A Blocking Probability Reduction Method in Path Computation Schemes for Inter Domain Networks

Fatemeh Banaie, Mohammad Hossein Yaghmaee, Nazbanoo Farzaneh

Department of Engineering
Ferdowsi University of Mashhad
Mashhad, Iran
Email: Banaie.F@stu-mail.um.ac.ir, yaghmaee@ieee.org, Farzaneh@stu-mail.um.ac.ir.

***Abstract*— Path computation algorithms pose new challenges when extending them to larger inter domain networks. The process of path computation in these complex cases could be delivered to the external nodes like PCEs. Path Computation Element(PCE) architecture has been proposed for calculating end-to-end routes with computational constraints in multi domain network. In inter-domain cases, path computation schemes are more prone to blocking due to the long response time of the requests. To address this issue, we propose a novel algorithm for path computation to increase the number of successful requests while minimizing the blockage in network. This will significantly improve the network utilization. The main advantage of the proposed algorithm is to reduce the overhead of message flooding, so by this method the network load could be decreased notably, which can be seen in simulation results.**

***Keywords*-path computation; inter-domain; PCE; blocking; flooding (key words)**

I. INTRODUCTION

The rapid growth of networks, especially the internet, and the advent of new services such as voice over IP, audio/video streaming, video conferencing and VPNs, has changed the requirements of current transport networks. As a result considering of the quality of service has became a key requirement for next generation high speed networks. This can be obtained by the Traffic Engineering techniques. In this context, MPLS/GMPLS are the best protocols for the Traffic Engineering. One of the main challenges with traffic engineering is to compute a reliable end to end paths, while respecting a set of constraints in network.

On the other hand, the global telecommunication ecosystem is multi-domain in nature. So, it is imperative for multiple domains to interact with each other in an scalable manner. This imposes the need for inter-domain traffic engineering with considering end to end QoS [1]. Aiming to support path computation in inter domain environments, IETF has proposed path computation element (PCE) [2]. In these scenarios, there is a significant gap, in time between the path computation request and the reception of a reply to this request. As a result, at the time of the actual reservation/deployment , resources that were available during the path computation, may not be available anymore and block the deployment of the TE-LSP[2]. Blocking errors waste network resources, then they will negatively influence on network overall performance.

In this paper, our main contribution is to propose a method for increasing the rate of successful replies while minimizing blockage in the network. The proposed method consists of two mechanisms: the timer adaptation mechanism with the path length, and the flooding reduction mechanism that prunes the traffic to decrease network load. Results show a significant improvement in the performance of the network and the increased rate of successful replies.

The rest of the paper is organized as follows: Section II, presents related work on the topic. In section III, we describe the proposed method in detail. Section IV includes the simulation results and discusses the obtained results. Finally in section V, we conclude the paper with final remarks.

II. RELATED WORK

There are a lot of ongoing researches on inter-domain path computation algorithms. The aim of these researches is to calculate end-to-end paths in MPLS based multi-domain network. Some of these researches focus on extending the BGP protocol to carry QoS information and the other is attempting to design a new architecture or protocol beyond BGP as PCE [3]. Among these work, one of the most considered architecture is PCE architecture. PCE-based routing architecture can be divided into two major group: peer-to-peer and hierarchical [4]. In a peer-to-peer model, neighboring PCEs create peer-to-peer relations and PCEs are explored sequentially to determine the availability of the path [5]. Okumus et al. in [6] propose a hierarchical PCE approach. In this approach a single PCE is used in each domain and a central PCE in global. The central PCE aggregates information from domains and calculates the optimal end-to-end path.

The literature has proposed different methods for multi-domain path computation, but only recent methods are based on the PCE architecture [1]. Among these work, per-domain method [7] and the Backward Recursive PCE-based Computation [8] are important schemes. In per-domain method, each domain calculates the local portion of end-to-end LSP, until the entire end-to-end path is computed. In the second scheme, referred as BRPC, optimal path is determined through the pre-defined domain sequences. The PCE in the destination domain computes a tree of shortest path to the destination

978-1-4673-2073-3/12/$31.00 ©2012 IEEE

(VSPT) and sends it back to the previous PCE until it reaches to the PCE in the source domain. In [9], Path Computation Flooding (PCF) has been introduced to find the end-to-end path without any pre-configured domain sequences.

As previously mentioned, the request may be blocked due to the long response time in an inter-domain environment. This needs more signaling, e.g. Crankback [10], and sometimes needs LSP tear down in previous domains and redeployment of the LSP that can complicate the situation more [1]. To address this issue [11] proposes a method that uses an admission control mechanism to reduce the blockage . A similar issue has been addressed by [12] in an ATM network. But these methods are not designed for PCE path computation.

The work of [1] has proposed a path computation procedure based on the pre-reservation of the resources. In this method resources are pre-reserved at the time of the path computation. In other words, After sending a request, they set a timer and pre-reserve the resources at reply time. The time of pre-reservation is very important, so it should not be very long to waste the resources and also should not be expire before the actual deployment/reservation. Therefore, this technique guarantees the availability of the resources and reduces the blocking probability.

III. THE PROPOSED MECHANISM

In this paper, a method for blocking reduction is presented. The proposed model consists of two parts:

A. *Timer adaptation*
B. *Flooding reduction mechanism*

In the first part, our goal is to determine the appropriate time for pre-reservation timer. As discussed previously, the pre-reservation timer has an important role in the network performance. This is because if we consider a long time duration for pre-reservation timer, it wastes the network resources. Because they remain reserved and as a result the other requests may be rejected due to resource shortage. On the other hand, if timer expire before the reservation/deployment time, it may block the deployment of the TE-LSP, because the resources may be reserved by another request. So, adjusting the pre-reservation timer has a significant impact on network performance.

Based on what is discussed in section II, we have to flood the request to find the PCE's sequence. This imposes a large overhead in terms of network load and resource reservation. When we flood a request, it passes through different routes to reach the destination. So this makes a great number of massages broadcast in the network and the resources are reserved in all paths. While by pruning the non-promising paths we can reduce the traffic load and increase the rate of successful replies. This will improve the network performance by increasing the successful deployment ratio and decreasing the blocking probability.

a) The Reservation Mechanism

In [1] three basic messages are introduced:

- *[Path/Qos request]* that is composed of source-destination path and required Qos.
- *[Path/Qos reply]* that is sent in response to the request and pre-reservation is done in this step.
- *[Path/Qos request confirm]* is for the actual reservation.

First, source sends a request to the destination. After pruning the non-feasible paths, one or a set of feasible paths will be returned to the source. So, it can select the best one through the available paths and signal the LSP, Fig.1.

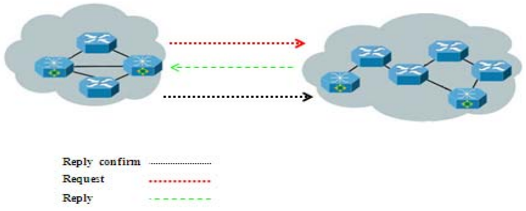

Fig. 1 The reservation mechanism

b) The blocking reduction mechanism

In inter-domain path computation scenarios, blocking probability increases due to the longer response time of the PCEs. We define the response time of the PCE as the time when a PCE sends a request until it receives the reply. This time can be affected by the network complexity and PCE limitations e.g. CPU limitation and workload of PCE [1]. Other factors also influence the blocking probability such as:

- Longer resource pre-reservation that is because of improper timer duration.
- A large number of paths may be reserved when we flood the request. This can cause the waste of resources in network.

To address these drawbacks we propose the following techniques.

A. Timer adaptation

As mentioned above, [1] proposes to pre-reserve the resources before the actual LSP deployment. In this method they set the pre-reservation timers for a fixed time based on experiment. But as we know the network conditions are constantly changing. Therefore, considering a constant value is not appropriate. Hence, our aim is to determine an appropriate time for the timer with considering the network conditions. This is done in two stages:

1) *Determining the initial value of the timer*
2) *Adjusting the timer value based on network conditions*

These two stages are described in detail in the following:

I. Determining the initial value of the timer

In this step, we first determine an initial value for the timer based on the network topology. We need this value for initializing the destination timer. This value can be obtained by experiment. So, we adjust the obtained value by the network conditions and path length.

II. *Adjusting the timer value based on the network conditions*

Parameters that are used in setting the timer are:

- Average queuing delay (D_q)
- Link delay (D_l)

End-to-end delay consist of queuing delay and propagation time, so they will impact the timer value. Since network conditions are changing, we must consider these parameters in setting the timer value. The amount of this value increases according to the increasing of the end-to-end delay.

Latency = Propagation Time + Queuing Delay

Thus the path latency can be written as follows:

$$latency = \sum_{j=0}^{P}(D_q^j + D_l^j) \qquad (1.1)$$

Where P is the number of nodes in the path from source to destination and D_q^j be the average queuing delay of PCE j and D_l^j be the link delay between PCE j and its neighbor through a path to the destination.

Queuing delay is a function of the number of packets in the queue. Packet arrival and packet departure changes queue length and as a result changes queuing delay. Thus a node monitors its queuing delay periodically. Due to pay attention to the past history of the node status, ${D_p}^k$ can be computed by using a Exponential Weighted Moving Average (EWMA). When a data packet arrives, each PCE monitors its current queuing delay (D_{cur}^k) and calculates an average value (D_q^k) using EWMA formula as follows:

$$D_q^k = (1-\omega) \times D_q^k + \omega \times D_{cur}^k \qquad 0 \leq \omega \leq 1 \qquad (1.2)$$

Where ω is the moving average coefficient with older values.

Since the resources must be reserved from receiving a reply until the time when confirm message is received; So PCEs that are closer to the source, require less time duration than the PCEs that are close to the destination. Therefore, we can adapt the time with the path length. In the proposed method, each node sets its timer according to its distance from destination and the queuing delay of the intermediate nodes. Therefore the value of the timer is not fixed in all nodes.

For adjusting the timer with the network conditions and path length, we subtract link delay and queue delay of the message in each PCE. Let T_d be the obtained time by simulation(which is calculated in pervious subsection) and D_q^k be the average queuing delay of PCE k and D_l^k be the link delay between PCE k and its neighbor through a path to the destination. So we can calculate PCEs pre-reservation timer by the following relation:

$$T_{timer} = T_d - \sum_{k=1}^{i}(\mathrm{D}_\mathrm{q}^\mathrm{k} + \mathrm{D}_\mathrm{l}^\mathrm{k}) \qquad (1.3)$$

Where i is the number of PCEs from destination to the current PCE, D_q^k is sum of the queuing delay from destination to the K*th* PCE and D_l^j is sum of the link delay from destination to the K*th* PCE.

With this technique resources will be kept less time duration in the PCEs that are closer to the source. As a result, resources will be more available and we have more reply successes. So we can service the requests more. This improves the network utilization and reduces the blocking probability of the requests. Fig.2 illustrates the pseudo-code of the proposed algorithm.

```
Algorithm  Timer Adaptation

• Given:
    R       is a set of [Path/Qos reply]
    Td      is a time that is computed based on topology
    $\mathrm{D}_\mathrm{q}^\mathrm{j}$   Queuing delay
    $\mathrm{D}_\mathrm{l}^\mathrm{j}$   Propagation time
• Algorithm:
Procedure Timer_adaptation()
    For each reply i in R
        Loop
            Compute path and set the destination timer on Td
            return the [path/Qos reply] to the requesting PCC
            For each PCE from destination to source
                Compute $latency = \sum_{j=0}^{P}(\mathrm{D}_\mathrm{q}^\mathrm{j} + \mathrm{D}_\mathrm{l}^\mathrm{j})$
                reserve resources and Set the timer on
                $T_{timer} = T_d - \sum_{k=1}^{i}(\mathrm{D}_\mathrm{q}^\mathrm{k} + \mathrm{D}_\mathrm{l}^\mathrm{k})$
                Send [Path/Qos reply] to previous PCE
            end for
        end loop
    end for
end procedure
```

Fig.2 Timer adaptation algorithm

B. *Flooding reduction mechanism(pruning mechanism)*

We propose a mechanism to decrease the number of messages that are flooded. Every request passing from a PCE, record its id and its cost to that PCE. This may happen in three states:

1. *Packet with the same Id have not received* : PCE records the packet Id and its cost , then send it to neighboring PCEs.
2. *We already received a packet with the same Id but its cost is greater than the received one*: we record the new values of the Id and cost, then send it to neighbors.
3. *We already received a packet with the same Id and its cost is lower than previous one*: In this case, the new request has more cost, so we can delete it because we have a path with lower cost.

According to what is discussed previously, we flood the request between PCEs until it reaches to the destination. Flooding in networks that connectivity is high, imposes a large overhead. The message complexity in the worst case is *O(N.D)*, where *N* is the number of nodes and *D* is the average degree of links at a node. Let d_i be the degree of node *i*, [13] prove that the number of messages that are generated by each flooding procedure is *N.(D-1)+1.*

$$d_1 + \sum_{i=2}^{N}(d_i - 1) = 1 + \sum_{i=1}^{N}(d_i - 1) \qquad (2.1)$$
$$= N.(D-1) + 1 \approx O\,(D.N)$$

This is the lower bound of flooding a request. But as seen in Fig.3 one node can receive the same request from all of its links and each time flood it to neighbors. Then we can rewrite the formula of equation (2.1) as follow:

$$D.\left[d_1 + \sum_{i=2}^{N}(d_i - 1)\right] = D.\left[1 + \sum_{i=1}^{N} d_i - N\right] \qquad (2.2)$$
$$= ND.(D-1) + 1 \approx O\,(D^2.N)$$

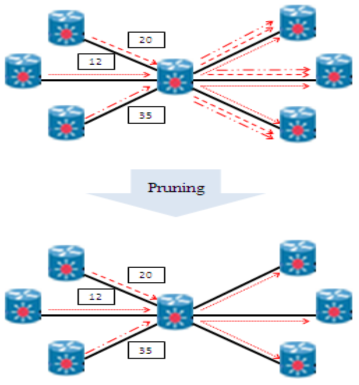

Fig. 3Example of flooding a request and pruning

```
Algorithm  Purring

• Given:
  Q     is a set of [Path/Qos request]
  T     is a table for recording the request Id and its cost
• Algorithm:
  Procedure Purring()
      For each request i in Q
          Loop
              Compare Ts indexes with the request ID
              If we already received a request  with the same ID
                  Compare its cost with the previous one
                  If its cost is greater than previous one
                      Delete the request
                  else
                      Record the new cost for that
                      request ID in T
                      Send  it to neighboring PCEs
                       and wait for [Path/Qos reply]
                  end if
              else
                  Record the request ID and its cost
                  Send it to neighboring PCEs and wait for
                   [Path/Qos reply]
              end if
          end loop
      end for
  end procedure
```

Fig. 4 The Pruning Algorithm

Therefore, pruning has two main advantages:

- Fewer routes are reserved, so we have more free resources and as a result the rate of successful replies and network utilization is higher in this case.
- It reduces the network traffic load almost by half

So, if PCE receives a request with the same Id, it compares the cost of the received request with the previous one. Request are sent only when its cost is less, otherwise it is deleted. Fig.4 illustrates the pseudo-code of the purring algorithm.

With this mechanism we can reduce the number of message flooding from order two to order one. This result in reducing the network traffic and Increasing the utilization. Fig.5 illustrates the pseudo-code of the proposed algorithm.

```
Algorithm

• Given:
  M       is a set of request, reply and reply confirm messages
• Algorithm:
  Procedure Reserve()
      For each message i in M
        Loop
            If the message is [Path/Qos request]
                If the destination is not in the PCE's scope,
                    Pruning();
                else
                    Timer adaptation();
                end if  the message is [Path/Qos reply]
                 Timer adaptaion();
                 Wait for [Path/Qos request confirm]
                 If we receive [Path/Qos request confirm]
                   Before the time of $T_{timer}$
                        Reserve resources
                   else
                        Release resources and send Err message
                        to the requesting PCC
                   end if
              end if
          end loop
  end procedure
```

Fig. 5 Reservation Algorithm

IV. SIMULATION RESULTS

In order to evaluate the performance of the proposed protocol, both of the proposed protocol and [1] are implemented in Opnet v.14 simulator [14]. For this purpose, a network topology as shown in Fig.6 is simulated. The following parameters have been used to evaluate the proposed protocol.

- Reply success ratio
- Network utilization
- Blocking rate of the requests
- Traffic load/ network load
- Path cost

The PCE request success that is the ratio of successful replies to the maximum number of requests. The network utilization that we define it as the ratio of the successful deployments to the maximum number of requests and network load that is the maximum number of messages that can be moved in the network. Path cost is the Maximum number of hop counts from source to destination.

In order to assess the performance of the proposed algorithms, the network has been examined in two scenarios: At first we consider the network parameters with implementing the timer adaptation scheme. After that, we also added the flooding reduction technique.

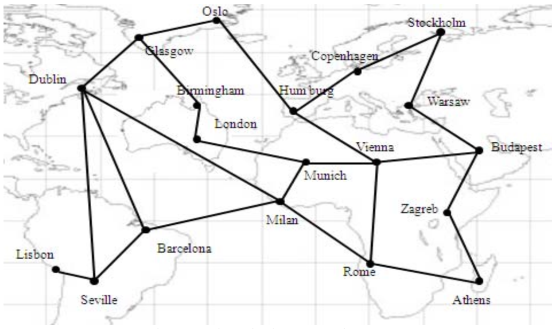

Fig. 5 Simulation topology

A. *Implementing the timer adaptation scheme*

At first scenarios only timer adaptation scheme is considered. In this mechanism, we first determine an initial value for the destination timer by experiment. Then, we reduce the reservation time of each intermediate PCE by regarding to the end-to-end delay of each PCE. Then, we reduce the reservation time of each intermediate PCE by regarding to the end-to-end delay of each PCE. The reduction in reservation time will be cause to increase the availability of resources in network, especially resources that are closer to the source. So we can service to the more requests in network. If we pre-reserve resources for a fixed time, resources may be kept for a long time duration. As a result, the other requests that need these resources may be rejected due to the shortage of resources. As seen in Fig.6 the rate of the request successes shows slight improvement when using the timer adaptation mechanism.

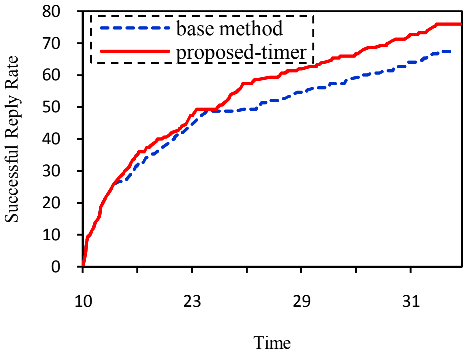

Fig. 6 Successful reply ratio

That's because the resources that are close to the source, are released earlier. So they are more available and this increase the reply success and decrease the blocking ratio of the requests, Fig.7.

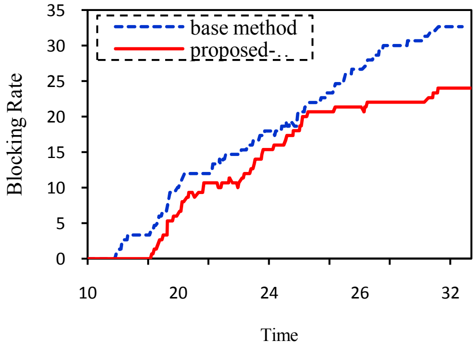

Fig. 7 Request blocking rate

The results in Fig.8 and Fig.9 show that we also have slight improvement in network utilization and reducing the network load. Because in this case, we have more successful deployment of LSPs, and according to definition of the network utilization, this can increase the network utilization. On the other hand, the reduction in blocking causes to decrease the number of messages that resends. Resending the messages not only lead to increase traffic in network, but also influences the network performance. With adjusting the timer, we can decrease the number of unsuccessful requests. So, network load can decrease due to the reduction in resending the messages.

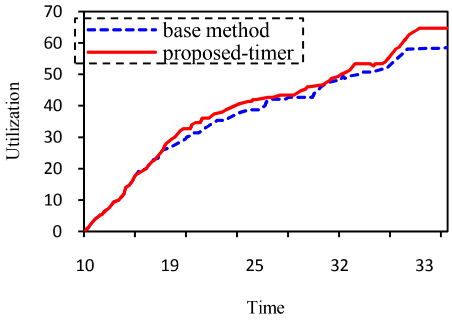

Fig. 8 network utilization

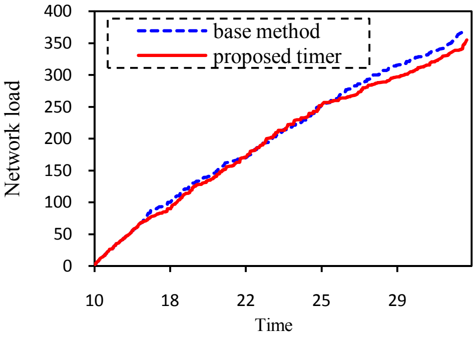

Fig 9 Network traffic load

B. *Implementing both the timer adaptation and flooding reduction scheme*

In order to evaluate the performance of the proposed protocol, both of the timer adaptation and flooding reduction scheme are implemented. Implementing the flooding reduction method will decrease the number of routes in network due to the pruning of the non-promising routes. As we have to pre-reserve resources in all of these routes, by pruning we can decrease the number of reserved resources in network. So, these resources can be used by other requests that need them and this increases the resource availability in network.

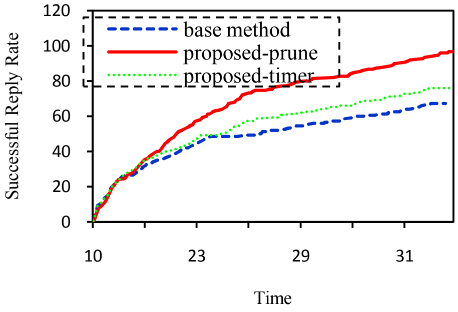

Fig. 10 Successful Reply ratio

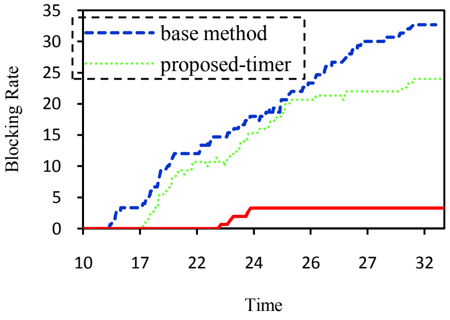

Fig. 11 Request blocking rate

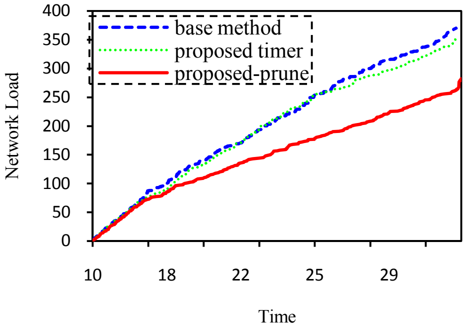

Fig.12 Network traffic load

As shown in Fig.10 simulation results show further improvement in the rate of request success. Because in this case we prune non promising paths. This lead to the reduction in the number of paths that are pre-reserved and as a result resources in the network are more available. This increases the rate of successful replies and decreases the request blocking rate.

We also measured the network load Fig.12 and network utilization Fig.13 that previously defined. Provided results show significant performance gain in the terms of network utilization and reducing network load.

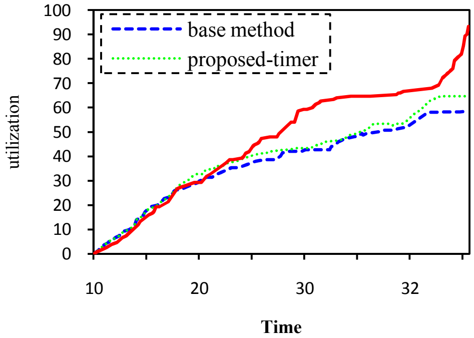

Fig. 13 network utilization

In the proposed method, we have more resource availability in network. If we consider hop count as the path cost, resource availability can cause to select shorter routes. Because, resource shortage in these routes may lead to select the longer routes. So, the average costs of the routes will be decrease by proposed method, Fig. 14.

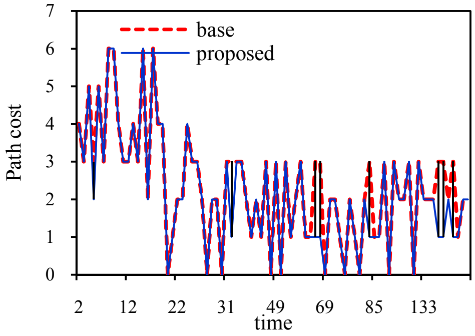

Figure 14.Path Cost

V. CONCLUSION

In this paper, we propose a method for increasing the successful deployment of TE-LSP in multi domain networks. It provides two mechanisms for increasing the network overall utilization. First, it increases the rate of successful replies by adjusting the pre-reservation timer with the path length and network conditions. Then, it reduces the overhead of the network by means of a flooding reduction mechanism that prunes non promising paths and only pre-reserve the promising ones. The simulation results give conclusive insight to the advantages of the proposed solution.

The results show that the proposed method increases the chance of the successful deployment of TE-LSP and decreases blocking probability without deteriorating the network utilization.

REFERENCES

[1] M. Shirazipour , S. Pierre, "End-to-end path computation schemes for traffic engineering in next generation multi-domain networks" , Ph.D

thesis, Montreal University of Canada, Juin 2010.

[2] A. Farrel, J.P. Vasseur, J. Ash, ''A path computation element (PCE)-based architecture'', RFC 4655, August 2006.

[3] R. Fu, "Next Generation Network Routing and Control Plane," Technical University of Denmark, Februuary 2011.

[4] O. Bonaventure, S.D. Cnodder, B.Quoitin, R.White, "Controlling the redistribution of BGP routes," IETF, 2003.

[5] L. Gommans, F. Dijkstra, C. de Laat, A. Taal, A. Wan, B. van Oudenaarde, T. Lavian, I. Monga, and F. Travostino, "Applications Drive Secure Lightpath Creation across Heterogeneous Domains", IEEE Commune. Mag., vol. 44, no. 3, pp. 100-106, Mar. 2006.

[6] T. Okumus, H. Junseok, H. A. Mantar, S. J. Chaplin, "*Inter-domain LSP setup using Bandwidth Management Points*," IEEE GLOBECOM 2001, pp. 7 – 11.

[7] J.P. Vasseur, A. Ayyangar, R. Zhang, "A per-domain path computation method for establishing inter-domain traffic engineering (TE) label switched paths (LSPs)" *,* RFC 5152, February 2008.

[8] J.P. Vasseur, R. Zhang, N. Bitar, J.L. Le Roux, ''A backward recursive PCE-based computation (BRPC) procedure to compute shortest constrained inter-domain traffic engineering label switched paths*''*, RFC 5441, April 2009.

[9] D. King, A. Farrel, "The application of the PCE architecture to the determination of a sequence of domains in MPLS & GMPLS*''*, IETF draft draft-king-pce-hierarchy-fwk-04.txt, July 2010.

[10] A. Farrel, A. Satyanarayana, A. Iwata, N. Fujita, G. Ash, " Crankback Signaling Extensions for MPLS and GMPLS RSVP-T"*,* IETF, RFC4920, 2007.

[11] H.A. Mantar, J. Hwang, I.T. Okumus, , S.J. Chapin, "A Scalable Model for Inter bandwidth Broker Resource Reservation and Provisioning*''*, IEEE Journal on selected areas in communications, vol. 22, no. 10, pp. 2019-2034, December 2004.

[12] S. Sanchez-Lopez, X. Masip-Bruin, J. Sole-Pareta, and J. Domingo-Pascual, "Fast setup of end-to-end paths for bandwidth constrained applications in an IP/MPLSATM integrated environment*''*, Computer Networks, vol. 51, no. 3, pp. 835-852, February 2007.

[13] S. Salsano, A. Botta, P. Iovanna, M. Intermite, A. Polidoro, "Traffic engineering with OSPF-TE and RSVP-TE: Flooding reduction techniques and evaluation of processing cost ", Computer Communications 29 , pp. 2034–2045, January 2006.

[14] OPNET Technologies, Inc., Opnet Modeler, http://www.opnet.com.