L2VPN Working Group
Internet-Draft
Intended Status: Experimental RFC
Expires: February 2013

Shankar Raman
Balaji Venkat Venkataswami
Gaurav Raina
I.I.T Madras
Bhargav Bhikkaji
Dell-Force10

August 17, 2012

Securing Model-C Inter-Provider VPLS L2 VPNs with Label Hopping and TicToc

draft-mjsraman-l2vpn-vpls-tictoc-label-hop-01

Abstract

In certain models of inter-provider Multi- Protocol Label Switching (MPLS) based Virtual Private Networks (VPNs) spoofing attack against VPN sites is a key concern. For example, MPLS-based VPN inter-provider model "C" for VPLS is not favoured, owing to security concerns in the dataplane, even though it can scale with respect to maintenance of routing state. Since the inner labels associated with VPN sites are not encrypted during transmission, a man-in-the-middle attacker can spoof packets to a specific VPLS site. In this paper, we propose a label-hopping technique which uses a set of randomized labels and a method for hopping amongst these labels using the time instant the packet leaves the port from a sending Provider Edge Router. To prevent the attacker from identifying the labels in polynomial time, we also use an additional label. The proposed technique can be applied to other variants of inter-provider MPLS based VPNs where Multi-Protocol exterior-BGP (MP-eBGP) multi-hop is used. As we address a key security concern, we can make a case for the deployment of MPLS based VPLS inter-provider model "C". Specifically we use the TicToc based Precision Time Protocol LSP to provide the timing for determining the time instant at which the packet is sent from the remote end Provider Edge Router and for calculating when it must have left the that peer at the Provider Edge Router at the near end / receiving end.

Status of this Memo

This Internet-Draft is submitted to IETF in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Copyright and License Notice

Table of Contents

1 Introduction

Multi-Protocol Label Switching (MPLS) [6] technology uses fixed size labels to forward data packets between routers. By stacking labels, specific customer services such as Layer 2 Virtual Private Networks (L2-VPNs) such as VPLS (Virtual Private Lan Service) based on Border Gateway Protocol (BGP) extensions are widely deployed in the Internet. BGP-based MPLS L2-VPN services are provided either on a single Internet Service Provider (ISP) core or across multiple ISP cores. The latter cases are known as inter-provider MPLS L2-VPNs which are broadly categorized and referred to as models: "A", "B" and "C".

Model "A" uses back-to-back VPN Routing and Forwarding (VRF) connections between Autonomous System Border Routers (ASBRs). Model "B" uses eBGP redistribution of labelled VPLS routes from Autonomous Systems (AS) to neighbouring AS. Model "C" uses multi-hop MP-eBGP redistribution of labelled VPLS MAC routes and eBGP redistribution of VPLS MAC routes from an AS to a neighbouring AS. Model "C" is more scalable for maintaining routing states and hence preferred for deployment in the Internet; refer to [2] for more details. Security issues in MPLS, especially MPLS-based VPNs has attracted attention [1]. The security of model "A" matches the single-AS standard proposed in [9]. Model "B" can be secured well on the control-plane, but on the data-plane the validity of the outer-most label (Label Distribution or Resource Reservation Protocol label) is not checked. This weakness could be exploited to inject crafted packets from inside an MPLS network core. A solution for this problem is proposed in [2]. Model "C" can be secured on the control-plane but has a security weakness on the data-plane. The Autonomous System Border Routers (ASBRs) do not have any VPN information and hence the inner-most label cannot be validated. In this case, the solution used for Model "B" cannot be applied. An attacker can exploit this weakness to send unidirectional packets into the VPN sites connected to the other AS. Therefore, ISPs using model "C" must either trust each other or not deploy it [4].

Control plane security issue in model "C" can be resolved by using IPSec. If IPSec is used in the data-plane then configuring and maintaining key associations could be extremely cumbersome. Even though model "C" is highly scalable for carrying VPN Routing and Forwarding (VRF) VPLS MAC routes, the vulnerability of the data-plane renders it unusable. The current recommendation is that model "C" must not be used. In model "C", there are at least two labels for each packet: the Provider Edge (PE) label, which defines the Label Switched Path (LSP) to the egress PE, and the VPN label, which defines the VPN associated with the packet on the PE.

In [5], the authors propose encryption techniques, such as IPSec, for securing the provider edge (PE) of the network. The authors also highlight that the processing capacity could be over-burdened. Further, if an attacker is located at the core of the network, or in the network between the providers that constitute an inter-provider MPLS VPN, then spoofing attacks are possible. The vulnerability of MPLS against spoofing attacks and performance impact of IPSec has been discussed in [3]. If the inner labels that identify packets going towards a L2 VPLS VPN site are spoofed, then sensitive information related to services available within the organizational servers can be compromised. As far as we know, there is no scheme available for installing an antispoofing mechanism for these VPLS VPN service labels.

This paper outlines a label-hopping technique that helps to alleviate the data-plane security problem in model "C". We propose a scheme that changes the inner VPLS VPN labels dynamically based on the time instant the packet is sent from the remote-end PE router. By using a mix of algorithms and randomized labels, we can guard against spoofing and related attacks. The advantage of our scheme is that it can be used wherever Multiprotocol-external BGP (MP-eBGP) multi-hop scenarios arise.

1.1 Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

2. Methodology of the proposal

2.1 PRE-REQUISITES FOR THE LABEL-HOPPING SCHEME

In this section, we briefly review the network topology for model "C", the PE configuration and the control-plane exchanges needed for our proposed scheme.

2.1.1 MPLS VPLS VPN model "C"

The reference MPLS-eBGP based VPLS VPN network for model "C" as described in [11] is shown in Figure 1, which also shows the control plane exchanges. The near-end PE (PEne) and far-end PE (PEfa) are connected through the inter-provider MPLS core. The VPN connectivity is established through a set of routers from different Autonomous Systems (AS) and their ASBRs. In the VPLS VPN, MP-eBGP updates are exchanged for a set of MAC based Forward Equivalence Classes (FECs). These FECs, which have to be protected, originate from the MAC

addresses / FECs behind PEne in a VPLS VPN site or a set of VPLS VPN sites.

2.1.2 PE configuration

Various configurations are needed on the PEs to implement the label hopping scheme. A set of "m" algorithms that generate collision-free labels (universal hashing algorithms) are initially implemented in the PEs. Each algorithm is mapped to an index A = (a1; a2; . . . am) where m >= 1. The bit-selection pattern used by the PEs for generating the additional label is also configured. PEne must be configured for a FEC or a set of FECs represented by an aggregate label (per VRF label) which will use the label-hopping scheme. For each FEC or a set of FECs, a set of valid labels used for hopping, K = (k1; k2; k3; . . . kn) where n >= 1 and, ki != kj if i != j, is configured in PEne. For the set of labels K time slices TS = (TS1; TS2; TS3 . . . TSn) are also exchanged. These time slices can be periodically changed and a new set of TS ranging from TS1 to TSn can be exchanged after a time duration TS_Exchange_Interval which itself can be randomized from time to time.In the case of bi-directional security, the roles of the PEs can be reversed. In addition to these data sets a random seed is also exchanged. This Random Seed which we will henceforth as Rseed is used to generate the label for the next time slot.

2.1.3 Control and data-plane flow

Initially, set K, set TS and the bit-selection pattern used by the PEs are exchanged securely over the control-plane. Optionally an index from A, representing a hash-algorithm, could also be exchanged. We propose that only the index is exchanged between the PEs, as it enhances the security, for two reasons. First, the algorithm itself is masked from the attacker. Second, the algorithm can be changed frequently, and it would be difficult for the attacker to identify the final mapping that generates the label to be used for a packet. Figure 1 depicts this unidirectional exchange from PEne to PEfa.

The control plane exchanges also involve a-priori constructing a Precision Time Protocol (PTP) LSP for deriving the clock at the PEne and PEfa for a forwarding direction. For the reverse direction another PTP LSP can be constructed as well. In the example that we illustrate we discuss about only a single forwarding direction. The PTP LSP port assigned for a forwarding direction is tied in with the configuration that goes into the inter-PEne-PEfa exchanges to setup the labelling control plane. So each pair of PEne and PEfa knows which PTP port and corresponding PTP LSP as per [12] to be used for the traffic. The PTP LSP is intended for providing the clocking between a pair of PEne and PEfa. The clock / timestamp derived from

this PTP LSP is used in the data plane operation to determine which label is valid at that time instant as will be seen in the Algorithms provided below.

Once the secure control-plane exchanges are completed, we apply the label-hopping technique, and PEfa forwards the labelled traffic towards PEne through the intermediate routers using the label-stacking technique (Figure 2). The stacked labels along with the payload are transferred between the AS and ASBRs before they reach PEne. Using the label-hopping algorithm PEne verifies the integrity of labels. Upon validation, PEne uses the label information to forward the packets to the appropriate VPLS VPN service instance or site. This data-plane exchange from PEfa and PEne is depicted in Figure 3. We now present the label-hopping scheme.

2.2 LABEL-HOPPING TECHNIQUE

In this section, we describe the label-hopping technique and discuss some implementation aspects. Once a data packet destined to the PEne arrives at the PEfa (a) a first-label is chosen using set K and set TS, and the random seed Rseed, and a first-label selected. Next (b) a selected number of bytes from the payload is chosen as input to the hashing algorithm. The hash-digest obtained as a result is used to obtain the additional label for the packet. The agreed bit-selection pattern is then applied on the hash-digest to obtain an additional label, which is then concatenated with the first label. Once PEne receives these packets it verifies both the labels.

The implementation steps for the control-plane at the PEne and PEfa are given by Algorithms 1 and 2. The implementation steps for the data-plane at the PEfa and PEne are given by Algorithms 3 and 4.

2.2.1 Algorithm 1 Control-plane PEne algorithm

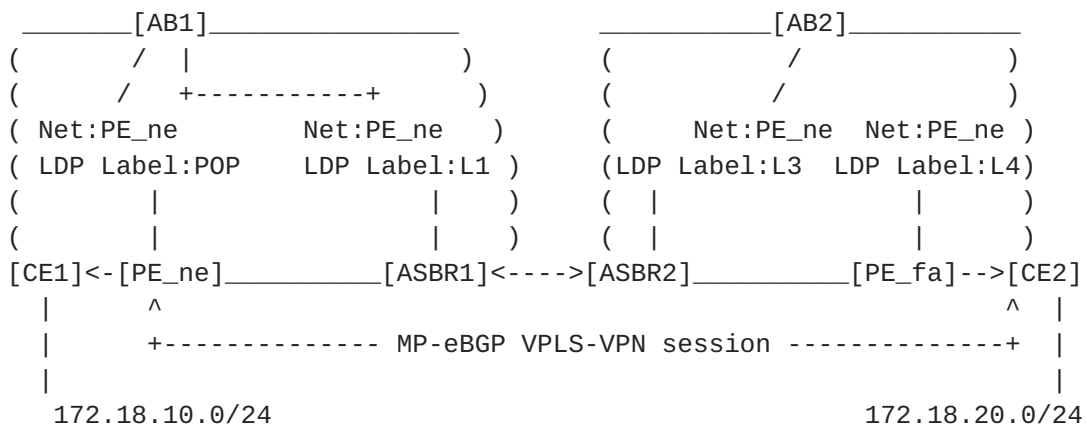
Require:

* FEC[] Forward Equivalence Classes,
* K[] valid labels,
* TS[] valid time slices,
* A[i] hash algorithm instance,
* I[] the bit-selection pattern chosen for the inner label.
* Random seed "Rseed" which is used for generating the index into set K (set of labels).
* PTP port and PTP LSP information

```
Begin
packet = makepacket(FEC,K, TS, A[i], I, Rseed);
CP-SendPacket(PEfa, MP-eBGP, packet);
End
```

Note: The values in K need not be contiguous and can be randomly chosen from a pool of labels to remove coherence in the label space. Also the algorithms used could be either vendor dependent or a set of standard algorithms mapped the same way by the PEne and PEfa. If the two PEs involved are from different vendors we assume that a set of standard algorithms are used.

Note: Also the values in set TS should be of a coarse granularity of seconds recommended to be higher than 2 seconds.



<(FEC[] Forward Equivalence Classes), (K[] valid labels),
(TS[] valid time slices), (A[i] hash algorithm instance),
(I[] the bit-selection pattern chosen for the inner label),
(Rseed - A random seed to generate the next label to be used in set K),
(PTP port and PTP LSP information)>.

Exchange all details as per Algorithm 1.

Figure 1: Control-plane exchanges for model C [11]

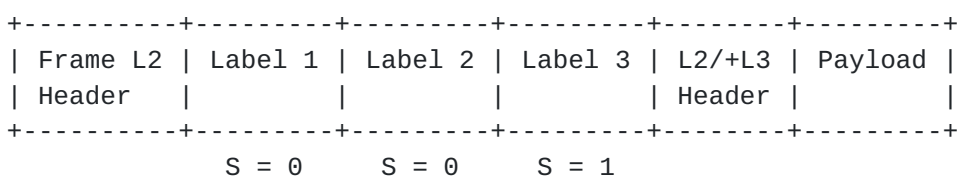


Figure 2: Label stack using scheme outlined for Model "C"

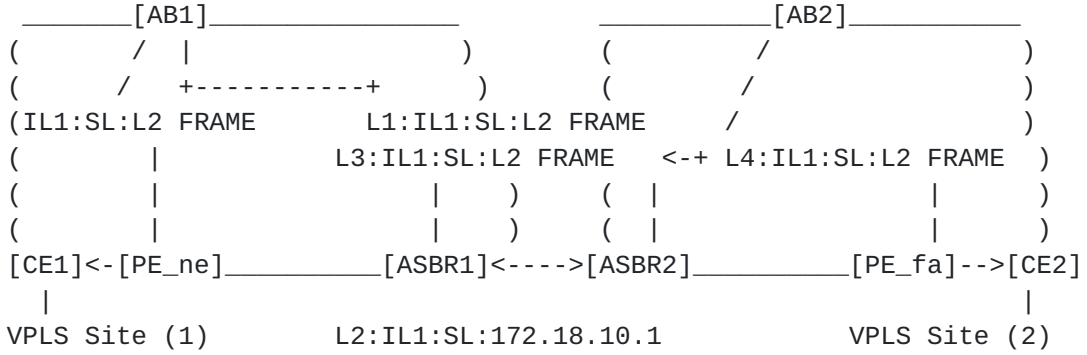


Figure 3: Data-plane flow for model C [11]

2.2.2 Algorithm 2 Control-plane PEfa algorithm

```
Require: None
Begin
packet = CP-ReceivePacket(PEne); // from PEne
FEC[] = ExtractFEC(packet); // extract FECs
K[] = ExtractLabels(packet); // extract the labels
TS[] = ExtractTimeSlices(packet); // extract the time slices
Rseed = ExtractRandomSeed(packet); // extract the Rseed value.
selectHashAlgorithm(A[i]); // hash algorithm to use
RecordValues(FEC); // information for PEfa
RecordValues(K);
RecordValues(TS);
RecordValues(I); // bit-selection pattern to be used
RecordValue(Rseed);
End
```

2.2.3 Algorithm 3 Data-plane PEfa algorithm

```
Require: None

Begin
Initialization :

One Time Init :

BeginInit

CurrentTimeSliceIndex = 0;

CurrentMasterClock = PTP LSP Master Clock Timestamp;

CurrentTimeInstant = CurrentMasterClock;

NextTimeInstant = CurrentMasterClock + TS[CurrentTimeSliceIndex];

EndInit

packet = DP-ReceivePacket(Interface);
match = CheckFEC(packet); // Is the algorithm enabled?
if match == 0 then
     return; // no match
end if
hash-digest = calculateHash(A[i],packet);
if (CurrentTimeInstant <= NextTimeInstant ((+ or -) configured
seconds)) then
     // do nothing;
else
     CurrentTimeSliceIndex++;
     if CurrentTimeSliceIndex == n then // check to wrap around
             CurrentTimeSliceIndex = 0;
     end if
     CurrentTimeInstant = NextTimeInstant;
     NextTimeInstant = CurrentTimeInstant + TS[CurrentTimeSliceIndex];
end if
first-label = K[GenerateRandom(Rseed) MOD n(K)];
end if
additional-label = process(hash-digest,I)
DP-SendPacket(PEne, first-label, additional-label, packet);
End
```

2.2.4 Algorithm 4 Data-plane PEne algorithm

```
Require: None
Initialization :
One Time Init :

BeginInit
CurrentTimeSliceIndex = 0;
CurrentMasterClock = PTP LSP Clock Timestamp;
CurrentTimeInstant = CurrentMasterClock;
NextTimeInstant = CurrentMasterClock + TS[CurrentTimeSliceIndex];
EndInit

Begin
packet = DP-ReceivePacket(Interface);
match = CheckFEC(packet);
if match == 0 then
     return; //no match
end if

label-in-packet=extractPacket(packet, LABEL);
inner-label=extractPacket(packet, INNER-LABEL);
hash-digest=calculateHash(A[i],packet);
if (CurrentTimeInstant <= NextTimeInstant ((+ or -) configured
seconds)) then
     // do nothing;
else
        CurrentTimeSliceIndex++;
        // Save the old RseedIndex into set K
        OldRseedIndex = RseedIndex;
        RseedIndex = (GenerateRandom(Rseed) MOD n(K));
        NextRseedIndex =
          LookAheadRseedIndex(GenerateRandom(Rseed) MOD n(K));
        RollbackRseed(Rseed by 1);
        if CurrentTimeSliceIndex == n then // check to wrap around
             CurrentTimeSliceIndex = 0;
        end if
     CurrentTimeInstant = NextTimeInstant;
     NextTimeInstant = CurrentTimeInstant + TS[CurrentTimeSliceIndex];
end if
// Check if label used before in the previous | current or future
// time slot can be used
// Check with OldRseedIndex, RseedIndex and NextRseedIndex
first-label-range = K[RseedIndex (+or- 1)];
additional-label = process(hash-digest,I)
if label-in-packet ! in first-label-range then
     error(); return;
end if
```

```
if inner-label != additional-label then
     error(); return;
end if
DP-SendPacket(CE1, NULL, NULL, packet);
End
```

Here configured seconds could be a fraction as well.

In order to avoid too many processing cycles in the line cards of PEne and PEfa, the hash- digest is calculated over a predefined size of the payload. An additional inner label is further added to enhance protection against spoofing attacks. With an increased label size, an attacker spends more than polynomial time to guess the VPN instance label for the site behind PEne. There could be two hash-digests that generate the same label. In this case, the two hash-digests is differentiated using the additional label. Collisions can be avoided by re-hashing or any other suitable techniques that are proposed in the literature [8]. If collisions exceed a certain number, then Algorithms 1 and 2 can be executed with a set of new labels.

Note :

It is to be noted that the change in the algorithm to randomly pick up a label for the next time slot will help in avoiding man-in-the-middle attackers from synchronizing with the time slots and the labels which in the previous version of the algorithm was predictable if a large number of packets were observed. The Random seed agreed upon will generate in lock step with the time slots at both the PEfa and PEna, the correct label to be used and that will throw off the attacker from synchronizing with such label changes. Thus even replay attacks may be harder to attempt in such a case.

2.2.1 Illustration

We now briefly illustrate the label-hopping scheme. In Figure 1, using Algorithms 1 and 2, a set of labels are forwarded from PEne to PEfa. The roles of PEne and PEfa are interchanged for reverse traffic. Figure 2 shows a packet from the data-plane for model "C", with the proposed scheme. In the figure, "Label 1" refers to the outermost label, while "Label 2" refers to the label generated from the set K and set TS and "Label 3" refers to an additional label generated as in Algorithm 3. This additional label has bottom of stack bit (denoted by S in Figure 2) set. These labels are stacked immediately onto the packet and the path labels for routing the packets to appropriate intermediary PEs are added. Figure 3 also shows these path labels used by the data packet to reach PEne. When the packet passes through the core of an intermediary AS involved in model "C", or through the network connecting the intermediary AS, the

intruder or the attacker has the capability to inspect the labels and the payload. However, the proposed scheme prevents the attacker from guessing the right combination of the labels. We can increase the size of the additional inner-labels thereby reducing threats from polynomial time attacks.

2.3 SIMULATION AND IMPLEMENTATION

In this section, we present the preliminary simulation results on performance, comparing the label-hopping technique with deep packet inspection where we encrypt and decrypt the complete packet. We also briefly highlight some implementation issues.

2.3.1 Simulation

Implementing the label-hopping scheme for all set of FECs belonging to any or all VPN service instances may cause throughput degradation. This is because the hashdigest computation and derivation of the inner-label / additional inner label calculation can be computation intensive. We therefore compared our technique by choosing a part of the payload as input to our hashing algorithm. We simulated our algorithm on a 2.5 GHz processor Intel dual processor quad core machine. We compared the performance of the label-hopping technique with a deep packet inspection technique where the complete packet was encrypted before transmission and decrypted on reception. These simulation figures indicate that we were able to process 10 million packets per second when we used 64-byte for hashing on a payload of size 1024 bytes. For a hash using 128-byte, we were able to process about 6.3 million packets per second. However with a deep packet inspection where we encrypted and decrypted the complete packet, we were able to process only about 1 million packets per second. In cases where performance becomes a bottleneck, this label-hopping scheme can be applied to specific traffic which are mission-critical, sensitive and most likely need to be protected as they travel from the PEfa to the PEne. Selective application of this service which could be offered as a premium for a selected set of FECs is a suitable option, there by protecting the traffic of organizations that are paranoid about the integrity of the switched traffic into their VPN sites.

2.3.2 Implementation

One of the concerns in the scheme is the use of payload for generating the random inner label / additonal label. If the payload does not vary between two packets then the control-plane exchanges have to be renegotiated with a different algorithm to be used for the hashing for the subsequent packets. The other concern in the scheme is to tackle the problem of fragmentation that can occur along the

path from PEfa to PEne. We can fragment the packet at PEfa and ensure that the size of the packet is fixed before transmission. We could also employ the Path Maximum Transfer Unit (Path-MTU) discovery process so that packets do not get split into multiple fragments. If packets are fragmented this scheme fails. However, networks usually employ the Path-MTU discovery process to prevent fragmentation and hence this problem may not occur.

2.4 CONCLUSION AND FUTURE WORK

In this paper, we proposed a label-hopping scheme for inter-provider BGP-based MPLS VPLS VPNs that employ MPe-BGP multi-hop control-plane exchanges. In such an environment, without label-hopping, the data-plane is subject to spoofing attacks.

The technique proposed uses a time-based label hopping scheme in addition to the use of the payload to generate an inner label to prevent attackers from easily deciphering labels and their respective VPNs. The scheme is less computationally intensive than encryption-based methods. It prevents the spoofed packets from getting into a VPN site even if the attacker is in the core or at an intervening link between ISPs. In our scheme, we chose the time instant that the packet leaves the first Provider Edge on the far end and this time instant serves as the variable component that the attacker cannot decipher. This requires the use of time synchronization mechanism. This is provided by the PTP LSP constructed for this purpose.

2.5 ACKNOWLEDGEMENTS

The authors would like to acknowledge the UK EP-SRC Digital Economy Programme and the Government of India Department of Science and Technology (DST) for funding given to the IU-ATC. The authors would also like to thank Chandrasekhar.R and Narayana Swamy for his review and valuable comments during the writing of this draft.

3 Security Considerations

The main objective of this proposal is to secure the Inter-Provider MPLS VPLS VPN Model-C data plane by preventing spoofing attacks and other unidirectional attacks against the customer site in this model. The suggestions and algorithms provided will mitigate these attacks to a large extent. The attacker will have many barriers to break through before he/she can successfully mount an attack against the customer site in this model with these algorithms implemented. The availability of TicToc as a method of clocking helps a great deal in this direction.

4 IANA Considerations

Appropriate IANA indicators would have to be provided to exchange the set of values that Algorithm 1 outlines in order to implement this scheme.

5 References

5.1 Normative References

[KEYWORDS] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC1776] Crocker, S., "The Address is the Message", RFC 1776, April 1 1995.

[TRUTHS] Callon, R., "The Twelve Networking Truths", RFC 1925, April 1 1996.

5.2 Informative References

[1] S. Alouneh, A. En-Nouaary and A. Agarwal, "MPLS security: an approach for unicast and multicast environments", Annals of Telecommunications, Springer, vol. 64, no. 5, June 2009, pp. 391-400, doi:10.1007/s12243-009-0089-y.

[2] M. H. Behringer and M. J. Morrow, "MPLS VPN security", Cisco Press, June 2005, ISBN-10: 1587051834.

[3] B. Daugherty and C. Metz, "Multiprotocol Label Switching and IP, Part 1, MPLS VPNS over IP Tunnels", IEEE Internet Computing, May-June 2005, pp. 68-72, doi:

10.1109/MIC.2005.61.

[4] L. Fang, N. Bita, J. L. Le Roux and J. Miles, "Interprovider IP-MPLS services: requirements, implementations, and challenges", IEEE Communications Magazine, vol. 43, no. 6, June 2005, pp. 119-128, doi: 10.1109/MCOM.2005.1452840.

[5] C. Lin and W. Guowei, "Security research of VPN technology based on MPLS", Proceedings of the Third International Symposium on Computer Science and Computational Technology (ISCSCT 10), August 2010, pp. 168-170, ISBN- 13:9789525726107.

[6] Y. Rekhter, B. Davie, E. Rosen, G. Swallow, D. Farinacci and D. Katz, "Tag switching architecture overview", Proceedings of the IEEE, vol. 85, no. 12, December 1997, pp. 1973-1983, doi:10.1109/5.650179.

[7] E. Rosen and Y. Rekhter, "BGP/MPLS IP Virtual Private Networks (VPNs)", RFC 4364, Standard Track, February, 2006.

[8] T. H. Cormen, C. E. Leiserson, R. L. Rivest and C. Stein, "Introduction to algorithms", 3rd edition, MIT Press, September 2009, ISBN-10:0262033844.

[9] C. Semeria, "RFC 2547bis: BGP/MPLS VPN fundamentals", Juniper Networks white paper, March 2001.

[10] Advance MPLS VPN Security Tutorials [Online], Available: "http://etutorials.org/Networking/MPLS+VPN+security/Part+II+Advanced+MPLS+VPN+Security+Issues/", [Accessed: 10th December 2011]

[11] Inter-provider MPLS VPN models [Online], Available: "http://mpls-configuration-on-cisco-iossoftware.org.ua/1587051990/ ch07lev1sec4.html", [Accessed 10th December 2011]

[12] Davari.S et.al, Transporting PTP messages (1588) over MPLS networks, "http://datatracker.ietf.org/doc/draft-ietf-tictoc-1588overmpls/?include_text=1", Work in Progress, October 2011.

[EVILBIT] Bellovin, S., "The Security Flag in the IPv4 Header",

RFC 3514, April 1 2003.

[RFC5513] Farrel, A., "IANA Considerations for Three Letter Acronyms", RFC 5513, April 1 2009.

[RFC5514] Vyncke, E., "IPv6 over Social Networks", RFC 5514, April 1 2009.

Authors' Addresses

Shankar Raman
Department of Computer Science and Engineering
I.I.T Madras,
Chennai - 600036
TamilNadu,
India.

EMail: mjsraman@cse.iitm.ac.in

Balaji Venkat Venkataswami
Department of Electrical Engineering,
I.I.T Madras,
Chennai - 600036,
TamilNadu,
India.

EMail: balajivenkat299@gmail.com

Prof.Gaurav Raina
Department of Electrical Engineering,
I.I.T Madras,
Chennai - 600036,
TamilNadu,
India.

EMail: gaurav@ee.iitm.ac.in

Bhargav Bhikkaji
Dell-Force10,
350 Holger Way,
San Jose, CA

U.S.A

Email: Bhargav_Bhikkaji@dell.com