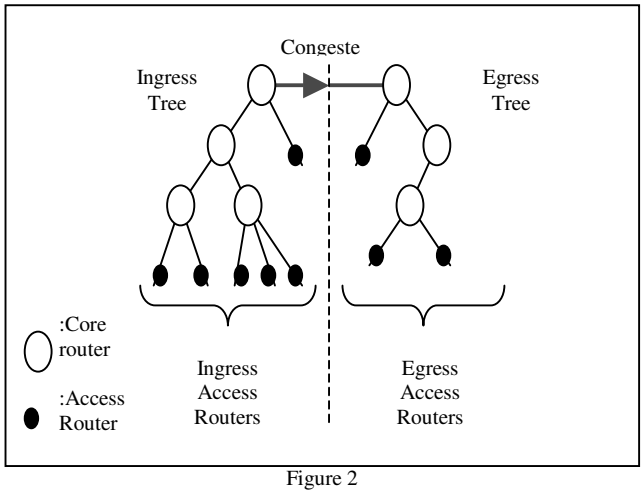

Figure 2

To show how the predictor works, we consider the following example.

Example

Let's consider the network as shown in Figure 3. In this example, there are six subnets, and each subnet is connected to the backbone through an access router. We assume that the routers are DiffServ enabled and there are four policies configured. Policies 1 through 3 specify better than best effort class of service and policy 4 is the default policy.

	Src IP	Dest IP	Src Port	Dst Port	Protocol	Service Level:
Policy 1:	*1.4.0.0*	*1.3.0.0*	*ANY*	*ANY*	*TCP*	*20kbps*
Policy 2:	*1.2.0.0*	*1.5.0.0*	*ANY*	*ANY*	*UDP*	*20kbps*
Policy 3:	*1.6.0.0*	*1.4.0.0*	*ANY*	*ANY*	*TCP*	*20kbps*
Policy 4:	*ANY*	*ANY*	*ANY*	*ANY*	*ANY*	*10kbps*
New Policy:	*1.2.0.0*	*1.3.0.0*	*ANY*	*ANY*	*TCP*	*30kbps*

IP Address	*Access Router Used*
1.1.0.0	Access Router 1
1.2.0.0	Access Router 2
1.3.0.0	Access Router 3
1.4.0.0	Access router 4
1.5.0.0	Access router 5
1.6.0.0	Access router 6

Table 1

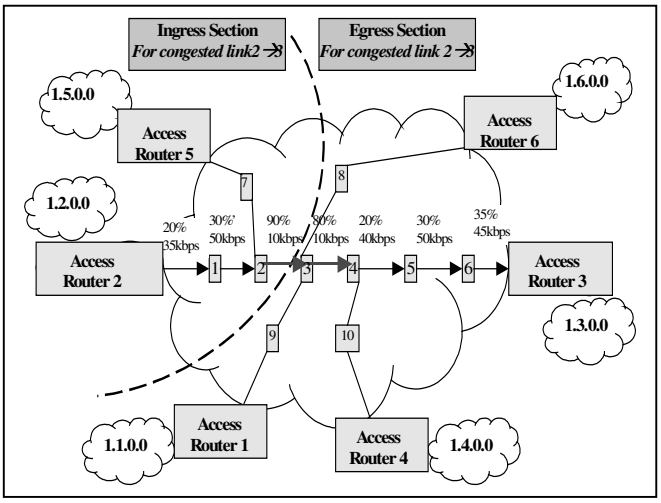

Figure 3

Error! Reference source not found. is used for finding the access router(s) used by each subnet for connecting to the backbone. With the current policies installed, all TCP traffic originating from the subnet 1.2.0.0 and terminating in the subnet 1.3.0.0, gets default class of service (10kbps). The new policy requires a better service (i.e. 30kbps) for this traffic flow. However, before adding this policy, the administrator needs to find out if this modification affects the traffic flows specified in policies 1 through 4. After finding the path between the two subnets (i.e. Access Router 2, router 1, router 2, router 3, router 4, router 5, router 6, Access Router 3), we calculate the free bandwidth on each link along this path. Let N be the difference between the old and the new required bandwidths and let b be the amount of free bandwidth on a link.

N = Old Rate – New Rate = 30kbps - 10kbps = 20kbps

Link 1: *Access Router 2 to Router 1*
b1 = 35kbps b1>N
Link 2: *Router 1 to Router 2*
b2 = 50kbps b2>N
Link 3: *Router 2 to Router 3*
b3 = 10kbps b3<N ∴ (Congested link)
Link 4: *Router 3 to Router 4*
b4 = 10kbps b4<N ∴ (Congested link)
Link 5: *Router 4 to Router 5*
b5 = 40kbps b5>N
Link 6: *Router 5 to Router 6*
b6 = 50kbps b6>N
Link 7: *Router 6 to Access Router 3*
b7 = 45kbps b7>N

In this example, there are two congested links identified (i.e. links 3 and 4). First, we consider link 3 (i.e. router 2 to router 3). We split our network into two sections, the

ingress section and the egress section. These two sections are on either side of link 3. As can be seen in Figure 3 the access routers 2 and 5 are the ingress routers and the access routers 1, 3, 4 and 6 are the egress routers that use link 3. Then we go through the installed policies and find the policies which specify traffic flows originating from any router in the ingress router section which are destined to any router in the egress section. As we see in Figure 3, there are no policies specifying such traffic flows. We follow the same procedures for the congested link 4 (i.e. router 3 to router 4). We find the ingress access routers to be 1, 2, 5 and 6 and the egress routers 3 and 4. From our policies, we find that policy 3 uses this congested link and therefore its traffic will be affected by the new policy.

III. SIMULATION AND RESULTS

To simulate the network prediction tool, we used the network simulator (NS2) with the DiffServ extensions added by Sean Murphy [7].

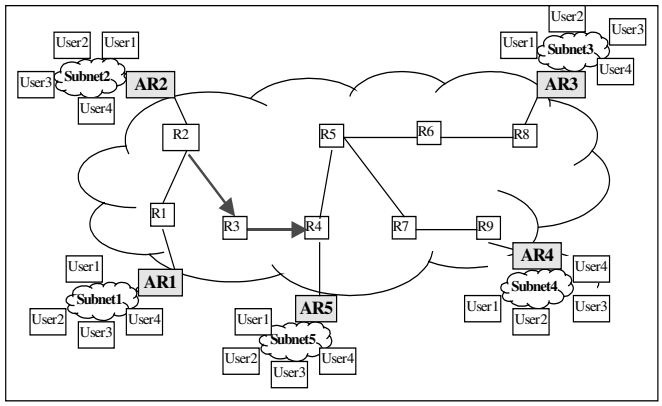

Figure 4

As seen in Figure 4, there are 5 access routers and 9 core routers in the simulated network. Each access router connects a LAN of four users to the backbone. The LAN speed is 10 Mbps and all the backbone links are 1.0 Mbps each. There are 11 policies specified. Policies 3, 6 and 9 give the highest class of service to all traffic matching the paths AR2 → AR5, AR4 → AR3 and AR5 → AR1, respectively. Policies 2, 5 and 8 give medium class of service to all traffic matching the paths AR1 → AR4, AR2 → AR3, and AR3 → AR2, respectively. Policies 1, 4, 7 and 10 give the lowest class of service to all traffic matching the paths AR1 → AR5, AR1 → AR3, AR4 → AR2 and AR5 → AR4, respectively. Finally, policy 11 is the default policy, which maps all other traffic flows not specified in policies 1 through 10 onto a best effort class of service. Note that the prediction tool does not determine the affect of the new policy on the best effort traffic.

	Source	**Dest**	**Class**	**Rate**
Policy 1:	*AR1*	*AR5*	*low/EF*	*100 kbps*
Policy 2:	*AR1*	*AR4*	*medium/EF*	*200 kbps*
Policy 3:	*AR2*	*AR5*	*high/EF*	*300 kbps*
Policy 4:	*AR1*	*AR3*	*low/EF*	*100 kbps*
Policy 5:	*AR2*	*AR3*	*medium/EF*	*200 kbps*
Policy 6:	*AR4*	*AR3*	*high/EF*	*300 kbps*
Policy 7:	*AR4*	*AR2*	*low/EF*	*100 kbps*
Policy 8:	*AR3*	*AR2*	*medium/EF*	*200 kbps*
Policy 9:	*AR5*	*AR1*	*high/EF*	*300 kbps*
Policy 10:	*AR5*	*AR4*	*low/EF*	*100 kbps*
Policy 11:	*ANY*	*ANY*	*BE*	
New Policy:	*AR1*	*AR3*	*high*	*300 kb/s*

A new policy is being added to move all traffic flows from AR1 to AR3 onto the highest class of service (300kbps). However, this traffic was previously being mapped onto the lowest class (100kbps) as specified in policy 4. The simulated prediction tool finds the old policy and calculates the difference between the old traffic rate and the new rate to be 200kbps (i.e. 300kbps – 100kbps). Table 2 shows the throughput on every link along the path from AR1 to AR3 before the new policy gets added. The links R2 → R3 and R3→ R4 (as highlighted in the table) show a high utilization where the free bandwidths are only 76.1kbps (i.e. 1000kbps – 923.9kbps) and 77.7kbps (i.e. 1000kbps – 922.3kbps), respectively. Since these free bandwidths are smaller than the needed extra bandwidth (i.e. 200kbps), the prediction tool marks these links as congested links. It then finds the affected policies to be policies 1 through 5 whose traffic passes through these congested links. The predictor then warns the administrator about a performance hit on these traffic flows so that he/she can decide whether or not to add the new policy.

Link	*Throughput Before New Policy*	*Unused Bandwidth*
AR1 → R1	424.2 kbps	575.8 kbps (> 200 kbps)
R1 → R2	425.2 kbps	574.8 kbps (> 200 kbps)
R2 → R3	923.9 kbps	76.1 kbps (< 200 kbps)
R3 → R4	922.3 kbps	77.7 kbps (< 200 kbps)
R4 → R5	625.4 kbps	374.6 kbps (> 200 kbps)
R5 → R6	609.4 kbps	390.6 kbps (> 200 kbps)
R6 → R8	608.6 kbps	391.4 kbps (> 200 kbps)
R8 → AR3	608.2 kbps	39.8 kbps (> 200 kbps)

Table 2: Table used for finding the congested links

To verify the prediction, the new policy is added and the traffic rates belonging to all the policies are again measured. Table 3 shows the rate of all the traffic flows in the network before and after the new policy is added. As predicted, adding of the new policy affects the traffic matching policies 1 through 5. All affected traffic flows use the links R2 → R3 and R3 → R4.

Path	*Corresponding Policy*	*Throughput Before New Policy*	*Throughput After New Policy*
AR1 → AR5	Policy 1	110.0 kbps	107.9 kbps
AR1 → AR4	Policy 2	204.1 kbps	199.9 kbps
AR2 → AR5	Policy 3	297.8 kbps	244.0 kbps
AR1 → AR3	Policy 4	103.7 kbps	284.8 kbps
AR2 → AR3	Policy 5	203.7 kbps	157.9 kbps
AR4 → AR3	Policy 6	300.7 kbps	300.7 kbps
AR4 → AR2	Policy 7	111.3 kbps	111.3 kbps
AR3 → AR2	Policy 8	205.8 kbps	205.8 kbps
AR5 → AR1	Policy 9	300.7 kbps	300.7 kbps
AR5 → AR4	Policy 10	111.7 kbps	111.7 kbps

Table 3: Table showing the affected policies

The following diagram is a plot of the accuracy of the predictor versus the link speeds in the core network. The solid line represents the accuracy of the predictor for token buckets of sizes 50kbits, 30kbits and 10kbits used for the high, medium and the low classes of service definitions, respectively, and the dotted line represents the accuracy for token buckets of sizes 5kbits, 3kbits and 1kbits. As can be seen from the plot, the accuracy of the predictor is high when the bottlenecks are the link speeds in the network and not the internals of the routers.

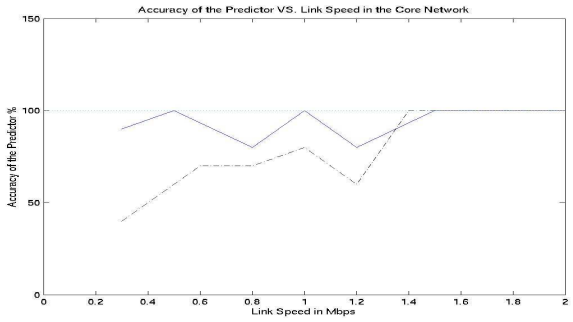

Figure 5

IV. COMPLEXITY AND SCALABILITY

In order to calculate the complexity of the network predictor, we define the following variables:

m: the number of core routers
n: the number of access routers
p: the number of installed/existing policies

To determine the congested links along the path of the new policy from the ingress access router to the egress access router the operations performed are on the order of m. Next, for every congested link, the predictor needs to traverse the ingress and the egress trees, which is on the order of m since there are a total of m nodes in both trees (combined) to be traversed. Next, in order to find the affected policies, the predictor performs nxp operations. The total number of operations performed by the predictor is linearly proportional to the total number of the core and access routers and the total number of the installed policies. Therefore, the network predictor is scalable and can easily be used for large networks with a large number of core and access routers.

$$o(\text{complexity}) = o(2m) + o(np)$$

CONCLUSION AND FUTURE WORK

The network predictor tool enables the administrator to determine whether changing a policy in a quality of service/policy enabled network would affect the performance of the traffic matching the existing policies. The network predictor tool is easy to implement and can greatly impact the ease of the configuration and management of a policy enabled network. It is also scalable and can be used for large networks having many core and access routers.

The network predictor described in this paper assumes that the quality of service measures are in terms of the traffic rate only. However, this measure can also be described in terms of the end-to-end delay and/or the packet loss percentage. As a future work item, we will investigate how the predictor would determine the congested links in the network being affected by the changes to the policy. We will also investigate how the predictor will determine the affected policies if they are associated with a life span. In this case, each policy is good only for certain periods of the time so the predictor needs to determine the overlaps of the policies during the same period of time.

REFERENCES

[1] D. Verma, M. Beigi and R. Jennings, "Policy based SLA Manaagement Management in Enterprise Networks", Workshop on Policies for Distributed Systems and Networks, January 2001.

[2] R. Rajan, D. Verma, S. Kamat, E. Felstaine and S. Herzog, "A Policy Framework for Integrated and Differentiated Services in the Internet", **http://www.allot.com/technology/PBN.htm**.

[3] S. Ibraheem, M. Kokar and L. Lewis, "Capturing a Qualitative Model of Network Performance and Predicting Behavior", Journal of Network and Systems Management, Vol. 6, No. 4, 1998.

[4] J. Goble and R. Mills, "COMNET III: Object-Oriented Network Performance Prediction", Proceedings of the 1994 Winter Simulation Conference, 1994.

[5] B. Frogner and A. Cannara, "Monitoring and Prediction of Network Performance", Proceedings of the International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems, 1998.

[6] C. Taylor and D. Meldrum, "Freeway Traffic Data Prediction using Neural Networks", National Technical Information Service, PB95-242343 , Springfield, 1995.

[7] S. Murphy, "DiffServ Additions to NS-2", http://www.teltec.dcu.ie/~murphys/ns-work/diffserv.