Overlay-Friendly Native Network: A Contradiction in Terms? *

Srinivasan Seetharaman, Mostafa Ammar
Networking and Telecommunications Group, College of Computing
Georgia Institute of Technology, Atlanta, Georgia 30332
{srini,ammar}@cc.gatech.edu

ABSTRACT

It is a widely accepted notion that the native layer of the current Internet has begun to stagnate in terms of the services offered. Consequently, overlay networks have gained attention as a viable alternative to overcome functionality limitations of the Internet. A common approach to overlay network design holds the native network inviolable, implying that the overlay has to take on all the tasks needed to provide the desired high-level services. This limits the performance achieved and can potentially overburden the overlay layer. To solve these problems, we envision that, as overlay applications proliferate, the native layer should gradually evolve to suit the overlay network requirements. This paper proposes a framework for such an *overlay-friendly native network* (OFNN), which will cater to the overlay applications without compromising on the performance of the non-overlay applications. However, it is arguable that such a modification to the native network is contradictory to the fundamental reason overlay networks were conceived. We address this argument in our paper and classify the OFNN approaches as *contradictory* or *non-contradictory* based on whether the native layer modification is invasive or not. Further, we discuss the option of tuning the native layer parameters as a simple, yet feasible, non-contradictory OFNN approach. As examples, we present the overlay-friendly tuning of the native layer IGP hello-interval, BGP MED attribute and IGP cost.

1. INTRODUCTION

Overlay networks have recently gained attention as a viable alternative to overcome functionality limitations (e.g., lack of QoS, difficulty in geo-positioning, multicast support) of the Internet. The basic idea of overlay networks is to form a virtual network on top of the native network so that these specialized overlay nodes can be customized to incorporate complex functionality without modifying the underlying native routers. Third-party service providers can use these overlays to offer services, currently unavailable in the native network, to their customers. Examples of such services include multicast (e.g., Narada[1], Overcast[2]), optimized paths (e.g., RON[3], Detour[4], X-Bone[5], Brocade[6]), customized forwarding (e.g., I3[7], Scattercast[8]) and quality of service (e.g., OverQoS[9], SON[10]).

The design of today's IP networks calls for the internal network elements to concentrate on the simple forwarding function, leaving the high-level functions to the end-points. In accordance with that design approach, overlay networks have emerged as an effective way to implement functionality which would otherwise require significant change at the native IP layer. These overlay networks are constantly evolving to address the increasing demand for new services, while the native network has been allowed to stay unchanged[1]. Such an approach can overburden the overlay networks themselves as they must assume the full responsibility for any functions or services beyond best-effort unicast forwarding. Further, even if current overlay networks can perform the services expected from them, the amount of achievable performance gain is limited due to this inflexible design approach. To overcome these long-term problems, we suggest the native network evolve gradually to suit the overlay network and the new services sought. Hence, we investigate ways to improve the performance by *modifying* the operation of the current native layer.

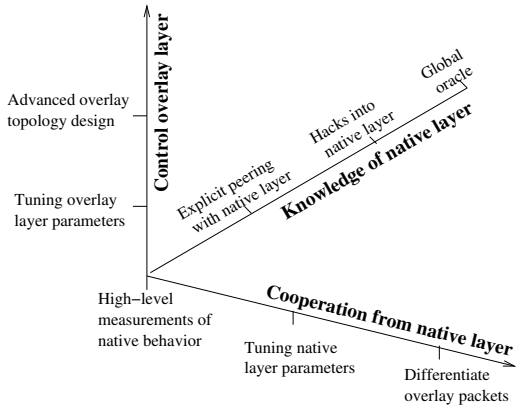

Figure 1: The various degrees of freedom available to overlay service developers.

In particular, the performance of the overlay network services is restricted because of unawareness of the native layer information and a lack of control over the native layer's decisions. Considering those reasons, there is a wide spectrum of optimizations one can propose for the overlay services, as indicated in Fig. 1. In the figure, design choices that are furthest away from the origin tend to provide the most benefit. Past research on improving overlay services has focused predominantly in two dimensions namely *controlling overlay layer* and *knowledge of native layer*. Examples include work on tuning the overlay layer parameters ([3, 11]), advanced overlay topology design ([12, 13, 14]), and obtaining native layer information ([6, 15, 16]). We explore a new degree of freedom that involves *cooperation from the native layer* for improving overlay services. The solutions in this direction are generally classified as providing an *overlay-friendly native network* (OFNN), briefly defined as a native network that caters to the overlay applications without compromising on the performance of the non-overlay applications. Such a native network should also be easily deployable, backward compatible and inexpensive to manage and operate.

The concept of an OFNN, however essential, comes across as a *contradiction in terms*. This is because overlay networks were conceived to obtain new network functionality without modification of the underlying native network. If

*This work was supported in part by NSF grant ANI-0240485.

[1]The overlay does not expect any help from the native layer because it cannot be modified and the native does not provide any help because no overlay requires it to - an instance of the classic chicken-and-egg problem.

it were feasible to modify the native network, the need for the overlay application is obviated. Therefore, modification of the native network to suit the overlay application seems to be a contradiction to the purpose. However, this is not the case always. There are modes of change in the OFNN which we argue does not constitute such a contradiction. For example, consider a certain feature that is currently implemented by means of an overlay and whose performance we would like to improve. Making a distinction that native layer operations are carried out by programmed *functions* which receive *parameters* (from managers or other functions) to guide their operation, we are faced with several choices to better support the feature at the overlay layer. Among these are:

A. Add a new function to the native layer.
B. Modify the existing functions in the native layer.
C. Tune the native layer parameters, without altering the functions at the native layer.

Clearly, the solutions put forward by options A and B are invasive procedures, requiring alteration of the native layer code. Hence, we do not see them as a feasible solution. In this paper, we advocate the use of option C, where we do not need to rebuild the native network and the functions used at the native layer remain fundamentally the same. We believe that this native layer tuning is a pragmatic approach to constructing an OFNN.

From the discussion above, we can see that some OFNN approaches represent a contradiction and some do not. Based on this observation, we classify the OFNN approaches into two categories - *contradictory* and *non-contradictory*. Furthermore, we present native layer tuning as a non-contradictory OFNN approach and illustrate the overlay-friendly tuning of the native layer IGP hello-interval, BGP multi-exit discriminator (MED) attribute and IGP cost.

Recent work on network virtualization highlighted the importance of addressing the impasse in progress of the current Internet[17, 18, 19, 20]. It suggests two perspectives in targeting the problem - that of a purist who considers that the overlay network is a tool to experiment a feature before full-fledged deployment in the native network, and a pluralist who considers that the diversity brought about by overlay networks should become a fundamental part of the native network. In this paper, we advocate an intermediate viewpoint where we consider the need for overlay networks to be inevitable, in addition to requiring some minor alteration of the native network. Our work on the OFNN is a step in that direction.

The remainder of the paper is organized as follows. We elaborate on the definition and design goals of the OFNN in Section 2. Some examples of the contradictory OFNN approach are briefly described in Section 3. We present our novel approach towards native layer tuning and identify some of the parameters that can be tuned in Section 4. Section 5 summarizes our position and suggests the required future work.

2. OVERLAY-FRIENDLY NATIVE NETWORK

We motivated the need for the overlay-friendly native network in Section 1. This section presents the design in better detail.

2.1 Definition and Design Goals

We define an *overlay-friendly native network* as a native network that incorporates special changes targeted towards the benefit of the overlay applications, without causing a negative effect on the performance of the non-overlay applications. We use the term *friendly* to highlight the special treatment rendered to the overlay application, irrespective of whether the native layer is aware of its existence. The change incorporated may not help all services, but is rather an optimization on a case-by-case basis. We refer to some changes as *invasive* to indicate whether the native layer functions need to be altered (reprogrammed).

The following are some of the design criteria mandated by the framework:

- The foremost concern is that adjusting the native layer must yield a significant performance gain for the overlay application. The performance metric depends on the type of service provided by the overlay network. For example, resiliency services are metered by recovery time, and multicast services by stress factor.
- It should have no negative effects on the way non-overlay applications and their traffic are handled. For instance, we should take special care that a change made in the native layer to support special packet scheduling of the overlay traffic must not lead to starvation of non-overlay traffic. In the presence of multiple coexisting overlay applications, the modification should have no negative effects on the traffic of the other overlay applications.
- The alteration made to the native layer must be conservative i.e., it should generate negligible, if not zero, extra overhead (in terms of processing, protocol, memory, and other resources) and must not cause a deterioration in existing performance of the overlay traffic (in terms of the classic metrics like throughput, forwarding rate, latency, recovery time).
- It should facilitate a practical roadmap for wide-spread deployment. In cases where the change to the native network is invasive, the benefits should gradually accrue as more native routers are altered to support the feature at the overlay layer. Such a change is considered incremental. To satisfy this requirement, the overlay network must provide minimal expected services even when the change is unavailable or partly available in the native network.
- The native network must be backward compatible in that it should still process legacy overlay applications (without the new functionality) in the expected manner. For example, consider a change in the native layer which aids in overlay path diversity (minimal overlap of the native route used by the overlay links). An overlay application that is incapable of using this added support from the native layer must still receive the basic connectivity between its overlay nodes.

From the design criteria stated above, we note that one key design goal is to avoid negative impact on non-overlay traffic. In some cases, the change made to the native layer might even be beneficial to certain non-overlay applications. To provide these potential benefits to non-overlay traffic, the OFNN should not distinguish between the different applications sharing the native network, wherever possible.

We also infer from the design criteria above that the change incorporated in the native layer does not necessarily depend on the proportion of native traffic that belongs to the overlay layer. However, certain invasive changes to the native layer become more justifiable in the presence of a high volume of overlay traffic.

2.2 Implementation Options

Having established the basic design goals of an OFNN, we proceed to list some of the possible design choices available, on top of the sample set listed in Section 1, for modifying the native layer[2]:

A. Add a new function to the native layer.

B. Modify the existing functions in the native layer.

C. Tune the native layer parameters, without altering the functions at the native layer.

D. Create packet level filters at strategic points of the native network to identify the overlay traffic or to provide functionality that can only be exploited by the overlay traffic. Once identified, the native layer can render some special treatment to these packets. This serves as a form of differentiated service.

E. Use a particular native layer function in a different manner, in combination with overlay-specific hacks. For example, the usage of native IP multicast for propagating queries in P2P networks, in association with reserving a permanent IP multicast address for each P2P network, is yet another approach for building an OFNN.

F. Make the native layer subsume the overlay feature i.e., add the whole feature that is currently being offered at the overlay layer to the native layer.

We can see that options A, D and E are inherently backward compatible by not affecting the set of functions currently in place. They treat all legacy traffic (both overlay and non-overlay) in the same manner as before. On the other hand, options B and C need special care that this change does not negatively impact the non-overlay applications. Option C is the only scheme that does not require a change to the native layer. It is interesting to note that option F obviates the need for the overlay application. Hence, it represents the highest level of modification at the native layer.

2.3 The Contradiction

We briefly explained the presence of a contradiction in Section 1. The contradiction arises in the OFNN when the overlay network, which was conceived to avoid modification of the native layer for obtaining new network functionality, demands a change in the native layer to improve its performance.

We would like to add that the contradiction arises because the overlay service is unable to provide the best performance autonomously. This inability can be attributed to the fact that overlay nodes are limited in number, mostly located at the edge of the network and are basically users of the native network services. For example, consider the following problems in previously proposed overlay services:

- Resilient overlays[3] - suffer from high chance of irresolvable overlay network partition, owing to lack of control over path diversity.
- Multicast overlays[1] - suffer from high stress and stretch, relative to native multicast.
- QoS overlays[9] - suffer from lack of absolute service guarantee, owing to presence of other non-conforming non-overlay traffic.

These problems motivate the need for support from the native layer, thereby leading to the contradiction.

We argue that some of the implementation options in Section 2.2 do actually represent a contradiction. Hence, based on the type of change, we classify the approaches adopted into the following two categories:

- *Contradictory OFNN* - when the required change is invasive, leading to alteration of the native layer functions. Options A,B,D,E and F in Section 2.2 represent this approach. We present existing proposals for these OFNNs in Section 3.
- *Non-Contradictory OFNN* - when the required change is non-invasive. Option C in Section 2.2 represents this approach. We present some instances of this OFNN in Section 4.

3. EXAMPLES OF CONTRADICTORY OFNN

The following are some OFNN proposals that require modification of the native layer to bring about a performance improvement for the overlay service.

3.1 Resource Sharing

One theme in past work on overlay network design is the addition of an intermediate layer to interface between the native and overlay layers. This new layer helps improve the performance of the overlays without exercising any control over the native layer. As suggested by Fig. 1, these schemes aid overlay services by obtaining a higher level of information (about topology, routes, and resources) from the native layer. However, there are obstacles to achieving that. The Internet has attained its current level of scalability mainly by information hiding. Hence, there is a fundamental limit on the amount of information one can obtain, which current probing schemes cannot subvert. Therefore, obtaining privileged information requires an invasive change at the native layer.

The work on network virtualization[17], diversified Internet[18, 19], Opus[21], routing underlay[15], service-oriented Internet[16], and Brocade[6] propose using this intermediate resource provisioning layer to allocate resources to the different overlay networks on top. In certain cases, this intermediate layer serves as a repository for routing services, high-level performance measurements and BGP information. It can also be used to provide optimized routes between peers by exploiting knowledge of underlying network characteristics.

All solutions listed above, albeit better than overlay networks that operate completely independent of the native network, are still plagued with the problem of limited gain (See Section 2.3 for details). Hence, we argue that greater performance gains can be obtained only when the functioning of the native layer is altered to bring about a synergy with the overlay layer.

[2]This list is by no means comprehensive.

3.2 Network Support for Overlay Networks

Jannotti proposed two new primitives for implementation in the native layer - *packet reflection* and *path painting*[22] - to provide advanced routing services like packet duplication and route examination. These two primitives support the efficient construction and operation of certain overlay networks. The work also showed ways to incrementally deploy the primitives in the Internet. However, this modification of the native network is invasive and inhibits widespread deployment.

3.3 Active Networks

Active Networking is an architecture proposed to address the problems of network stagnation[23, 24]. It suggests enhancing the network elements with more processing abilities, so that the applications at the end user can obtain higher benefits by uploading new protocols and code to the intermediate routers. This results in a higher programmability of the native layer. One can see that the active network is very much an OFNN, where the native layer helps enhance the performance of the overlay applications by being programmable. However, the active networking approach is highly complex and needs significant modification of the native layer.

4. EXAMPLE OF NON-CONTRADICTORY OFNN: TUNING NATIVE LAYER PARAMETERS

The current native layer parameters are tuned for the existing users of the native network services. However, there are parameters which can be tuned in an overlay-friendly manner without any negative impact on other applications. We focus on the tuning of native layer parameters that affect functions like routing (IGP or BGP), scheduling (IP priority), multicast, and security (firewalls, address translators).

In general, network administrators may leave some parameters at the default value configured by the equipment vendor. In such cases, it is usually easier to justify the overlay-friendly tuning. It is perhaps inadvisable to tune parameters that have been purposefully set at a particular value, so as to avoid any conflict with pre-established policies.

From our literature survey of existing overlay services, we observe that it would help the overlay layer if the native layer provided the following support:

- Earlier failure detection by the native layer[11]
- Symmetric routing for native routes between end hosts[11, 15, 25]
- Coherent cost metrics between the two layers[11, 26, 27, 28]

The following subsections discuss some ideas about how tuning can help the three requirements stated above. In particular, we tune the routing protocol hello-interval, BGP MED attribute and the IGP cost of each link, respectively, at the native layer, to aid the overlay applications. We assume that both the native and the overlay layers employ a dynamic routing protocol to adapt to changing network conditions.

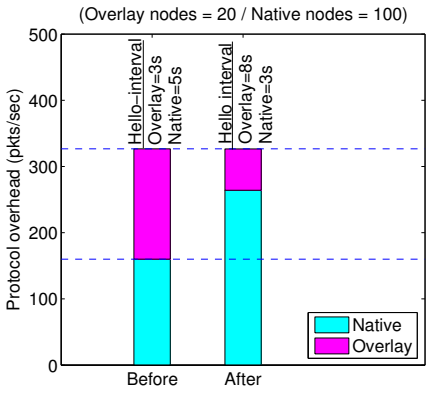

Figure 2: Example to illustrate the overlay-friendly tuning of the native layer hello-interval for a particular topology combination.

4.1 Tuning the Routing Protocol hello-interval for Faster Detection

In our previous work[11], we investigated the functionality overlap between the two layers - native and overlay, with respect to rerouting around link failures. We concluded that it is better to allow the native layer to detect the failure first for the following three reasons:

- The overlay paths experience a large number of route oscillations when the overlay layer attempts recovery first, owing to the independent operation of the routing protocol at the overlay and native layer.
- The overlay layer failure detection is vulnerable to congestion effects because of the lack of priority in the *hello* packets, leading to false positives. Moreover, the problem of false positives is amplified in the case of the overlay layer as each *hello* packet is sent over multiple native links, in contrast with native layer *hello* packets that are sent over a single native link.
- The native layer rerouting yields the optimal (shorter) alternate paths.

We achieved this earlier detection by decreasing the native layer routing protocol hello-interval from the default value[3]. Previous work on IGP convergence[29] illustrated how the hello-interval can be substantially reduced to achieve earlier detection, without incurring any stability problems. However, this has not been widely adopted for reasons of high protocol overhead[30]. In [11], we propose to increase the overlay layer hello-interval, which decreases the overhead at the overlay layer and compensates for the increase in protocol overhead at the native layer.

We would like to adhere to the following two constraints that keep the tuning operation conservative:

- Maintain the same overall protocol overhead (defined as the sum of the protocol overhead at both layers).
- Maintain the same effective detection time (defined as the minimum of the native and overlay layer detection times).

[3]The default value of the OSPF hello-interval for a Cisco 7600 series router is 10 secs.

In accordance with the above constraints, we tune the hello-interval used by the routing protocol in each layer. As illustrated by Fig. 2, the protocol overhead has the same value of 310 packets/sec and the effective detection time is 9 seconds (Assuming that the layer declares a failure after the loss of three *hello* packets). Once tuned, the native layer *hello* protocol detects the failure before the overlay layer and helps the overlay network achieve the best performance possible. This tuning does not have any negative impact on the non-overlay applications sharing the native network, but rather has the positive effect of earlier failure detection on all applications.

4.2 Other Examples

We present some of our preliminary ideas on the overlay-friendly tuning of two native layer parameters in this subsection.

A) Tuning the BGP Multi-exit Discriminator for Route Symmetry:

There are many overlay services that require the IP routing to be symmetric[11, 15, 25]. However, this is not true with the current Internet[31]. This can potentially cause the following problems:

- Uneven failure recovery process[11] and complicated troubleshooting, as either direction of the overlay link may not share the same network elements[31].
- Added complexity in having to maintain state information for either direction of an overlay link[15, 25].

Consider the scenario in Fig. 3 where the native layer picks route $KIGDBA$ in one direction and route $ACEFHJ$ in the other. To improve the chances of symmetric routes[4], we need to ensure that the domain uses the same border router for the exit as the entry. As the neighboring AS prefers exit routers with lowest MED value[5], one can tune the BGP MED attribute to pick the required exit router and thereby achieve route symmetry. Since the default scenario of hot-potato routing does not really use the MED value, we do not have any conflict with pre-established BGP policies.

As shown in Fig. 3, when advertising a route to a neighboring AS, the network administrator should set the MED value of each entry point to reflect the relative values of the IGP distance to the entry point. We do this setting only when the AS number of the current domain is bigger than that of the neighboring AS. This causes hot-potato routing in one direction and cold-potato in the other, thereby ensuring piecewise symmetry at the inter-domain level. We can see in Fig. 3 that AS100 prefers to use the exit router G to reach AS200 based on the MED values advertised. In the opposite direction, AS200 uses the closest router I as its exit router to reach AS100, as AS100 did not advertise a particular MED value. Ultimately, the MED tuning causes route $ACEGIK$ to be selected in both directions. This helps the overlay service in question, without harming the non-overlay applications that just look for basic connectivity. The MED tuning inherently requires cooperation between ISPs belonging to different administrative domains and hence brings in the question of incentive, which is out of the scope of this paper.

[4]This does not guarantee an end-to-end symmetric route or a symmetry in link properties.

[5]The default value for the MED attribute in a Cisco 7600 series router is 100.

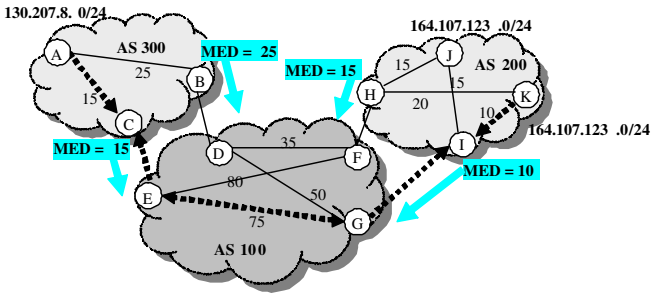

Figure 3: Example to illustrate setting of the MED attribute. The numbers indicated over each iBGP session represent the IGP distance between the two BGP routers. The dashed line indicates the path selected between the two prefixes.

B) Tuning IGP Cost for Coherence between Native and Overlay Layers:

A misalignment in the route computation strategy between the native and overlay layers can potentially lead to the following problems[11, 26, 27, 28]:

- Route oscillations that take longer to stabilize
- Defeat of traffic engineering
- Longer overlay paths (in terms of hop count, overall latency etc.)

These problems may also be caused when the two layers adopt mismatching cost schemes. The cost of the individual IGP links at the native layer can be based on hop-count, delay, Euclidean distance, special weights, load or capacity. Similarly, the overlay application might adopt a particular cost scheme based on the type of service offered. These two cost schemes may not necessarily match. For example, an overlay link L_1 with 10 native hops and a delay of 300 ms may be considered longer than link L_2 with 8 native hops and a delay of 400 ms.

In most cases, the overlay application might be able to determine the type of cost metric used by the underlying native layer by means of high-level performance measurements[3]. However, the overlay application might not be able to do that when the cost is based on special weights established by the ISP policy. Such cases will exacerbate the problems due to misalignment. Hence, it is desirable that the native layer IGP costs be configured using easily determinable metrics like propagation delay or hop-count.

5. DISCUSSION

In this paper, we stress the need for change at the native layer to suit the evolving overlay services and make the case for deploying an *overlay-friendly native network*. We also present the minimum criteria for designing a native layer change to support a particular overlay service. However, this modification of the native layer may be construed as a contradiction to the reason overlay services

were conceived. We argue that while invasive changes are indeed contradictory, other types of changes (such as tuning of the native layer parameters) are non-contradictory.

We consider the evolution of an overlay-friendly native network to be imperative, owing to the high resource usage by the overlay - caused by the potential for widespread use of overlays in the modern Internet, presence of multiple coexisting overlay networks on the same native topology, and usage of multiple native elements (links and nodes) by each overlay link.

The following questions that arise from our design section are worth answering so as to get a better understanding of the design:

- *Does the construction of OFNNs risk reducing the network transparency? And, does it involve an associated lack of predictability?*
- *How much cost will the ISP bear for these value-added overlay services? Is there sufficient economic incentive?*
- *Can such analysis be extended to other multi-layer models (as the network layer is nothing but an overlay of the link layer)?*

After describing the basic design of the OFNN, we presented examples on tuning existing native layer parameters to improve the performance of the overlay services, as an instance of non-contradictory OFNN design. This form of change at the native layer is non-invasive and is quite easily done by the network administrator. Such a change is not a contradiction in terms. Future work in this direction involves identifying other parameters that need to be tuned.

Though we adopt a stand against the contradictory OFNN in this paper, it is perhaps wise not to eliminate it altogether as an option for supporting overlay services. One possible future of the Internet is where overlay services are widely in use and, in such cases, it is easier to justify substantive changes to the native layer. If this does indeed become the case, future work should consider the development of overlay-friendly native network primitives (e.g., the ones proposed by Jannotti[22]) and study the deployability of such modifications. These changes must foster modularity and reusability, so that they can be applicable to a broad range of overlay applications.

We see the following as some of the required follow-up work:

- Develop more macros that satisfy our design criteria to incorporate in the native layer to help the existing overlay services.
- Designing other overlay services of interest to the consumers, under the assumption that the native layer is willing to cooperate.
- Creating a realistic testbed for these native layer changes. Most testbeds currently available only provide access to the overlay layer and do not yield to a multi-layer test environment.
- Develop ways to prevent a misuse by the overlay layer. When the native layer offers better control of its operation to the overlay layer, there is a chance that the overlay layer might violate the design semantics we specified.

6. REFERENCES

[1] Y. Chu, S. Rao, and H. Zhang, "A Case for End System Multicast," in *Proceedings of ACM SIGMETRICS*, June 1999.

[2] J. Jannotti, D. Gifford, K. Johnson, F. Kaashoek, and J. O'Toole, "Overcast: Reliable Multicasting with an Overlay Network," in *4th USENIX OSDI Symposium*, October 2000.

[3] D. Andersen, H. Balakrishnan, M. Frans Kaashoek, and R. Morris, "Resilient Overlay Networks," in *Proceedings of 18th ACM SOSP*, October 2001.

[4] S. Savage et al., "Detour: a case for informed internet routing and transport," Tech. Rep. TR-98-10-05, U. of Washington, Seattle, 1998.

[5] J. Touch, "Dynamic Internet Overlay Deployment and Management Using the X-Bone," *Computer Networks*, July 2001.

[6] B. Zhao, Y. Duan, L. Huang, A. Joseph, and J. Kubiatowicz, "Brocade: Landmark routing on overlay networks," in *1st Intl. Workshop on Peer-to-Peer Systems (IPTPS)*, March 2002.

[7] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet Indirection Infrastructure," in *Proceedings of ACM SIGCOMM*, August 2002.

[8] Y. Chawathe, "Scattercast: an adaptable broadcast distribution framework," *Multimedia Systems*, pp. 104–118, July 2003.

[9] L. Subramanian, I. Stoica, H. Balakrishnan, and R. Katz, "OverQoS: offering Internet QoS using overlays," in *Proceedings of ACM SIGCOMM*, 2003.

[10] Z. Duany, Z. Zhangy, and Y. Houz, "Service Overlay Networks: SLAs, QoS and Bandwidth Provisioning," in *Proceedings of ICNP*, November 2002.

[11] S. Seetharaman and M. Ammar, "On the Interaction between Dynamic Routing in the Native and Overlay Layers," Technical Report, http://www.cc.gatech.edu/%7Esrini/papers/layerAware.pdf, June 2005.

[12] J. Han, D. Watson, and F. Jahanian, "Topology Aware Overlay Networks," in *Proceedings of IEEE INFOCOM*, Mar 2005.

[13] S. Shi and J. Turner, "Placing Servers in Overlay Networks," *SPECTS*, July 2002.

[14] Z. Li and P. Mohapatra, "The Impact of Topology on Overlay Routing Service," in *Proceedings of IEEE INFOCOM*, March 2004.

[15] A. Nakao, L. Peterson, and A. Bavier, "A routing underlay for overlay networks," in *Proceedings of ACM SIGCOMM*, Aug 2003.

[16] J. Chandrashekar, Z. Zhang, Z. Duan, and Y. T. Hou, "Service oriented internet," in *Proceedings of the 1st ICSOC*, December 2003.

[17] S. Shenker, L. Peterson, and J. Turner, "Overcoming the Internet Impasse through Virtualization," in *Proceedings of HotNets-III*, November 2004.

[18] J. Turner and D. Taylor, "Diversifying the Internet," in *Proceedings of IEEE GLOBECOM*, 2005.

[19] F. Kuhns, M. Wilson, and J. Turner, "Diversifying the Network Edge," in *Work in progress*, 2005.

[20] "Overcoming Barriers to Disruptive Innovation in Networking," NSF workshop report, http://www.planet-lab.org/doc/barriers.pdf, Jan 2005.

[21] R. Braynard, D. Kostic, A. Rodriguez, J. Chase, and A. Vahdat, "Opus: an overlay peer utility service," in *Proceedings of the 5th OPENARCH*, June 2002.

[22] J. Jannotti, *Network layer support for overlay networks*, Ph.D. thesis, Massachusetts Institute of Technology, 2002, Supervisor-M. Frans Kaashoek.

[23] D. Tennenhouse, J. Smith, W. David Sincoskie, D. Wetherall, and G. Minden, "A Survey of Active Network Research," *IEEE Communications Magazine*, vol. 35, January 1997.

[24] David Reed, Jerome Saltzer, and David Clark, "Active networking and end-to-end arguments," *IEEE Network*, pp. 67–71, May 1998.

[25] Y. Chen, D. Bindel, H. Song, and R. Katz, "An Algebraic Approach to Practical and Scalable Overlay Network Monitoring," in *Proceedings of ACM SIGCOMM*, 2004.

[26] T. Roughgarden and E. Tardos, "How bad is selfish routing?," *Journal of the ACM*, vol. 49(2), pp. 236–259, March 2002.

[27] Y. Liu, H. Zhang, W. Gong, and D. Towsley, "On the Interaction Between Overlay Routing and Traffic Engineering," in *Proceedings of INFOCOM*, 2005.

[28] R. Keralapura, N. Taft, C. N. Chuah, and G. Iannaccone, "Can ISPs take the heat from Overlay Networks?," in *Proceedings of HotNets-III*, November 2004.

[29] C. Alaettinoglu, V. Jacobson, and H. Yu, "Toward millisecond IGP convergence," in *Proceedings of NANOG*, October 2000.

[30] "Complex Deployment and Analysis of Link-state Protocols," Cisco Press, Networkers, 2003.

[31] V. Paxson, "End-to-end Internet packet dynamics," *IEEE/ACM Transactions on Networking*, pp. 277–292, 1997.