On the Interaction between Dynamic Routing in the Native and Overlay Layers

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

***Abstract*— Overlay networks have recently gained attention as a viable alternative to overcome functionality limitations of the Internet. We are concerned with scenarios where a dynamic routing protocol is employed in the overlay network to adapt overlay routing tables to changing network conditions. At the same time, the native network over which the overlay is built also runs its own set of dynamic routing protocols. We are interested in investigating the behavior of this mixed routing environment and in particular the characteristics of the interaction between these two routing layers. In this paper, we focus on the specific problem of rerouting around failed links. We first study a *Dual Rerouting* scenario in which the two routing layers run completely independent of each other. Our goal is to understand the effect of the various settings of routing protocol parameters on the packet loss, number of route flaps, and the optimality of the adopted overlay path. We show that Dual Rerouting provides relatively fast path recovery. But, it tends to be sub-optimal in terms of the number of route flaps and the overlay path cost inflation. This is due to the overlap of functionality between the two layers, unawareness of the other layer's decisions, and lack of flexibility. We next investigate schemes that increase awareness of the native routing protocol and its parameters at the overlay layer. We consider three such approaches: *Probabilistically Suppressed Overlay Rerouting*, *Deferred Overlay Rerouting* and *Follow-on Suppressed Overlay Rerouting*. We show that with such schemes one can trade off longer path recovery times with improvements in route flapping and path cost inflation. However, there is a fundamental limit on the amount of achievable gain if we receive no support from the native layer. To counter that, we propose a novel approach towards the tuning of native layer parameters to suit the functioning of the overlay layer.**

I. 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 physical network so that overlay nodes can be customized to incorporate complex functionality without modifying native routers[1]. Typically, overlay networks are deployed by means of IP-tunneling and maintain an overlay routing table that is independent of underlying networks[1], [2], [3], [4].

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

[1]Our work pertains to persistent or infrastructure overlays, rather than peer-to-peer networks.

In this work, we are concerned with scenarios where a dynamic routing protocol is employed in the overlay network to adapt overlay routing tables to changing network conditions. At the same time, the native IP network over which the overlay is built also runs its own set of dynamic routing protocols. We are interested in investigating the behavior of this *mixed routing* environment and in particular the characteristics of the interaction between these two routing layers.

The complexity in the interaction between routing layers makes it quite challenging to study in general. In order to get a first set of insights into this issue we focus on the specific problem of *rerouting around failed native links*. An analysis of routing updates on the Sprint network has identified a significant portion (70%) of unplanned network outages to be due to single link failures[5]. It also highlighted that the failures are fairly common even in the modern Internet. This makes the specific problem we are addressing interesting in its own right, in addition to providing insights into mixed routing interactions in general.

We explore the following two scenarios:

- The overlay network is built on top of a single Internet domain and both the overlay network and the native network are running a link-state routing protocol (e.g., OSPF[6])
- Each node in the overlay network resides in a separate Internet Autonomous System. The overlay network runs a link-state routing protocol while a path-vector protocol (e.g., BGP[7]) is run among the ASes.

Before proceeding, we first observe that the mixed dynamic routing environment can be avoided by not using dynamic routing in the overlay layer and always counting on the native network to re-configure routes as network conditions change. This is undesirable. To illustrate this, consider, for example, the topology in Fig. 1, where the overlay path AI is statically routed through the overlay node G. When the native link FG fails, the overlay links AG and GI fail. Node G has been separated from the native network. Hence, native rerouting of individual native paths AG and GI will not work. Because the overlay network's routes are static, the overlay network cannot recover from this failure. If on the other hand, the overlay network is capable of performing dynamic routing, then the routing tables may change to allow for the path AEI to be used for overlay traffic. This shows that overlay dynamic routing can significantly enhance the overlay network's survivability.

In our investigation, we first consider a *Dual Rerouting*

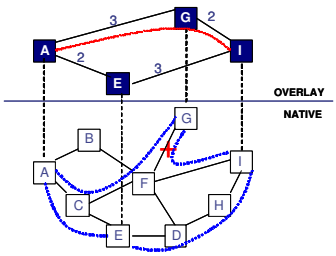


Fig. 1. An illustration of an overlay network. The dotted lines in each layer represent the path between the two nodes.

scenario in which the two routing layers run completely independently. Our goal is to understand the effect of the various settings of protocol parameters on the routing performance as measured by several metrics (discussed in Section III-C. Our investigation shows that Dual Rerouting suffers a performance degradation as a result of the overlap of functionality between the two layers, unawareness of the other layer's decisions, and lack of flexibility.

We next suggest two approaches to mitigate the problems of Dual Rerouting:

1) Adjust the overlay layer functioning without affecting the native layer
2) Adjust the native layer functioning

The first approach is motivated by the fact that the native network is tuned for the native applications and it seems pragmatic to not alter its operation. As part of the first approach, we investigate simple schemes that increase the awareness of the native routing protocol and its parameters at the overlay layer. We consider three such approaches: *Probabilistically Suppressed Overlay Rerouting*, *Deferred Overlay Rerouting* and *Follow-on Suppressed Overlay Rerouting*. We show that with such schemes one can trade off a deterioration in one metric with improvements in another. Though one can devise more complex approaches, it is questionable whether these schemes will be practical. Hence, we restrict our analysis to modest alterations of overlay routing.

The improvements achievable using the scheme mentioned above are limited because we have no control over the native layer. Hence, we propose the second approach to obtain more benefits. As part of this approach, we tune the native layer routing protocol to achieve a performance gain for the overlay traffic. In particular, we tune the keepAlive-time of the native layer to improve the overall rerouting efficiency. This can be construed as a new form of awareness at the native layer.

Our contributions are three-fold:

1) We provide an understanding of Dual Rerouting, its effects and compare rerouting at the two layers.
2) We develop means to provide better control at the overlay through layer awareness, with constraints of restricted access to native layer information.
3) We motivate the need for an overlay-aware native network to break free of all performance limitations and recommend tuning the native layer's parameters.

The remainder of the paper is organized as follows. Related work is briefly described in Section II. We explain the model adopted for analysis in Section III. We elaborate on the characteristics of Dual Rerouting in Section IV. We develop the layer aware rerouting algorithms in Section V. Simulation results and the corresponding inferences are presented in Section VI. We make the case for tuning the native layer routing protocol and present our novel approach in Section VII. This paper is concluded in Section VIII.

II. Related Work

Multilayer recovery is a popular problem faced in many high-speed networks[8], [9], [10], [11], [12], [13]. Previous work investigated the tradeoffs involved in using an IP-only recovery scheme, a WDM-only scheme, a hybrid scheme that partitions the network traffic, or a layer-coordinated scheme that can be tuned with external parameters. However, all these mechanisms fail when the two layers work in an independent manner, making our problem all the more interesting.

The problem of rerouting around failures in a single layer is a well-studied problem in different types of networks[14], [15], [16]. Typical solutions suggest resource reservation for a backup path that will be activated in the event of a failure or reactive approaches that perform dynamic restoration. Each strategy varies in the success rate of recovery, bandwidth used and the recovery speed. There has been little work in the field of overlay network recovery. Backup path allocation schemes exploiting the high correlation of overlay links have been studied in [17]. A substantial work in the field of overlay layer survivability is the work on resilient overlay networks (RON)[1], which is an effort to make the overlay network robust in the face of native layer problems. RON uses application-specific probes to determine the best path for particular metrics and performs dynamic reconfiguration.

Our work differs from previous research in four different ways. First, this paper analyzes the effect of different levels of awareness in the overlay layer on the routing performance. Second, in contrast with the IP over WDM fault recovery work, our lower layer takes substantially longer to detect failures and reroute a path, as link level notification is not always guaranteed. Third, in our work a limited set of nodes are available for operating the overlay network. Finally, our work captures the dynamics in both single-domain and multi-domain systems. Unless otherwise mentioned, the schemes and models discussed are applicable to both systems.

A complementary work on failure detection algorithms for overlay networks can be found in [18]. This work aims at reducing detection time, probability of false positive, control overhead and packet loss rate, by using information sharing across overlay nodes and storing state information of neighbors. Our work is independent of the detection algorithm used and hence orthogonal to their work. The work in [19] emphasizes the importance of a synergistic co-existence between the native and overlay layers, in the light of load constraints. It also characterizes the oscillations that occur due to the presence of multiple independent overlay networks. In our work, we do not consider load constraints as it is not yet a practical

feature in most overlay networks. We are trying to address the more fundamental questions of functionality overlap and layer awareness.

Recently, researchers analyzed the interactions between selfish overlay routing and traffic engineering in a game theoretic approach[21], [20]. Unlike our work, the game theory analysis applies only to cases where the native and overlay layers operate independent of each other. The previous work does not consider the interaction of protocol timers and fails to present ways to mitigate the effects of the interaction.

J. Han et al[22] present proactive ways for careful placement of overlay nodes to aid the robustness of the overlay network. We restrict our work to reactive approaches so as to cater to any topology configuration.

III. Rerouting Model

A. Framework

Overlay networks represent a virtual network of selected nodes communicating at a layer higher than the network layer. Each overlay node has a well-defined set of neighbors, not necessarily adjacent in the native network. The virtual link between any two overlay nodes constitutes *the native path* between the two nodes. The data communication between the overlay nodes can be accomplished by means of IP-in-IP tunneling[2], [23], where the non-overlay nodes route packets over native links based on the first IP header and the overlay nodes route packets over overlay links based on the second IP header[2]. To facilitate that, the overlay layer maintains a routing table that is independent of the native layer's routing table. We assume a dynamic routing protocol is used to maintain the overlay routing table.

In order to get a first set of insights into the issue of inter-layer interaction, we focus on the specific problem of rerouting around failed native links owing to the following two reasons:

- Failures exacerbate any negative effects of the interaction.
- Native links represent the smallest unit of abstraction one can analyze (i.e., failure of any intermediate native node can be translated to the failure of multiple native links)

Past research has primarily dealt with network oscillations caused by load constraints and traffic engineering[19], [20]. Owing to lack of data on actual traffic demands and lack of support for QoS in the current Internet, we do not consider the issue of load in our analysis. Thus, we are more concerned with the change in routes and the corresponding timing, which is a more practical approach applicable under any load condition.

We consider two scenarios:

1) Single-Domain Overlay over Single-Domain Native: In this scenario an overlay network is built on top of a single native network domain. Both networks use a link-state routing protocol (e.g., OSPF). Any of the links in the native network can fail in this scenario.
2) Single-Domain Overlay over Multiple-Domain Native: Here each node in the overlay network resides in a separate Internet AS. The overlay network operates as a single domain and uses a link-state dynamic routing protocol. A path-vector (BGP-like) protocol is used among the ASes to dynamically route around failures in the inter-domain links. To focus on the inter-domain aspects of dynamic routing, we consider only inter-domain link failures in this model. We assume that a link-state routing protocol is used as the intra-domain routing protocol. However, in this scenario, we are not concerned with intra-domain link failures.

The routing protocols adopted in each layer have the following generic parameters:

- An appropriate cost assigned to individual native or overlay links.
- An algorithm to determine the shortest path between two given nodes.
- *KeepAlive messages*: These are exchanged between nodes at both ends of the same native or virtual link for the purposes of link management. The *keepAlive* message exchanges are only of interest when they occur across links that are subject to failure. For example in the single-domain overlay over multiple-domain native case, only *keepAlive* message exchange across overlay links and inter-domain native links are of interest to us.
- A *keepAlive-time*[3]: This represents the frequency at which neighbors exchange *keepAlive* messages.
- A *hold-time*[4]: A native or virtual link is generally assumed to be down if no *keepAlive* messages are received during the hold-time. This parameter directly influences the delay in detecting a link failure.

B. Rerouting schemes

We consider three approaches for the operation of the dynamic routing protocols at the native and overlay layers:

a) No awareness: This corresponds to what we have dubbed *Dual Rerouting*, where the two layers operate independently to reroute around a failure. This serves as the benchmark for performance comparison.
b) Awareness of lower layer's existence: The overlay layer is aware that the native layer might attempt rerouting on its own, leading to a functionality overlap. This scheme tries to mitigate the problems arising from such an overlap without any further knowledge. We propose two solutions in this context - *Probabilistically Suppressed Overlay Rerouting* and *Deferred Overlay Rerouting*.
c) Awareness of lower layer's parameters: The overlay layer is informed of at least some of the native layer routing protocol parameters. We propose one solution in this context - *Follow-on Suppressed Overlay Rerouting*.

We are interested in exposing the performance limitations of the Dual Rerouting approach and then understanding the extent to which functionality overlap can be reduced by adding more awareness of the native layer at the overlay layer. We are also interested in the effect that such increased awareness has on the performance of the mixed routing scheme.

[2]Another approach would use a different non-IP addressing scheme for overlay nodes and an overlay header encapsulated within the native IP headers. The overlay routing table in this case would be indexed by overlay addresses.

[3]KeepAlive-time is also called hello-interval in certain routers.

[4]Hold-time is also called dead-interval in certain routers.

C. Performance metrics

We evaluate the performance of the mixed routing approaches from the perspective of the availability of the overlay path(s) affected by an individual native link failure. We define *recovery* as the rerouting immediately following a failure. Though a rerouting event at a particular layer does not necessarily cause the traffic to change course, a recovery event does. Our focus is on the performance of the recovery process required to re-establish an overlay path affected by the failure[5]. Such a path will be established when either:

- The native network establishes a new native path between the two ends of the broken overlay link, or
- The overlay network determines a new overlay route between the two end points of the broken overlay path.

The exact order in which these two events occur can affect the routing performance.

We are also interested in evaluating the performance of the routing protocol when the failed native link is repaired.

We use the following four metrics.

1) Hit-time: Hit-time is defined as the time period after a native link failure during which there is no communication between the two overlay nodes at the ends of a failed overlay link. Based on the traffic characteristics, the systems incurs a certain amount of packet loss during this period. Hit-time is made up of the following components:

- *Detection time*: This is the time from when a link fails and a node that is at one end of the failed link determines that the link has failed. The failure can happen anytime during the life of the hold-timer. Hence, the detection time is generally a random time between (0, hold-time).
- *Convergence time*: This is the time from when a link fails until all nodes in the network are aware of the failure and the routing changes are enforced[24], [25].
- Time to compute the route
- Device time to activate the new route

When both layers attempt to reroute around a failure, hit-time is the time taken by the layer that completes rerouting first. Thus, we define the *effective detection time* as the smaller of the detection times at either layer. If neither layer is able to reroute around the native link failure, the hit-time is infinite.

2) Success rate of rerouting: The success rate of rerouting is defined as the proportion of failed overlay paths that get rerouted successfully. A path may not be rerouted successfully because a native link failure caused a partition in the native network or the overlay network.

It can be inferred from the cumulative distribution of hit-times observed for each affected overlay path.

3) Number of route flaps: During a rerouting process, the route used by the overlay link is changed to avoid the failed link. This can happen multiple times and can be the result of the overlay or the native dynamic routing process. Each such change is defined as a *route flap*[6]. Route flaps can be a serious problem in case of TCP and other traffic that relies on packet ordering[26]. It not only increases the packet loss and latency, but also burdens the network with extra path computation overhead. As a result, the network efficiency will be degraded and end-to-end performance is hurt. The route flaps are also an indication of the instability in the system. However, these route flaps are not persistent (i.e., overlay routing converges eventually[20]) and may not impact the traffic any greater than the actual loss of packets. Nevertheless, they trigger oscillations which take a longer time to resolve in the presence of load constraints or traffic engineering.

For our analysis, we assume that all failed native links are repaired at some time instance. Such link repair is first detected by the native layer and then communicated to the overlay layer by means of updated overlay link cost. This repair may also cause another route flap as the network switches back to the original (pre-failure) route.

Each path affected by a native failure can display a different sequence of route flaps, based on the temporal dynamics and the layer yielding the optimal path. Hence, we are interested in the average number of route flaps, calculated as:

$$\text{Average route flaps} = \frac{\text{Number of route flaps}}{\text{Number of failed overlay paths}}$$

In our scenarios, one of the two layers definitely performs rerouting, unless a failure causes the network to be disconnected. A maximum of one route flap occurs following a repair and it depends on the state of the system at the end of the previous rerouting operation.

4) Path cost inflation due to a route flap: Path cost inflation represents the increase in the cost of the path between the overlay nodes at the ends of the overlay link(s) broken as a result of a native link failure. We measure this cost relative to the cost of the original (pre-failure) path used by the overlay link. It conveys the penalties for choosing a longer path. The path cost inflation ratio is defined as:

$$\text{Path cost inflation} = \frac{\text{Path cost after rerouting}}{\text{Path cost before failure}}$$

We are interested in this metric after a sufficient time has elapsed since native link failure but before native link repair. This value shall be referred to as *stabilized inflation*. The dynamic protocols in both layers constantly strive to achieve the least path cost. Hence, after a sufficient time has elapsed the path cost inflation of all affected paths will be the same across different schemes.

Immediately after a failure, path cost inflation starts at ∞ when the ends of the path are disconnected. Once a path is re-established as a result of the rerouting process, a path cost inflation attains a finite value. This inflation may decrease over time as the rerouting process continues. Thus, we are also interested in the maximum (non-infinite) path cost inflation observed per failed overlay path as a result of rerouting. This value, referred to as *peak inflation*, indicates the level of sub-optimality suffered by the overlay path before stabilizing. When the stabilized inflation is constant across different schemes, the

[5]Following standard terminology, an overlay path is made of one or more overlay (virtual) links, which in turn comprises one or more native links.

[6]Also called route oscillation.

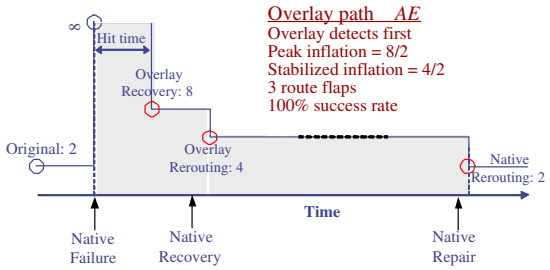


Fig. 2. Sequence of route flaps and the path cost progression for link failure in the example. The numbers indicate path cost in terms of native hop count. In the above figure, the overlay layer detects failure first.

peak inflation acts as a measure of the overall sub-optimality.

We use the notations T_{peak} and T_{stable} to indicate the time instance when the peak inflation and stabilized inflation are observed respectively. T_{stable} represents the time instance when the system attains steady state and should be as small as possible.

We illustrate the dynamics of the performance metrics in Fig. 2 where the numbers indicate the actual overlay path cost, in terms of native layer hop count. Consider the example topology in Fig. 1 where both layers use hop count as the link cost. When the native link CE fails, the overlay link AE is broken. Assume the overlay layer detects the failure first. The link is now overlay-rerouted through G and I leading to one route flap and a sub-optimal path of cost 8. Once the native timers expire, the native path AE is native-rerouted through F and D. The new native path is now the optimal one with a cost of 4. The overlay senses this and adopts the direct link, thereby causing another route flap. The system is now stable until the native link is repaired. Once the native repair is triggered, the native path switches back to the original path as it provides a lower cost of 3. This leads to a third route flap. Thus we get the performance values mentioned in Fig. 2.

IV. Characteristics of Dual Rerouting

Dual Rerouting refers to the scheme where each layer operates completely independent of the other. This independent operation leads to a large number of route flaps that cause wastage of resources[7], affect the performance of traffic that needs accurate packet ordering and lead to general system instability. The overlay path cost in-between the route flaps can be sub-optimal as the overlay network is unaware as to which layer provides the optimal rerouting.

Assuming we have no control over the native layer parameters, these problems in Dual Rerouting can be mitigated by adjusting the values of hold-time, keepAlive-time and cost scheme at the overlay layer. By tuning the timers at the overlay layer, we vary the layer at which failure detection is likely to happen first. We also investigate waiting for fewer *keepAlive* messages without risking any false alarms in failure detection. Section VI presents the effects in detail.

[7]Time for computing new routes and bandwidth overhead for exchanging routing protocol updates.

Past research on resilience suggests adopting a low keepAlive-time at the overlay layer so that it can detect the failure first[1]. This tends to aggravate the problems in Dual Rerouting by instigating both layers to perform rerouting. We investigate the associated negative effects and determine whether an earlier overlay detection is completely beneficial.

Notice that Dual Rerouting has the best hit-time compared to any other scheme as there are two different layers aiming at rerouting around the failed link. Hence, it is not possible to improve the hit-time performance any further. Dual Rerouting, however, performs poorly with respect to the other metrics (viz. number of route flaps and path cost inflation). We, therefore, consider layer-aware schemes that allow us to trade off improvements in these other metrics with longer hit-times.

V. Layer Aware Overlay Rerouting

In this section, we present three native layer aware overlay rerouting schemes - *Probabilistically Suppressed Overlay Rerouting*, *Deferred Overlay Rerouting* and *Follow-on Suppressed Overlay Rerouting*. These schemes require knowledge of the native layer's routing protocol existence and in the case of the last scheme some minimal knowledge of the state of the native layer rerouting efforts.

The layer-aware schemes operate based on the assumption that the native layer cannot be modified. This is because the overlay networks share the native network with other non-overlay applications. The operation of the native network is thus tuned for the native applications. We are, therefore, only allowed to control the overlay layer by suppressing or delaying the rerouting process. It is possible to construct more complex inter-layer coordination and information exchange (e.g., assuming that the overlay layer has knowledge of the native layer topology[22], [27] or of the other coexisting overlay networks in the system[28]). However, this may not be very practical. More importantly, we find that significant control over the tradeoffs between hit-time and the other metrics is possible through the simple approaches we consider.

a) Probabilistically Suppressed Overlay Rerouting: In this scheme, we suppress overlay rerouting without any particular knowledge of the native layer. The suppression operation is done with probability p on each overlay rerouting attempt (irrespective of whether it follows a failure or a repair). Various values of p lead to different recovery performance and are of interest to us. The main advantage to be gained with the suppression operation is the decrease in the number of route flaps. If the value of p is configured appropriately for an overlay path, it is possible to achieve the best path cost values inbetween the route flaps. Profiling the performance for each p will help in choosing the right value for the network in consideration. A value of 0 for p corresponds to the case of Dual Rerouting. When p is 1, all overlay rerouting attempts are suppressed and there is a possibility the overlay path may never be rerouted if the native rerouting does not succeed.

b) Deferred Overlay Rerouting: In this scheme, we delay overlay recovery by a constant value to give the native layer a chance to recover the path. After that time has elapsed, if

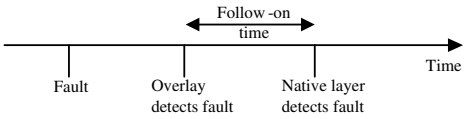


Fig. 3. Possible scenario for failure detection in two layers. The decision in the *Follow-on Suppressed* scheme depends on the amount of time between the overlay detection and the native layer detection.

the native network has not yet recovered, overlay recovery is performed. As the overlay network is made to wait until the native layer rerouting, the system incurs a longer hit-time. The duration by which we delay overlay recovery is determined based on the amount of additional hit-time the overlay traffic can handle. This scheme will have fewer route flaps relative to Dual Rerouting and a longer hit-time.

c) Follow-on Suppressed Overlay Rerouting: In this scheme, the overlay layer keeps track of the native layer's timer values. As the overlay layer hold-timers are independent from that of the native layer's, there is a possibility the native layer keepAlive-timer closely follows that of the overlay layer, relative to the time of failure. The overlay rerouting effort might be wasteful because the native layer recovery is not too far off. The decision of whether to suppress the overlay rerouting depends on the *follow-on time*, which is defined as the time remaining for the next native layer detection. Fig. 3 illustrates the follow-on time which must be known at the overlay layer to avoid undesired recovery and functionality overlap between layers. If the follow-on time is less than a particular threshold value, the overlay rerouting is suppressed. The concept of follow-on time is applicable only when the overlay layer detects the failure first. The threshold value needs to be determined by the amount of additional hit-time the system can tolerate.

The characteristics of this scheme are similar to *Deferred Overlay Rerouting*, except that this scheme has a relatively smaller hit-time. This is because the overlay rerouting is initiated right away in cases where the follow-on time is higher than the threshold, unlike *Deferred Overlay Rerouting* where the overlay layer always waits for the constant delay period anyways. This scheme establishes an upper bound on the hit-time observed.

This scheme requires the knowledge of when the timer expires at the node closely associated with the failure. Obtaining this information is even harder in the multi-domain scenario. We postulate that signaling between layers, in combination with explicit BGP peering, can accomplish this, but the details are outside the scope of this paper. We are interested first in measuring the benefits we derive from this scheme before developing techniques to do it.

VI. Results

We now present NS-2 simulation results for Dual Rerouting and the three layer-aware schemes proposed. The strategies work similarly in single-domain and multi-domain scenarios. Hence, the results are presented together and the differences mentioned when applicable.

A. Simulation Setup

It should be noted that the performance of the rerouting schemes is topology-specific. Hence, we need to simulate multiple overlay topologies over multiple native networks to improve the generality of the results. We use GT-ITM[29] to generate random network topologies for the simulations. We generate 5 native network topologies and 5 overlay topologies at random. The 25 possible combinations generated by mapping the overlay topology to the native topology are simulated and the results averaged over them. The links used in the simulation are bi-directional and each end triggers the detection process once a native link fails. We assume that the native and overlay layers perform symmetric routing. This is reasonable as the average statistics will still be the same.

The intra-domain native paths use hop-count based costs (by setting a native link's cost to 1), while the inter-domain native paths use the length of the AS path. The overlay paths use a native hop count based cost scheme[8].

The next heavily influencing parameter is the hold-time of each layer. To insure consistency of treatment, all solutions use the same timer values at the native layer. We assume here that the timer values are consistent between both ends of any link. The lowest configurable keepAlive-time for commercial routers (such as the Cisco 7200 series router) is 1 sec. We adopt the same value for the intra-domain keepAlive-time. The native layer waits for the absence of three consecutive *keepAlive* messages before declaring a link failure. Hence, the native hold-time is 3 secs. For inter-domain native links, we set the keepAlive-time and hold-time as 5 secs and 15 secs respectively. The native hold-times are close to the practical values used by modern routers[30]. At the instance of a failure, the native detection time is randomly calculated in the range of (hold-time - keepAlive-time, hold-time) and detection is enforced at its expiry.

In the overlay layer, we have complete freedom in configuring the keepAlive-time and the hold-time. As hold-time represents the time period during which no *keepAlive* messages are received, it is a multiple of the keepAlive-time period. In this paper, we consider hold-time being two times the keepAlive-time and three times the keepAlive-time. Waiting for too few *keepAlive* messages can cause the node to mistake congestion effects as failure.

1) Network Topology: For the single-domain native network scenario, each native topology we simulate contains 100 nodes, while each overlay topology contains 10 nodes. The placement of overlay nodes and link connectivity at either layer are decided at random.

For the multi-domain native network scenario, each native topology we simulate contains 500 nodes, while each overlay topology contains 10 nodes. There are 20 stub domains with 24 nodes each and 5 transit domains with 4 nodes each. The connectivity is decided at random. The non-border node with highest edge degree is selected as the overlay node in each stub-

[8]We did not see much difference when experimented with an overlay hop-count based cost scheme.

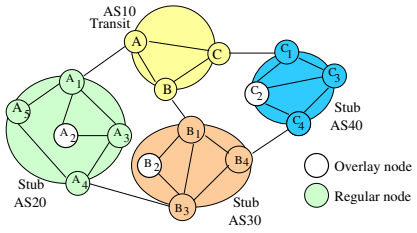


Fig. 4. Advantages in overlay rerouting in the multi-domain scenario. The double-circled nodes represent the overlay node

domain. Thus, no two overlay nodes are in the same domain. The stub networks that host overlay nodes are multi-homed to incorporate redundancy and increase the success rate of rerouting.

We experimented with multiple choices for topology size in both scenarios and got similar results. This implies that the effect of topology size is negligible. Hence, we only present the results for the above mentioned choice.

BGP++ is used for simulating the inter-domain routing dynamics[31][9]. We use the *community* configuration in the simulator to enforce the policy that the private stub-stub links are to be used only for exchanging native traffic between the two stubs. This policy brings in an added advantage for overlay rerouting which will be able to use a redundant stub-stub link. Consider the topology in Fig. 4 where the stub-stub link A_4B_3 can be used only to exchange traffic between the two domains - AS20 and AS30. Initially, the overlay path A_2C_2 passes through AS10. In case of failure in link AA_1, the native network will be unable to reroute the overlay link. This is because it cannot use the private link A_4B_3 due to policy constraints. However, the overlay layer will be able to reroute the traffic over the two overlay links A_2B_2 and B_2C_2. In certain topologies, the overlay-rerouted path can be optimal because it does not have to pass through the transit domain AS10.

2) Link Failure Modeling: Failure modeling is complicated as failure history information is unavailable for the Internet or other overlay networks. To avoid a specific failure model and its set of assumptions, we use a *stateless all-link failure* approach. In this approach, all of the native links (intra-domain links in the case of single-domain and inter-domain links in the case of multi-domain) are failed one at a time and the statistics tabulated for all possible overlay paths. The state of the system is reset before simulating each failure. This helps study the influence of failure in any location of the network and tries to reduce the dependence on the topology. Occasionally, both overlay rerouting and the native rerouting fail. This can happen when the failure of a particular link breaks the native network into more than one component, or when the policy restrictions prevent the native network from using the available gateway routers. Our simulation results do not include such cases.

B. Dual Rerouting

We simulated Dual Rerouting and measured the performance of overlay rerouting based on hit-time, number of route flaps, and path cost inflation. This section presents those results and derives its relation to the hold-time and keepAlive-time.

Note that the values we present in the tables are normalized against those observed for native-only rerouting, unless mentioned otherwise. We expressed these normalized values in terms of percentage.

Fig. 5 shows the cumulative distribution function (CDF) of hit-times experienced by overlay paths in our simulations. Recall that hit-time is made up of two main components - the detection time, which depends on the hold-time and the convergence time, which depends on the dynamics of the routing protocol in consideration[6], [25]. Fig. 5(a) and Fig. 5(b) show that increasing the overlay hold-time gradually increases the observed hit-time until the overlay hold-time is equal to the native hold-time. When this happens, the native layer completes the rerouting first and further increases in overlay hold-time have no effect. Hence, the curves begin to merge for higher values of overlay hold-time.

The overlay hold-time was varied between twice the keepAlive-time (case A, B) and three times the keepAlive-time (case C). Fig. 5(c) shows that case A has the lowest hit-time. When the keepAlive-time is the same, case A has a smaller hold-time. When the hold-time is the same at 9 secs, case B has a bigger detection window of (4.5s, 9s), unlike case C that has a narrower detection window of (6s, 9s). These two factors cause the earlier detection when hold-time is twice the keepAlive-time.

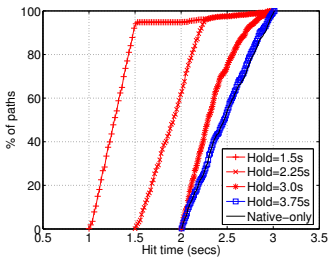
The knee bend in the graphs of Fig. 5 corresponds to the time instance when the overlay layer failure detection is complete. The flat section of each curve corresponds to the idle time where the system waits to perform native rerouting on the unrecovered overlay links.

The average number of route flaps per link failure has been listed in Table I. We also present the percentage of failed overlay paths successfully recovered at a particular layer. The sum of the two values (*% overlay recovered* and *% native recovered*) reflect the success rate of the rerouting scheme. From the table, we can observe that the overlay layer can recover a maximum of 95.8% of the paths in the single-domain scenario and 94.2% in the multi-domain scenario. We also observe that native-only rerouting has a success rate of 100% in the single-domain scenario, in contrast to a 78.8% success rate in the multi-domain scenario. This demonstrates the importance of dynamic overlay rerouting.

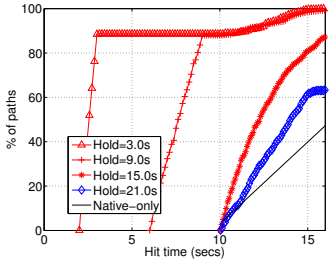
We also infer from Table I and Fig. 5 that the overlay hold-time is inversely proportional to the number of route flaps, while being directly proportional to the hit-time. This shows a clear tradeoff between the number of route flaps and hit-time.

Table II shows the average values of path cost inflation for different overlay hold-times. We see from the table that the hold-time does not have a direct influence on the individual path cost inflation. By regulating the proportion of overlay rerouting and native rerouting, hold-time controls the extent of inflation. Hence, the value of peak inflation decreases with an increase in hold-time. We also see from the peak inflation value that native-only rerouting is the best, though it tends to have a

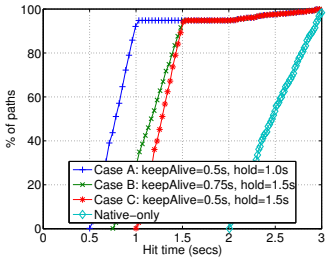
[9]The overlay nodes in our simulation do not try to explicitly peer with any BGP router and are hence oblivious to the inter-domain dynamics.



(a) Single-domain
(hold-time=3*keepAlive-time)



(b) Multi-domain
(hold-time=3*keepAlive-time)



(c) Single-domain
(Comparing keepAlive schemes)

Fig. 5. Cumulative distribution of hit-time for different overlay timer values, when path cost is native-hops

TABLE I

AVERAGE NUMBER OF ROUTE FLAPS

—**Single-domain**—

	Hold=1.5s	**Hold=3.0s**	**Hold=4.5s**	**Native-only**
% overlay recovered	95.8	45.0	0	0
% native recovered	4.2	55.0	100	100
Average route flaps	140.58%	125.08%	109.57%	1.567
	(Normalized by native-only)			(Absolute)

—**Multi-domain**—

	Hold=9s	**Hold=15s**	**Hold=21s**	**Native-only**
% overlay recovered	94.2	52.2	21.4	0
% native recovered	5.8	47.8	78.6	78.8
Average route flaps	157.25%	153.60%	150.53%	1.207
	(Normalized by native-only)			(Absolute)

TABLE II

AVERAGE PATH COST INFLATION

—**Single-domain**—

Hold-time	**1.5s**	**3.0s**	**4.5s**	**Native-only**
	(Normalized by native-only)			(Absolute)
Stabilized inflation	Equal value: 100%			1.202
T_{stable} (secs)	97.98%	113.70%	107.25%	2.481
Peak inflation	130.28%	114.22%	100%	1.202
T_{peak} (secs)	53.44%	94.23%	100%	2.481

—**Multi-domain**—

Hold-time	**9s**	**15s**	**21s**	**Native-only**
	(Normalized by native-only)			(Absolute)
Stabilized inflation	Equal value: 108.59%			1.117
T_{stable} (secs)	90.81%	114.66%	125.46%	12.41
Peak inflation	134.29%	117.81%	106.71%	1.117
T_{peak} (secs)	67.22%	98.39%	108.78%	12.41

low success rate in the multi-domain scenario. The stabilized inflation for different hold-times are almost equal, implying that Dual Rerouting attains the same inflation in steady state.

In the single-domain scenario, the overlay paths at the end of native link repair were noticed to be of the same length as the original path. But, this is not true in the case of multi-domain scenario where the final path cost was different than the original one. This is because the native network computes paths using the AS path length and not hop count. Two AS paths of equal length can have different number of hops as it does not publicize the internal routes. Hence, the cost ratio does not necessarily fall back to 1 after link repair.

Table II also presents results for the time instance of peak inflation and stabilized inflation. The table shows that, for both single-domain and multi-domain scenarios, the T_{peak} is closely related to the overlay hold-time. This confirms that overlay rerouting is what causes the peak inflation. In single-domain scenarios, the T_{stable} is reduced when the overlay hold-time exceeds native hold-time because the native layer rerouting typically occurs first and provides the stabilized inflation. But, this is not observed in multi-domain scenarios as a certain proportion of overlay paths achieve their least cost route with overlay rerouting.

We also computed the 95% confidence interval for each of the statistics presented in this paper. We noticed that the confidence interval stretches to a maximum of $\pm 4\%$ of the mean value.

Summary of factors affecting Dual Rerouting:

From the above simulations, we observe that native rerouting provides the optimal alternate path in both the single-domain and multi-domain scenarios, though it suffers from a low success rate in the latter case. Thus, we can obtain lower number of route flaps and lower peak inflation by giving a higher precedence to the native rerouting attempts. We adopt this form of control as the fundamental strategy behind the design of the layer-aware schemes, where the tradeoffs between the performance metrics can be controlled by the three parameters - *probability of suppression*, *delay* and *follow-on threshold*.

Typically, the overlay timers are calculated based on the desired amount of routing protocol overhead and packet loss during failure. Performance of Dual Rerouting can be substantially improved by adjusting just the value of overlay hold-time, which has the same effect as suppressing the overlay rerouting operation. The following choices are recommended for achieving optimal performance with Dual Rerouting:

- Overlay hold-time value very close to the native hold-time, irrespective of the keepAlive-time chosen.
- Declare failure after the absence of two $keepAlive$ messages, rather than three.

In the rest of the section we discuss layer-aware rerouting schemes. They use the above observation of Dual Rerouting as the base line to enhance its performance. Hence, the keepAlive-time for native and overlay were set to the same value (1 sec in the single-domain scenario and 5 secs in the multi-domain scenario). This implies that native rerouting can lag overlay rerouting only by a maximum value equal to the keepAlive-time. The hold-time for both layers were set to three times the keepAlive-time of that layer.

TABLE III

PERFORMANCE ANALYSIS OF PROBABILISTICALLY SUPPRESSED OVERLAY REROUTING

Probability of suppression	**0.25**	**0.5**	**0.75**	**Native-only** **1.0** (Absolute)
	(Normalized by native-only)			
—Single-domain—				
Avg. hit-time for recovered paths	95.60%	96.89%	98.50%	2.481 secs
Stabilized inflation	103.24%	104.24%	102.99%	1.202
Peak inflation	112.48%	110.00%	109.23%	1.202
—Multi-domain—				
Avg. hit-time for recovered paths	98.94%	99.18%	100%	12.405 secs
Success rate	92.8%	84.1%	79.5%	78.8%
Stabilized inflation	111.19%	108.68%	105.37%	1.117
Peak inflation	115.04%	110.74%	105.82%	1.117

C. Probabilistically Suppressed Overlay Rerouting

The main parameter we can control in this scheme is the value of p. Simulation results, obtained by suppressing all the overlay rerouting operations with a constant probability, have been tabulated in Table III. The table shows that the hit-time increases on increasing the probability p for suppressing. This scheme suffers a higher hit-time than Dual Rerouting because some of the earlier overlay recovery attempts are disabled and the failed paths are made to wait longer. The hit-time is observed to be in-between native-only and Dual Rerouting.

As the suppression probability increases, some of the overlay rerouting operations required to achieve the optimal path may be suppressed. Hence, the stabilized inflation tends to decrease, as can be seen from the table. Table III also shows that the peak inflation decreases gradually with an increase in p because more native rerouting operations, that yield shorter paths, are performed. In the multi-domain scenario, we see that the success rate decreases almost linearly with an increase in p. This is because the native layer was unable to recover the path after overlay rerouting was completely suppressed. We do not present the values for the number of route flaps as the results are straightforward - increasing the probability of suppression causes a decrease in the number of route flaps. The above mentioned trends represent the tradeoffs that can be used to select the value of p.

D. Deferred Overlay Rerouting and Follow-on Suppressed Overlay Rerouting

The *Follow-on Suppressed Overlay Rerouting* tends to reroute paths in the overlay layer right away, if the follow-on time is higher than the threshold. Thus, *Follow-on Suppressed Overlay Rerouting* has a smaller hit-time compared to *Deferred Overlay Rerouting*. Intuitively, the hit-time increases with an increase in delay or follow-on threshold because the network has to wait for the native recovery to be initiated. The hit-times of both these schemes are higher than in Dual Rerouting as some of the overlay recovery operations are suppressed or postponed. As no overlay rerouting is suppressed indefinitely, these schemes have a 100% success rate and obtain the optimal path cost eventually.

Both schemes have similar characteristics in terms of route flaps and path cost inflation, as they perform exactly the same sequence of overlay rerouting operations (albeit at a slightly different time). The performance of the *Deferred Overlay Rerouting* scheme for different delay values is tabulated in Table IV. From the table, it can be seen that a higher delay implies a reduced number of overlay rerouting attempts and thereby shorter paths. Hence, the peak inflation decreases with an increase in the delay. As with the previous scheme, we do not present the values for the number of route flaps as the results are straightforward - increasing the delay causes a decrease in the number of route flaps.

TABLE IV

PERFORMANCE ANALYSIS OF DEFERRED OVERLAY REROUTING

Delay	**0.250s**	**0.375s**	**0.5s**	**Native-only** (Absolute)
	(Normalized by native-only)			
—Single-domain—				
Avg. hit-time for recovered paths	97.70%	98.75%	99.43%	2.481
Peak inflation	111.89%	110.73%	108.48%	1.202
—Multi-domain—				
Delay	**1.5s**	**2.0s**	**2.5s**	**Native-only** (Absolute)
	(Normalized by native-only)			
Avg. hit-time for recovered paths	104.24%	106.05%	107.75%	12.405
Peak inflation	115.66%	114.32%	113.60%	1.117

E. Performance Comparison

This section compares the performance of Dual Rerouting, *Probabilistically Suppressed Overlay Rerouting*, *Deferred Overlay Rerouting*, and *Follow-on Suppressed Overlay Rerouting*. We set the overlay layer hold-time to the same value as the native layer hold-time. The delay and follow-on threshold were set to 0.375 secs or 2 secs depending on the scenario and the suppression probability was set to 0.5.

Table V shows the route flap and path cost inflation statistics for the layer-aware schemes and the native-only rerouting scheme. Among the layer-aware schemes, we observe from the table that *Probabilistically Suppressed Overlay Rerouting* has the lowest route flaps, because it suppresses more overlay rerouting operations. For the same reason, it has lower peak inflation value. *Deferred Overlay Rerouting* does not suppress any overlay rerouting operation after traffic recovery. Hence, it has the lowest stabilized inflation. *Deferred Overlay Rerouting* also has a higher success rate because we will never eliminate overlay rerouting completely.

Based on Table V, we can not comment on any particular relation between the different values of T_{stable} or T_{peak} because the actual overlay rerouting operations that were suppressed were random. However, the results are consistent with the following two observations. 1) Native-only rerouting must attain steady state much faster than the other schemes. 2) Dual Rerouting, which has no suppressed overlay rerouting operations, must attain the peak earliest.

Fig. 6 shows the CDF of hit-time for the three rerouting schemes. Dual Rerouting is always the best as there are more overlay rerouting operations trying to recover the failed path. This comes at the expense of more route flapping during recovery as shown in Table V. The curves for *Probabilistically Suppressed Overlay Rerouting* and *Follow-on Suppressed Overlay Rerouting* closely follow each other because they recover an approximately equal proportion of the failed links in the overlay layer. *Deferred Overlay Rerouting* has higher hit-time

TABLE V

COMPARISON OF ALL REROUTING SCHEMES

Type of rerouting	Dual Rerouting	Suppressed	Deferred/ Follow-on	Native-only
	(Normalized by native-only)			(Absolute)
—Single-domain—				
Average route flaps	125.08%	101.59%	109.85%	1.567
Stabilized inflation	100%	108.32%	100%	1.202
T_{stable} (secs)	113.70%	100.48%	107.33%	2.481
Peak inflation	114.22%	109.98%	110.73%	1.202
T_{peak} (secs)	94.23%	96.89%	98.75%	2.481
—Multi-domain—				
Average route flaps	153.60%	114.00%	146.56%	1.207
Success rate	100%	84.1%	100%	78.8%
Stabilized inflation	108.59%	112.23%	108.59%	1.117
T_{stable} (secs)	114.69%	104.93%	117.42%	12.405
Peak inflation	117.81%	110.74%	114.32%	1.117
T_{peak} (secs)	98.45%	99.18%	106.05%	12.405

as it delays the overlay rerouting irrespective of the subsequent native rerouting.

Summary of performance of layer-aware schemes:

None of the three layer-aware schemes are the best according to all of the four performance metrics, but they provide the best hybrid situation. Based on whether the system is sensitive to hit-time, route flap or path cost inflation a different scheme can be chosen. Choosing the right scheme also depends on the amount of awareness the overlay layer has. In certain cases, Dual Rerouting can be made to perform best by varying the hold-time and keepAlive-time.

We notice that *Follow-on Suppressed Overlay Rerouting* does not provide a substantial advantage over *Deferred Overlay Rerouting*. Hence, the increase in implementation complexity is not justified. In most cases, the other two simple layer-aware schemes are capable of providing the required control.

The exact degree of control can be varied by tuning the five parameters - namely keepAlive-time, hold-time, suppression probability, delay and follow-on threshold. Tuning these parameters in a real-life overlay network requires the operator to perform a tradeoff analysis similar to our methodology and make a multi-objective decision for the appropriate traffic type. However, we are unable to posit a general rule of thumb that can be applied widely. This is because the importance of each metric is unequal and is specific to the type of traffic. In most situations, packet loss has a more serious effect on the performance of the overlay traffic and therefore reducing the hit-time should be given a high precedence. Hence, we can conclude that Dual Rerouting, despite its problems with the functionality overlap, can be considered to achieve the most desirable performance. This shows that the practical limitations brought by the lack of information cannot be circumvented. We recommend tuning of the overlay layer timer values, in combination with vanilla Dual Rerouting, as the best rerouting scheme in most systems.

VII. TUNING NATIVE LAYER PARAMETERS

A. Motivation

The previous sections aimed at improving the performance of overlay rerouting by assuming that we cannot change the parameters of the native layer routing protocol. That is the most pragmatic approach to the problem. In this section, we investigate ways to improve the performance by adjusting the operation of the native layer.

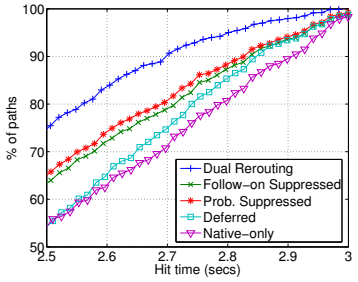


Fig. 6. Hit-time comparison of all schemes in the single-domain case

We envision that as overlay applications proliferate, the native layer should gradually evolve to suit the overlay network requirements. Our work is further motivated by the results from the previous sections that show that changing the parameters of the overlay layer only yields modest gains.

The rest of the section illustrates how one can tune the native layer keepAlive-time in the best interest of the overlay network, without causing much overhead or harm to non-overlay traffic.

B. Tuning the keepAlive-time

In Section I, we highlighted the fact that a dynamic routing protocol is essential in the overlay layer to significantly enhance the overlay networks survivability. However, in cases where both the native and overlay layer were capable of rerouting around a link failure, the native layer rerouting was shown to be the optimal one in terms of path cost inflation and number of route flaps, as seen from the results in Section VI (See the results for route flaps and path cost inflation of Dual Rerouting when the overlay hold-time is bigger than the native hold-time).

In Dual Rerouting, insuring that recovery will take place at the native layer first can be achieved in multiple ways:

- Using a device or hardware notification to trigger the native rerouting.
- Adding an overlay-to-native signaling protocol to jump-start the native rerouting as soon as the overlay layer detects a failure.
- Setting the native layer's keepAlive-time to a value much smaller than that at the overlay layer.

The first option is not always possible due to limitation of the physical layer. The second option is an invasive procedure that requires alteration of the native layer code to support the new feature. Hence, we do not see it as a feasible solution. The last option of tuning the native layer timer value is the most feasible. We do not need to rebuild the native network and the functions used remain fundamentally the same. However, we need to adhere to the following two constraints while tuning the keepAlive-timers:

1) The tuning should not generate any extra overhead.
2) The effective detection time (defined earlier as the smaller of the detection times at either layer) should be the same.

The keepAlive-time for each layer is typically chosen based on the amount of *keepAlive* message overhead the layer can

deal with[30], [32] and the amount of hit-time the system can tolerate in the event of a failure. This indicates a tradeoff that exists between the responsiveness to a failure and the protocol overhead. We define the routing protocol overhead as the number of *keepAlive* packets sent per second on the link under consideration. The sum of protocol overhead in each native link, contributed by both the layers, represents the overall protocol overhead. The overall protocol overhead is calculated as:

$$\text{Overall protocol overhead} = \frac{\text{Number of native links}}{\text{Native keepAlive-time}} + \sum_{\text{Overlay link}} \left(\frac{\text{Native hop count of the overlay link}}{\text{Overlay keepAlive-time}}\right) \quad (1)$$

It is widely believed that the overlay network can reroute traffic around native failures earlier than the native network[1]. The main reason is that the native layer recovery time can be long due to the reduced periodicity of *keepAlive* messages to conserve native routing protocol overhead[32].

This earlier recovery at the overlay can be achieved by setting the keepAlive-time to be relatively short. The additional overlay routing protocol overhead incurred is considered to be manageable because the overlay network tends to have fewer nodes and links. However, this is a flawed argument. This can be seen by examining our estimates of the routing protocol overhead at the two layers. We noticed that the overhead incurred at the native and overlay layers are quite comparable. This can be attributed to the following three factors:

- Each overlay link is comprised of multiple native links.
- The overlay network has higher degree of link connectivity than the native network.
- There can be multiple coexisting overlay networks on the same native topology.

The significance of the above reasons can be verified from the expression for overall routing protocol overhead in Equation (1). The fact that the protocol overhead in the two layers are comparable leads us to revisit the choices of native and overlay keepAlive-time.

We consider the effect of decreasing the native layer keepAlive-time parameter. This can cause an increase in the overall routing protocol overhead. We offset that by increasing the overlay layer keepAlive-time value, which ultimately keeps the overall routing protocol overhead for both the overlay and native network under control. Maintaining the protocol overhead at the same value is possible because the overall protocol overhead can be the same for multiple pairs of native and overlay keepAlive-time. Fig. 7 shows such tuples for a particular topology with 20 overlay nodes and 100 native nodes. Each point on the curve gives us the keepAlive-time setting for the two layers (X-coordinate value for overlay layer and Y-coordinate value for native layer) so that the desired overall routing protocol overhead can be obtained.

Fig. 8 presents multiple scenarios created by differing protocol timer values for a single overlay network, with a dynamic link state routing protocol, running on top of a single-domain native topology. The amount of overhead in each layer is

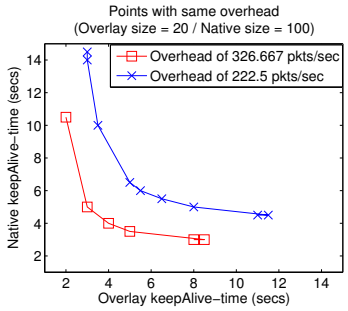


Fig. 7. Possible choices for keepAlive-times when protocol overhead is same.

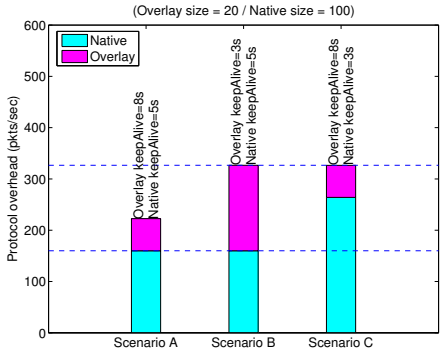


Fig. 8. Possible scenarios with multi-layer protocol overhead. Scenario B→A leads to optimal paths, fewer route flaps and conserves overall routing protocol overhead. Scenario B→C is what we recommend here.

different for each scenario. Scenario A has a longer overlay keepAlive-time, but the same native keepAlive-time as Scenario B. As a consequence, the total routing protocol overhead incurred by Scenario B is higher. However, the link failure detection time at the overlay layer is smaller in Scenario B. Our results have shown that it is not necessarily desirable for the overlay layer to detect the failure first. It seems, therefore, that Scenario A is more desirable because it insures that the recovery takes place at the native layer first and it incurs less overall protocol overhead.

The assumption that we can change native network parameters, however, makes a third scenario (Scenario C) possible. In this scenario, the overlay keepAlive-time is kept the same as Scenario A, but the native keepAlive-time is decreased so as to trigger an earlier failure detection. The overall routing protocol overhead is the same as Scenario B. But, in Scenario C, we have the desirable property that the native network is able to detect link failures first[10].

The problem of determining the keepAlive-timer values has traditionally been concerned with identifying the right mix of protocol overhead and detection time. We extend the problem by highlighting the need to have the right ratio between native and overlay keepAlive-time, while keeping our solution conservative (same overall protocol overhead and the same effective

[10]There is a possibility that the *keepAlive* messages of the overlay layer are lost due to momentary congestion in the intermediate routers leading to false alarms. But, this does not happen with the native layer as they are exchanged at a higher priority in most networks. Thus, we have an added advantage of lower false alarms with having the native layer detect the failure first.

TABLE VI

PERFORMANCE GAIN WITH NATIVE LAYER TUNING

(OVERLAY SIZE = 20 / NATIVE SIZE = 100)

	Overlay keepAlive	Native keepAlive	Avg. hit-time (s)	Avg. num route flap	T_{stable}	Peak inflation
A	8	5	12.72	1.571	14.67	1.168
B	3	5	9.166	1.610	11.61	1.247
C	8	3	7.793	1.571	9.739	1.168

detection time).

Table VI presents the relevant results for the three scenarios mentioned (averaged over 25 random single-domain topologies). The native layer tuning we proposed achieves the best performance in all our metrics (viz. hit-time, number of route flaps and path cost inflation), as can be seen from the results in the third row (Scenario C). The value of T_{stable} indicates that the system reaches an earlier stabilization of the paths at the optimal cost.

Summary of performance gain with native layer tuning:

Based on the observations from Fig. 8 and Table VI, we conclude the following when the native network is used to support overlay services:

- The native layer keepAlive-time can be substantially reduced without much negative impact. This helps Dual Rerouting achieve the best performance in all our metrics.
- The increase in the routing protocol overhead of the native layer, caused by reducing the keepAlive-time at the native layer, can be offset by longer keepAlive-time at the overlay layer. This maintains the overall routing overhead the same.
- Reducing the keepAlive-time at the native layer also benefits the non-overlay applications sharing it.

VIII. CONCLUDING REMARKS

In this paper, we investigated a mixed routing environment in which an overlay network deploys a dynamic overlay routing protocol on top of an existing native network dynamic routing protocol. We focused on the interaction between the two routing layers and their performance when rerouting around native link failures. The Dual Rerouting scheme, in which the layers operate independently, was sub-optimal in terms of the number of route flaps and the overlay path cost inflation, though it was able to provide the fastest path recovery. To reduce the sub-optimalities, layer awareness is crucial. We considered three schemes for intelligent overlay rerouting in the specialized layer-aware overlay network. The *Follow-on Suppressed Overlay Rerouting* and *Deferred Overlay Rerouting* schemes perform the best in terms of path cost inflation and success rate. The *Probabilistically Suppressed Overlay Rerouting* has the least number of route flaps. By using *Follow-on Suppressed Overlay Rerouting* scheme in networks that allow the overlay layer to access the native layer attributes, we can obtain hit-time lower than *Deferred Overlay Rerouting*. While more complex layer-aware rerouting schemes are possible, our work shows that the relatively simple schemes we consider provide us with sufficient flexibility to control the tradeoffs between the various rerouting performance metrics. Finally, we make the case for tuning of the native layer routing protocol to circumvent the limitations of an uncooperative lower layer. In particular, we recommend tuning the keepAlive-time of the native layer to achieve the best possible rerouting performance.

REFERENCES

[1] D. Andersen et al., "Resilient Overlay Networks," in *Proceedings of 18th ACM SOSP*, October 2001.

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

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

[4] J. Touch, "Dynamic Internet Overlay Deployment and Management Using the X-Bone," *Computer Networks*, pp. 117–135, July 2001.

[5] A. Markopoulou et al., "Characterization of Failures in an IP Backbone," in *Proceedings of IEEE INFOCOM*, April 2004.

[6] John T. Moy, *Ospf: Anatomy of An Internet Routing Protocol*, Addison Wesley Professional, 1998.

[7] Y. Rekhter and T. Li, "A Border Gateway Protocol 4 (BGP-4)," RFC 1771, March 1995.

[8] M. Kodialam et al., "Dynamic routing of bandwidth guaranteed tunnels with restoration," in *Proceedings of IEEE INFOCOM*, March 2000.

[9] L. Sahasrabuddhe, S. Ramamurthy, and B. Mukherjee, "Fault Management in IP-Over-WDM Networks: WDM Protection vs. IP Restoration," *IEEE Journal on Selected Areas in Communications*, January 2002.

[10] A. Fumagalli et al., "IP Restoration vs. WDM Protection: Is There an Optimal Choice?," *IEEE Network Magazine*, pp. 34–41, Nov. 2000.

[11] A. Autenrieth et al., "Evaluation of multi-layer recovery strategies," in *Proceeding of Eunice 98 Open*, September 1998, pp. 33–42.

[12] P. Demeester et al., "PANEL - Protection across network layers," in *Proceeding of ECOC*, June 1997.

[13] J. Vasseur et al., *Network recovery: Protection and Restoration of Optical, SONET-SDH, IP, and MPLS*, Morgan Kaufmann, July 2004.

[14] B. Davie and Y. Rekhter, *MPLS: Technology and Applications*, San Francisco, CA, Morgan Kaufmann, 2000.

[15] G. Mohan, C. Siva Ram Murthy, and A. Somani, "Efficient algorithms for routing dependable connections in WDM optical networks," *IEEE/ACM Transactions on Networking*, pp. 553–566, October 2001.

[16] Y. Xiong and L. Mason, "Restoration Strategies and Spare Capacity Requirements in Self-healing ATM Networks," *IEEE/ACM Transactions on Networking*, pp. 98–110, February 1999.

[17] W. Cui, I. Stoica, , and R. Katz, "Backup Path Allocation based on a Link Failure Probability Model in Overlay Networks," in *Proceedings of ICNP*, November 2002.

[18] S. Zhuang, D. Geels, I. Stoica, and R. Katz, "On Failure Detection Algorithms in Overlay Networks," in *Proceedings of INFOCOM*, 2005.

[19] R. Keralapura et al., "Can ISPs take the heat from Overlay Networks?," in *Proceedings of ACM HotNets-III*, November 2004.

[20] Y. Liu et al., "On the Interaction Between Overlay Routing and Traffic Engineering," in *Proceedings of INFOCOM*, 2005.

[21] L.Qiu, R.Y.Yang, Y.Zhang, and S. Shenker, "On Selfish Routing in Internet-Like Environments," in *Proceedings of ACM SIGCOMM*, 2003.

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

[23] W. Simpson, "IP in IP tunneling," RFC 1853, October 1995.

[24] Scott Poretsky, "Considerations for Benchmarking IGP Data Plane Route Convergence," Work in progress, Internet Draft draft-ietf-bmwg-igp-dataplane-conv-app-05.txt, February 2005.

[25] C. Labovitz et al., "Delayed Internet routing convergence," *IEEE/ACM Transactions on Networking*, pp. 293–306, June 2001.

[26] E. Blanton and M. Allman, "On making tcp more robust to packet reordering," *ACM Computer Communication Review*, January 2002.

[27] Y. Chen et al., "Algebraic Approach to Practical and Scalable Overlay Network Monitoring," in *Proceedings of ACM SIGCOMM*, 2004.

[28] M. Kwon and S. Fahmy, "Toward Cooperative Inter-overlay Networks," in *Proceedings of ICNP*, November 2003.

[29] K. Calvert, M. Doar, and E. Zegura, "Modeling Internet Topology," *IEEE Communications Magazine*, June 1997.

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

[31] X. Dimitropoulos et al., "Creating Realistic BGP Models," in *Proceedings of IEEE/ACM Intl. Symposium on Modeling*, October 2003.

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