Resource Aggregation for Fault Tolerance in Integrated Services Networks

Constantinos Dovrolis Parameswaran Ramanathan
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
Madison, WI 53706–1691
{dovrolis,parmesh}@ece.wisc.edu

Abstract

For several real-time applications it is critical that the failure of a network component does not lead to unexpected termination or long disruption of service. In this paper, we propose a scheme called RAFT (Resource Aggregation for Fault Tolerance) that guarantees recovery in a timely and resource-efficient manner. RAFT is presented in the framework of the Reliable Backbone (RBone), a virtual network layered on top of an integrated services network. Applications can request fault tolerance against RBone link and node failures. The basic idea of RAFT is to setup every fault tolerant flow along a secondary path that serves as a backup in case the primary path fails. The secondary path resource reservations are aggregated whenever possible to reduce the overhead of providing fault tolerance. We show that the RSVP resource reservation protocol can support RAFT with simple extensions.

1 Introduction

Faults occur in communication networks due to physical cable cuts, router or switch crashes, hardware or software implementation bugs, power failures, natural disasters, and maintenance or operator errors [14]. Telephone network providers have long realized the importance of *survivability* in the presence of faults and they have incorporated sophisticated techniques to achieve very high availability and reliability goals. For example, the AT&T switching systems are designed with an availability of less than two hours of downtime in 40 years, and with a reliability of less than 0.01% of incorrectly handled calls [18]. Survivability has also been a major issue in the design of packet-switched networks. It was identified as the second most important goal in the design of the Internet protocol suite [6].

Integrated services networks will support distributed real-time multimedia applications that range from packet telephony and videoconferencing, to scientific visualization and virtual reality. Two important characteristics distinguish these applications from more conventional ones. First, the duration of the network flows that such applications generate is several minutes to hours, as opposed to the duration of an FTP flow, for example, that is a few seconds. Longer duration means increased likelihood that the flow will encounter a failure of a network component. Second, some of these applications cannot tolerate service disruptions because of their importance and/or cost. For example, dropping the network flow of a remote medical or industrial operation can be life threatening. Similarly, frequent disruptions in a commercial video-on-demand service can lead to loss of customers. Also, if the network charges for these applications, the users will demand a high reliability, similar to that of the telephone service. In this paper, applications with the above characteristics are called *Reliability Sensitive* (RS). Integrated services networks must be able to provide RS applications with a higher degree of reliability, at possibly extra cost. This additional service can be viewed as an extension to the basic QoS, called *Reliability-of-Service* (RoS). We refer to the network flows that are generated by RS applications as *FT-flows* (fault-tolerant flows).

The network fault-management schemes can be broadly classified in two ways. First, there are reactive and proactive schemes. *Reactive* schemes deal with a fault only after the fault occurrence [4]. A typical recovery action is to reallocate

The work reported here is supported in part by the National Science Foundation grants MIP-9526761.

the unreserved network resources among the flows that were affected by the fault. The drawbacks of this approach are, first, that the amount of unreserved resources may not be adequate and some flows may have to be rejected, and second, that the recovery latency can be several seconds or even longer[1], especially in heavily loaded networks. *Proactive* schemes reserve some resources a priori solely for the purpose of facilitating recovery from possible faults [3]. The key advantages of proactive schemes are that flows are not normally rejected and the service disruption period is less than in reactive schemes. However, the resources that are reserved for fault-recovery cannot be granted to other QoS-based flows (although they can be used by best-effort flows). Consequently, given the same amount of network resources, proactive schemes usually result in a lower network utilization than reactive schemes. Fault-management schemes can also be classified as local or global, depending on the locality of the recovery process. In *local* schemes the re-routing of the affected flows is performed by the nodes that are adjacent to the failed network component, while in *global* schemes the affected flows are re-routed on an end-to-end basis. Local schemes are simpler and in some cases faster [22], but global schemes can be more effective in recovering all the affected flows and usually they achieve recovery using less resources [2].

In this paper, we propose a proactive and global fault-management scheme, called RAFT (Resource Aggregation for Fault Tolerance). RAFT guarantees that a FT-flow will not be rejected after a network fault. A major characteristic of RAFT is that it reduces the amount of resources reserved for fault tolerance. Specifically, two disjoint paths are associated with each FT-flow. The first, called *primary path*, is used in the fault-free case. The flow is switched to the *secondary path* when a component in its primary path fails. RAFT aggregates the resource reservations along the secondary path of a FT-flow with the secondary path reservations of other FT-flows, if it is certain that the shared resources will not be used by these flows simultaneously.

[1]The recovery latency depends strongly on the layer that performs the recovery. For physical-layer mechanisms the recovery can take just a few tens of milliseconds, while for network-layer mechanisms it can take up to several seconds [13]. In current IP routers a typical recovery delay is around 30 seconds, mainly due to the large fault detection delay [10]. In a QoS-capable network, however, much faster fault detection is possible (see Section 2).

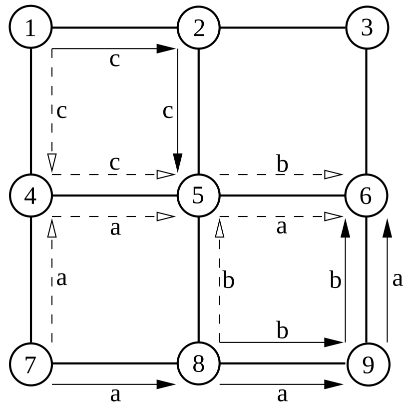


Figure 1: The primary (solid arrows) and secondary (dashed arrows) paths of flows a, b, and c.

The basic idea is illustrated in the simple network of Figure 1. There are three FT-flows a, b, and c. The primary and the secondary paths of these flows are shown with solid and dashed arrows, respectively. Based on the flow QoS requirements, a certain amount of network resources (e.g., bandwidth, buffer space) is reserved on the output ports of the primary and secondary paths of each flow[2]. In this simplified example we assume that only a single network link can fail. The interesting cases of node failures and of multiple component failures are addressed later.

Consider the port (4,5). This port is used in the secondary paths of flows a and c. The straightforward approach is to reserve the *sum* of the bandwidth requirements of a and c on port (4,5). However, because the primary paths of flows a and c do not share any link, a link failure can cause at most one of these flows to be switched to its secondary path. Thus, flows a and c cannot simultaneously use port (4,5), and therefore, they can share the reserved bandwidth on this port. Hence, the bandwidth that has to be reserved on port (4,5) is the *maximum* of the bandwidth requirements of flows a and c. Such aggregation of secondary path reservations reduces the resource overhead for providing fault tolerance. Consider now the port (5,6) that is used in the secondary paths of flows a and b. Since the primary paths of these flows use link (8,9), if this link fails both flows will be switched to their secondary paths. Hence, the bandwidth that must be reserved on

[2]An output port corresponds to a directed link between two nodes. It is used here as an abstraction of the entity that manages the resource reservations.

port (5,6) is the sum of the bandwidth requirements of flows a and b.

The rest of this paper is organized as follows. The network model and our assumptions are presented in Section 2. Section 3 presents RAFT formally, while the RSVP extensions to support RAFT are discussed in Section 4. A quantitative evaluation of the performance of RAFT is presented in Section 5. In Section 6 we compare RAFT with other fault-management schemes. We conclude in Section 7.

2 The Reliable Backbone

In this section we present the network model, the protocols that are related with RAFT and the underlying assumptions for their operation, the fault model that we consider, and the corresponding RoS classes.

RBone. We consider a logical network called *Reliable Backbone* (RBone), layered on top of an integrated services network, that is able to provide RoS guarantees to RS applications[3]. The interconnect between two RBone nodes is a logical link comprised of one or more network elements (physical links, routers, or other subnetworks). An RBone link fails when any of its constituent components fails. Consequently, there may be several fault scenarios which result in the same RBone link failure. The RBone topology must have at least two physically node-disjoint paths between any two RBone nodes. Most current Internet backbone providers design their networks such that this assumption is satisfied (for example, see the UUNET backbone [21]).

RBone Routing Protocol. The RBone uses a routing protocol that provides two node-disjoint paths between any pair of RBone nodes. Such routing protocols have been proposed elsewhere [7, 11], and they can be used by the RBone to support RAFT, as well as QoS-routing, congestion control and load balancing, or other fault-management methods that may coexist with RAFT.

Fault Detection. Fault-management schemes require a fast and reliable way of detecting network faults. This can be achieved with the fault detection mechanisms of current routing protocols, that are based on the periodic exchange of 'KEEPALIVE' messages between neighboring nodes [19]. Since the RBone is layered on top of an integrated services network, the 'KEEPALIVE' messages can be exchanged with guaranteed QoS to ensure fast and reliable delivery. In this way, the fault detection latency can be decreased to just a few seconds or even less, depending on the lower layer technologies.

QoS Guarantees. It is reasonable that a flow with RoS requirements will also have strict QoS requirements regarding its delay, throughput, or packet losses. The IETF Guaranteed service class can meet such QoS requirements reserving bandwidth and buffer space for a flow along its path [20]. In this paper, we consider for simplicity that the only resource that is being reserved is bandwidth. It is straightforward to extend RAFT for buffer space reservations also.

Resource Reservation Protocol. It is assumed that a resource reservation protocol, such as RSVP [5], is used for setting up, maintaining, and terminating the reservations of FT-flows. In Section 4 we show that RSVP can support the protocol functionality required by RAFT with simple extensions.

Fixed FT-Flow Paths. RAFT requires that the two paths of a FT-flow are fixed throughout the flow's lifetime. This is true in a connection-oriented network. In connectionless networks, where the path of a flow can change dynamically, the recently suggested technique of "path pinning" can be used to fix the paths of FT-flows [8]. Note that, regular flows can still use dynamic path changes (as in [5]) and/or other fault recovery methods.

RBone Fault Model. We consider non-coincident RBone link and node failures. The term "non-coincident" means here that the time between successive failures is large enough for the RBone to recover after each failure. This is clarified next. The RBone is in the *normal state* when the required resource reservations for every FT-flow have been setup along both the primary and the secondary paths. After a component failure some FT-flows are switched to their secondary paths (*victim FT-flows*), and the RBone switches

[3] Similar to the Multicast Backbone (MBone) that currently provides multicast services over the Internet.

to the *recovery state*. In the recovery state, the RBone may not be able to tolerate additional component failures. The reasons for this are twofold. First, an additional failure can disable secondary paths that are used by victim FT-flows after an earlier failure. Second, RAFT's resource aggregation method makes the secondary path reservations just "wide enough" to accommodate the victim FT-flows of one component failure. It is not guaranteed that these reservations will be sufficient for all victim FT-flows after multiple failures (see Section 4.3). The RBone switches back to the normal state when there is again a primary and a secondary path with the required resource reservations for every FT-flow. The recovery procedure is presented in Section 4. The non-coincident RBone component failure model is valid if the time between successive RBone component failures is much larger than the RBone recovery delay.

RoS Classes. We consider the following two RoS classes that are based on the above fault model:

- LF: tolerance to an RBone link failure,
- LNF: tolerance to an RBone link or node failure.

The LNF class is more general than LF because it guarantees that the FT-flow will recover from any RBone component failure. However, the RBone links in general consist of several physical network components, and so they can normally have a larger failure rate than the RBone nodes. Consequently, the LF RoS class may be adequate for some FT-flows, especially if it is provided at a lower cost.

3 RAFT

For clarity of presentation we first focus on the LF RoS class. Then, RAFT is extended to support the LNF RoS class.

Notation. Let $\mathcal{N}$, $\mathcal{L}$, and $\mathcal{P}$ be the set of all RBone nodes, links, and ports, respectively. A link $l \in \mathcal{L}$ is comprised of two ports corresponding to the two directions of flow which take place in that link. Let P_f be the set of all ports in the primary path of FT-flow f, S_f the set of all ports in the secondary path of FT-flow f, and β_f be the bandwidth that must be reserved for FT-flow f in order to meet its QoS requirements. Also, let $\mathcal{F}_p^P$ be the set of FT-flows which use port p in their primary path, and Φ_p^P the total bandwidth reserved on port p for the flows $f \in \mathcal{F}_p^P$. Similarly, let $\mathcal{F}_p^S$ be the set of FT-flows which use port p in their secondary path, and Φ_p^S be the total bandwidth reserved on port p for the flows $f \in \mathcal{F}_p^S$. Φ_p^P is the *primary reservation* on port p and Φ_p^S is the *secondary reservation* on port p. Also, slightly overloading the terminology, we say that a link l is in P_f ($l \in P_f$) when one of its two ports is in P_f, and a node n is in P_f ($n \in P_f$) when one of its links is in P_f.

3.1 The SUM scheme

What is the appropriate secondary path reservation Φ_p^S on a port p? The straightforward approach is to reserve the same amount of bandwidth in both the primary and the secondary path ports. Formally,

$$\Phi_p^S = \sum_{f:\ p \in S_f} \beta_f \tag{3.1}$$

We call this approach *SUM scheme*. It is easy to see that the SUM scheme guarantees that the reservations along the secondary path of a FT-flow are adequate even if several components in its primary path fail simultaneously. However, the SUM scheme does not exploit the fact that component failures occur infrequently and so it results in excessive secondary path reservations.

3.2 The RAFT scheme (supporting only the LF RoS class)

In this subsection, assume that all FT-flows request the LF RoS class. The secondary path reservations are calculated as shown in Figure 2. $W_{p,l}$ is the set of FT-flows that use port p in their secondary paths and link l in their primary paths. $\phi_{p,l}$ is the sum of bandwidth requirements of the flows in $W_{p,l}$. The secondary path reservation Φ_p^S is the maximum $\phi_{p,l}$ over all links l that correspond to non-empty $W_{p,l}$ sets.

We now show that the RAFT scheme can support the LF RoS class. Consider any port $p \in \mathcal{P}$ and let Φ_p^S be as in Figure 2. Assume now that

For $p \in \mathcal{P}$ and $l \in \mathcal{L}$ **define**
$W_{p,l} = \{f \in \mathcal{F}_p^S : l \in P_f\}$
$\phi_{p,l} = \sum_{f \in W_{p,l}} \beta_f$
At each $p \in \mathcal{P}$
$\Phi_p^S = \max_{l \in \mathcal{L}: W_{p,l} \neq \emptyset} \{\phi_{p,l}\}$

Figure 2: The RAFT scheme (supporting only the LF RoS class).

a link $m \in \mathcal{L}$ fails. $W_{p,m}$ is the set of FT-flows that use link m in their primary paths and port p in their secondary paths. After the link failure, these victim FT-flows are switched to their secondary paths and they start using port p. If the link failure occurred while the RBone was in the normal state, these are the only FT-flows that use the secondary reservation on port p. The bandwidth that is required on p for the flows in $W_{p,m}$ is $\phi_{p,m} = \sum_{f \in W_{p,m}} \beta_f$. Since $\Phi_p^S \geq \phi_{p,m}$, and this is true for every port p, the RBone can recover all victim FT-flows.

We now show that the secondary reservation Φ_p^S is the *minimum* bandwidth requirement on port p for tolerating an RBone-link failure[4]. Consider a port $p \in \mathcal{P}$ and suppose that there exists a link $m \in \mathcal{L}$ such that $\Phi_p^S < \phi_{p,m}$. If link m fails, the bandwidth requirement on port p for secondary path reservations will be $\phi_{p,m}$, which is more than the secondary path reservation on p. In this case the RBone would fail to support some victim FT-flows on port p. Hence, the minimum value for Φ_p^S is as in Figure 2.

The RAFT algorithms. We now present the algorithms and data structures that RAFT requires. A *Link Fault Management Table* (Link-FMT) has to be stored in every RBone port. The Link-FMT has two columns. The first column contains the identity of the links $l \in \mathcal{L}$ that correspond to nonempty $W_{p,l}$ sets. The second column contains the corresponding value of $\phi_{p,l}$. An example of a Link-FMT for a port p is shown in

[4]Assuming that routing is performed independently of the resource reservations. An optimal value for Φ_p^S would be achieved if the two paths of a FT-flow were chosen so that the total resource reservations for the flow, taking into account the aggregation of secondary path reservations, are minimized. This is a fairly hard problem that we do not consider in this paper.

Update_Link-FMT:Initiate(β_f, P_f)
/* As executed on port p */
/* f is the initiating flow */
/* Port p is in f's secondary path */

foreach link $l \in P_f$
  if (l is not in Link-FMT)
    create the Link-FMT entry (l, β_f)
    $\Phi_p^S = \max\{\Phi_p^S, \beta_f\}$;
  else
    $\phi_{p,l} = \phi_{p,l} + \beta_f$;
    $\Phi_p^S = \max\{\Phi_p^S, \phi_{p,l}\}$;
}
Φ_p^S is the new secondary reservation on p;

Figure 3: Algorithm to update the Link-FMT of a port p when a FT-flow f initiates.

Update_Link-FMT:Terminate(β_f, P_f)
/* As executed on port p */
/* f is the terminating flow */
/* Port p is in f's secondary path */

foreach link $l \in P_f$
  $\phi_{p,l} = \phi_{p,l} - \beta_f$;
  if $\phi_{p,l} == 0$
    delete the l-entry of the Link-FMT;
}
$\Phi_p^S = \max_{l \in \mathcal{L}: W_{p,l} \neq \emptyset} \phi_{p,l}$;
Φ_p^S is the new secondary reservation on p;

Figure 4: Algorithm to update the Link-FMT of a port p when a FT-flow f terminates.

Link l	Bandwidth $\phi_{p,l}$
a	$\phi_{p,a} = \beta_2 + \beta_4 + \beta_8 + \beta_{10} = 4\alpha$
b	$\phi_{p,b} = \beta_1 + \beta_2 + \beta_4 + \beta_7 + \beta_{10} = 5\alpha$
c	$\phi_{p,c} = \beta_3 + \beta_4 + \beta_5 + \beta_6 = 4\alpha$
d	$\phi_{p,d} = \beta_5 + \beta_8 + \beta_9 = 3\alpha$
e	$\phi_{p,e} = \beta_1 + \beta_2 + \beta_8 + \beta_9 = 4\alpha$

Table 1: An example of a Link-FMT.

Table 1. We assume that ten FT-flows use port p in their secondary paths. The primary paths of these flows use one or more of the links a, b, c, d, and e. Suppose that all the FT-flows have the same bandwidth requirement α. With RAFT, the required secondary reservation on port p is $\Phi_p^S = 5\alpha$, while with SUM it would be $\Phi_p^S = 10\alpha$.

The Link-FMT must be updated each time a FT-flow initiates or terminates. Figures 3 and 4 show the algorithms executed on a port p when a FT-flow initiates or terminates, respectively. Consider a FT-flow f that uses port p in its secondary path ($f \in \mathcal{F}_p^S$). When f initiates, the the Link-FMT entries are updated to include the bandwidth requirement β_f of f. If necessary, the secondary reservation at p is increased to accommodate the new flow. The complexity of this algorithm is linear on the number of links in the primary path of f. Similarly, when f terminates, the Link-FMT entries are updated to remove the bandwidth requirement β_f of f. If possible, the secondary reservation at p can be reduced, that requires the maximum $\phi_{p,l}$ of all the Link-FMT entries of port p to be computed. The complexity of this algorithm is linear on the number of entries in the Link-FMT. Since the RBone is a virtual network, its links will normally be much fewer than the links in the actual underlying network. Consequently, both the memory space required for the Link-FMT and the complexity of the termination algorithm should not be prohibitive.

3.3 The RAFT scheme (supporting the LF and the LNF RoS classes)

RAFT can be easily extended to support the LNF RoS class as well. In this case, some FT-flows request tolerance to an RBone-link failure, while other FT-flows request tolerance to either an RBone-link or node failure. To support the

For $p \in \mathcal{P}$ and $n \in \mathcal{N}$ **define**
$W_{p,n} = \{f \in \mathcal{F}_p^S : n \in P_f\}$
$\phi_{p,n} = \sum_{f \in W_{p,n}} \beta_f$
At each $p \in \mathcal{P}$
$\Phi_{p,node}^S = \max_{n \in \mathcal{N}: W_{p,n} \neq \emptyset} \{\phi_{p,n}\}$

Figure 5: The RAFT scheme for calculating $\Phi_{p,node}^S$.

LNF class, two modifications are required. First, a table called *Node-FMT* is required in each port, in addition to the Link-FMT. A row of the Node-FMT stores the address of an RBone node n and the bandwidth $\phi_{p,n}$ that will be needed on p if node n fails. Second, the algorithms for the initiation and the termination of an LNF class FT-flow have to be extended so that both the Link-FMT and the Node-FMT entries are updated. Specifically, the Link-FMT updating is as shown in Figures 3 and 4. The Node-FMT updating is similar, except that the entries that correspond to each node (instead of link) of the FT-flow's primary path have to be updated. From the Link-FMT the secondary reservation $\Phi_{p,link}^S$ is derived, required to guarantee recovery from an RBone-link failure (see Figure 2). From the Node-FMT the secondary reservation $\Phi_{p,node}^S$ is derived, required to guarantee recovery from an RBone-node failure (see Figure 5). The final secondary reservation on port p is then given by

$$\Phi_p^S = \max\{\Phi_{p,link}^S, \Phi_{p,node}^S\} \qquad (3.2)$$

Φ_p^S is the minimum secondary reservation that is required for recovering all the LF- and LNF-class victim FT-flows after a link failure, or all the LNF-class victim FT-flows after a node failure.

4 Resource Reservation Protocol Support

RAFT requires support from the resource reservation protocol for the following three operations: (i) setup a secondary path for a FT-flow, (ii) initiate the recovery phase after a failure in the primary path of a FT-flow, and (iii) return RBone to the normal state (recovery completion). In this section, we show that the above operations can be realized with simple extensions on the emerging Resource Reservation Protocol (RSVP). Note that the focus in this section is on the sequence of message exchanges for implementing RAFT, and not on the latency of the recovery procedure, or on the increase of the signalling load. Further work is required to investigate the delays that RAFT may introduce in the flow setup, tear-down, or recovery after a fault.

RSVP Overview. RSVP can setup, manage, and tear-down resource reservations for unicast or multicast data flows in an integrated services network. A *Path* message is sent from the sender to the receiver(s) along a route selected by the routing protocol; this route is also used by the data flow. The receiver(s) send *Resv* messages upstream towards the sender. These messages create a *reservation state* at each network element along the flow's path. The *Resv* messages are routed hop-by-hop in the reverse direction from the *Path* messages, using the *path state* that the latter setup. Other RSVP messages are the *PathErr* and *ResvErr* error messages, and the *PathTear* and *ResvTear* tear-down messages. All the RSVP messages carry a variable number of variable length typed objects. The proposed extensions to RSVP are some additional objects in these messages. In the following we are limiting our discussion to unicast flows. Some comments regarding RAFT for multicast FT-flows are given in Section 7.

4.1 Setup a secondary path

To setup a FT-flow, the *Path* message sent in the secondary path has to include the primary path route. This is used by the RBone routing protocol for finding a node disjoint path, and by the RAFT algorithms at the initiation and termination of the FT-flow. The sequence of *Path* and *Resv* messages for setting up a primary and then a secondary path is shown in Figure 6. An object *PRIM_PATH* in the *Resv1* message is gradually filled with the primary path route, as the *Resv1* message follows the reverse primary path. The *PRIM_PATH* object is copied by the sender to the *Path2* message that is then routed along a node-disjoint path to the receiver. The path state stored in the nodes of the secondary path identifies it as a secondary path. Finally, the *Resv2* message is routed hop-by-hop along the reverse secondary path to setup and finalize the secondary reservations. If a secondary path cannot be found, a *PathErr* message is sent hop-by-hop from the intermediate node which encountered the routing failure to the sender. Since the receiver will not receive the *Path2* message, it knows that a secondary path has not been setup. If the resource requirements are not available along the secondary path, a *ResvErr* is sent hop-by-hop from the intermediate node which encountered the reservation failure to the receiver, releasing the reservations that have been made. In either case, an application, if it so chooses, can use the primary path and operate in a non-fault-tolerant mode, since the network is unable to setup a secondary path.

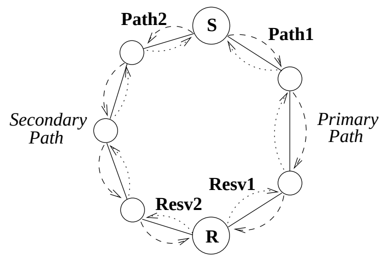


Figure 6: The primary and the secondary paths of a FT-flow, and the related *Path* and *Resv* messages.

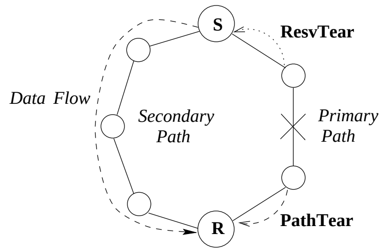


Figure 7: The recovery initiation procedure after a link failure.

4.2 Recovery Initiation

After a failure is detected, the RBone nodes that are adjacent to the failed component send a reservation tear-down message *ResvTear* to the sender and a path tear-down message *PathTear* to the receiver (see Figure 7). When the sender receives the *ResvTear* message it starts forwarding the data flow along the secondary path output interface. When the receiver gets the *PathTear* message it starts expecting the data flow from the secondary path input interface. An additional object *PATH_FAILED* has to be included in the *ResvTear* and *PathTear* messages to indicate that the primary path is being torn down due to a component failure. Since the reservations are still alive when the *PathTear* and *ResvTear* messages are sent over the primary path, it is expected that they will be delivered to their recipients reliably and in a timely manner.

4.3 Recovery Completion

Recall that after a component failure the RBone enters the recovery state. In general, the RBone cannot tolerate an additional failure in this state. This is illustrated in the following example. Suppose p is an RBone port that is used by some FT-flows in their secondary paths after the failure of a link or node x. Φ_p^S is the secondary reservation on p, and $\phi_{p,x}$ is the secondary path bandwidth that is actually used on p after the failure of x. According to RAFT, $\phi_{p,x} \leq \Phi_p^S$. The failure of another component y requires an extra bandwidth $\phi_{p,y}$ on p, and it may happen that $\phi_{p,x} + \phi_{p,y} > \Phi_p^S$.

The transition from the recovery to the normal state can be achieved by 'upgrading' every secondary path that is used by one or more victim FT-flows after a failure, to a primary path. This upgrading procedure requires the reservation of additional bandwidth on every port p in these secondary paths. First, the Link-FMT and the Node-FMT of port p are updated by deleting the entry that corresponds to the failed link or node. Then, the secondary reservation on port p is recomputed, and let $\hat{\Phi}_p^S$ be the resulting value. The additional bandwidth reservation that is required for the upgrading procedure on port p is $\hat{\Phi}_p^S - (\Phi_p^S - \phi_{p,x})$. The RBone returns to the normal state when all the ports that serve victim FT-flows have completed this upgrading procedure.

The additional bandwidth that is required for upgrading a secondary to a primary path may not be available in some ports. If the upgrading procedure is not successful within a certain time, new primary paths must be searched for the FT-flows that use these ports. A timer *TRY_UPGRADE* can be used for accomplishing this. After a port starts being used by a victim FT-flow, it sets this timer and it tries to reserve the required additional bandwidth. The admission of new flows can be temporarily stopped in this phase, so that the required bandwidth for the upgrading procedure to become available sooner. If the *TRY_UPGRADE* timer expires before the upgrading procedure succeeds, the port sends an object *UPGRADE_FAILED* with a *PathErr* message to the receiver and with a *ResvErr* to the sender. This object releases the additional reservations that have been made in the ports where the upgrading procedure succeeded, and it notifies the sender and the receiver that a new primary path has to be setup (see Figure 8).

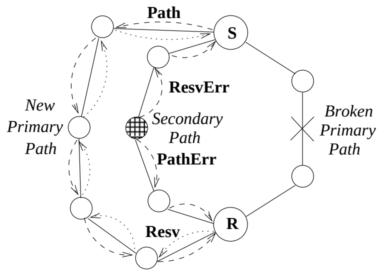


Figure 8: The upgrading procedure has failed on a port of the secondary path and a new primary path is setup.

5 Evaluation

In this section we evaluate the effectiveness of RAFT in reducing the required secondary reservations. The term 'flow' here refers to a guaranteed QoS flow that requires a fixed bandwidth reservation in every port of its path. These flows can be regular or FT-flows. Best-effort traffic or flows with looser QoS guarantees are not considered.

5.1 Performance Measures and Simulation Process

The evaluation is carried out with a simulator of the network resources and their reservation dynamics[5]. The three input parameters to the simulator are:

- A: The average arrival rate of flow setup re-

[5] Notice that the simulator does not capture packet level QoS degradations. Shortly after a fault there will be a backlog of dropped packets that, depending on the application, may have to be retransmitted. This transient degradation of the QoS is expected to be limited, however, because of the a priori reserved resources in the secondary path.

quests. The arrival distribution follows the Poisson model, and the duration distribution of the admitted flows follows the exponential model with an average duration of one simulation time unit. A indicates the load that is offered to the network.

- F: The fraction of requests for FT-flows. The rest are for regular flows.
- L: The fraction of FT-flow requests for the LF RoS class. The rest are for the LNF RoS class.

The performance metrics evaluated by the simulator are:

- R: The *acceptance ratio* of flow requests, measured as the fraction of flows that were admitted in the network. Sometimes we refer to the acceptance ratio for FT-flows only, or to the acceptance ratio for regular flows only.
- T, P: T is the *total network load*, i.e., the ratio of the reserved network bandwidth to the total network bandwidth. P is the *primary network load*, i.e., the network load that accounts for primary path reservations only.
- V: The *fault-tolerance overhead*. It is defined as the ratio of the total secondary reservations to the total primary reservations of the FT-flows, i.e.,

$$V = \frac{\sum_{p \in \mathcal{P}} \Phi_p^S}{\sum_{p \in \mathcal{P}} \Phi_p^P(FT)} \tag{5.1}$$

  where $\Phi_p^P(FT)$ is the primary reservations for FT-flows on port p. V shows, on the average, how much bandwidth is reserved in the secondary path of a FT-flow relative to the bandwidth that is reserved in its primary path. Secondary paths are often longer than primary paths and so V can be greater than one. V is an indication of the cost of providing fault tolerance. Suppose, for example, that for a certain topology, route selection, and secondary path reservation scheme, the fault-tolerance overhead is 1.3. If the cost of a flow is measured in terms of the reserved bandwidth, a FT-flow will cost on the average 130% more than the corresponding flow without fault-tolerance.

- G: The *aggregation gain*. It is defined as the relative reduction in the secondary reservations due to aggregation, i.e.,

$$G = \frac{\sum_{p \in \mathcal{P}} \Phi_p^{S,SUM} - \sum_{p \in \mathcal{P}} \Phi_p^S}{\sum_{p \in \mathcal{P}} \Phi_p^{S,SUM}} \tag{5.2}$$

  where $\Phi_p^{S,SUM}$ is the secondary reservation on port p using the SUM scheme, as defined in Equation 3.1, and Φ_p^S is the secondary reservation on port p using the RAFT scheme, as defined in Equation 3.2. G is always between 0 and 1 and is an indication of the benefit, in terms of network bandwidth, of aggregating secondary reservations. Suppose again that, for a certain topology and route selection, the aggregation gain is 0.5. A FT-flow will cost on the average twice as much with the SUM scheme, compared to the RAFT scheme.

The simulator implements the SUM as well as the RAFT scheme. Two node-disjoint routes are first selected for each pair of network nodes using the shortest path algorithm. These routes serve as the primary and the secondary paths for all the flows between the corresponding two nodes. The sender-destination pairs of the flow requests are uniformly distributed over the set of network nodes. The bandwidth requirement of each flow request is selected uniformly from the range 1.5% to 2.5% of the minimum link capacity of the network. A regular flow is accepted if its bandwidth requirement is available on all the ports of its primary path, while a FT-flow is accepted if the bandwidth requirement is available on both the primary and the secondary paths. Otherwise, the flow is rejected. The time required for the reservation protocol operations is not simulated, i.e., the initiation or termination of a flow is instantaneous. The maximum simulation time is set to 200 time units. The performance measures (R, T, P, V, G) are computed on 100 time instants that are uniformly distributed within the latter half of the simulation interval. The reported values at the end of the simulation are the average of these measurements. The above values are adequate for estimating the performance measures of interest with a 95% confidence at an accuracy of at least two significant digits.

As expected, the performance of RAFT depends significantly on the network load. The following results cover the range of primary load that

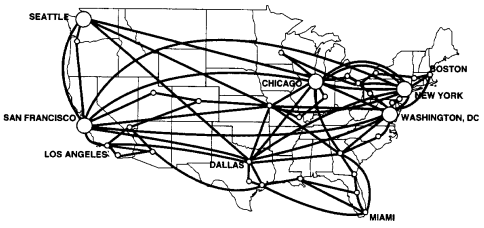


Figure 9: The UUNET backbone in USA.

results in a FT-flow acceptance ratio of more than 70%. The acceptance ratio of regular flows is always larger, while the acceptance ratio of LNF FT-flows is slightly less than the acceptance ratio of LF FT-flows. A lower acceptance ratio than 70% would probably indicate a badly dimensioned network.

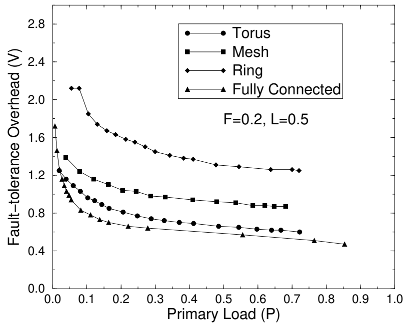
5.2 Simulated Networks

The performance of RAFT depends significantly on the network topology. We have experimented with several regular, random, and real network topologies. The topologies that are presented in this paper and some of their characteristics are summarized in Table 2. The *average degree*, is the average number of ports per node. It is a measure of the connectivity of the topology. The *path ratio* is the ratio of the average secondary path length to the average primary path length, over all the paths of the network. The path ratio depends both on the network topology and the selection of routes for the primary and secondary paths.

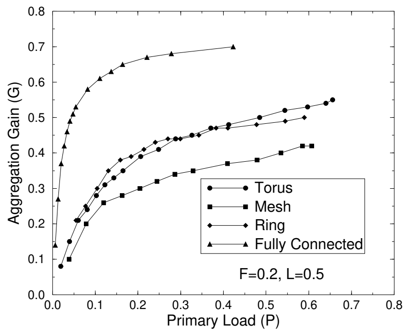
Network	Nodes	Links	Average Degree	Path Ratio
Ring	9	9	2.0	2.6
Mesh	9	12	2.7	1.5
Torus	9	18	4.0	1.3
Fully Connected	9	36	8.0	2.0
UUNET	33	68	4.1	1.5

Table 2: The simulated network topologies and their characteristics.

The *Ring*, *Mesh*, *Torus*, and the *Fully connected* topologies have 9 nodes. All the links of these networks have the same capacity. The simulation results for the regular topologies provide insight on the effect of the connectivity of the network topology on the performance of RAFT. Real networks do not usually have regular topologies, so most of the results in this study refer to the *UUNET* network, that is the high-speed backbone of UUNET in USA in the last quarter of 1997 (see Figure 9). All the links have the same capacity [6]. Other backbone topologies, such as the CERFnet or the ARPAnet, or random topologies that have some locality of interconnections, have given similar results with the UUNET network.



(a) Fault-tolerance overhead



(b) Aggregation gain

Figure 10: The performance of RAFT in regular network topologies.

5.3 The performance of RAFT in regular networks

The fault-tolerance overhead and the aggregation gain for the four regular topologies are shown

[6] In the topology shown there are three links between New York and Chicago, while we simulated only one link.

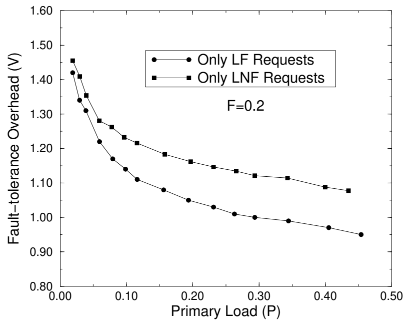
in Figure 10. In these simulations, the fraction of FT-flows F is 20%, while half of the FT-flows request the LF RoS class, and the rest request the LNF class (L=50%). The fault-tolerance overhead curves start from a large V value and drop significantly as the network load increases, because of the aggregation of secondary reservations. Another observation is that the initial value of V is close to the path ratio of the network. This is explained as follows: at very low network loads ($P < 5\%$) there are not many FT-flows in the network and the aggregation of secondary reservations is limited. In this operating region RAFT performs similarly with SUM. The fault-tolerance overhead V of the SUM scheme, however, is always close to the path ratio of the network, because the secondary reservations of a flow with the SUM scheme are proportional to the number of hops in the secondary path of the flow.

Note that the fault-tolerance overhead curves are ordered in the same way as the connectivity of the topologies, measured in terms of the average degree. As the connectivity of the topology increases, the chances of having disjoint primary paths sharing a common secondary path port are increased, and consequently, the aggregation of secondary reservations becomes more possible. We have encountered similar results with other topologies. The aggregation gain curves, on the contrary, are not ordered in the same way as the connectivity of the topology. The reason is that the aggregation gain also depends on the fault-tolerance overhead of the SUM scheme, that is the path ratio of the network. For example, the Ring has a larger aggregation gain than the Mesh, although it is less connected, because the reduction of the fault-tolerance overhead using RAFT is more significant in the Ring than in the Mesh.

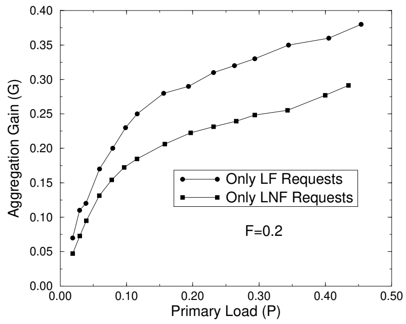
5.4 The performance of RAFT in the UUNET network

The evaluation in this subsection aims to show the effect of the following three parameters on the performance of RAFT in a real network topology: primary load (P), fraction of requests for FT-flows (F), and fraction of FT-flow requests for the LF RoS class.

The effect of P. The fault-tolerance overhead and the aggregation gain curves are shown in Figure 11 as a function of the primary load P. In these graphs, $F = 20\%$. Two curves are shown in each graph: one for LF RoS requests only ($L = 1$), and another for LNF RoS requests only ($L = 0$). When the network is moderately loaded ($P > 20\%$), the fault-tolerance overhead for the LNF case is approximately 10% larger compared to the LF case. This means that an LNF FT-flow in this network costs (in terms of bandwidth reservations) about 10% more than an LF FT-flow. LNF FT-flows are more 'expensive' for the following reason: since the RBone nodes are always fewer than the RBone links, there is a larger probability that the primary paths of two FT-flows share a node, rather than a link. Since two FT-flows that share a node in their primary paths cannot have common secondary reservations, the fault-tolerance overhead is not so much reduced with RAFT. Similar observations arise from the aggregation gain curves. Quantitatively speaking, the aggregation gain when the primary load

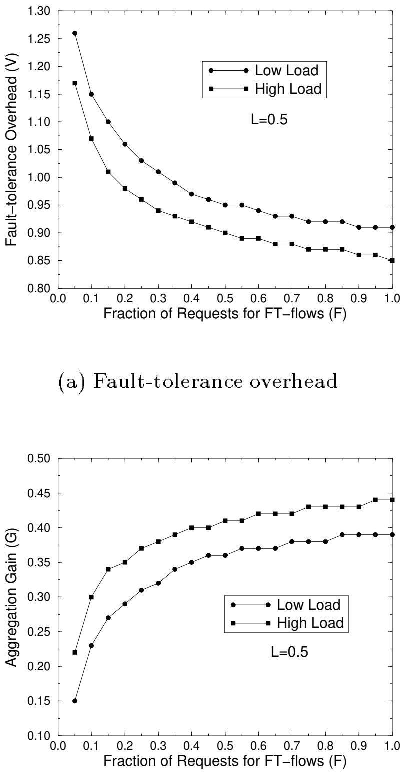


(a) Fault-tolerance overhead



(b) Aggregation gain

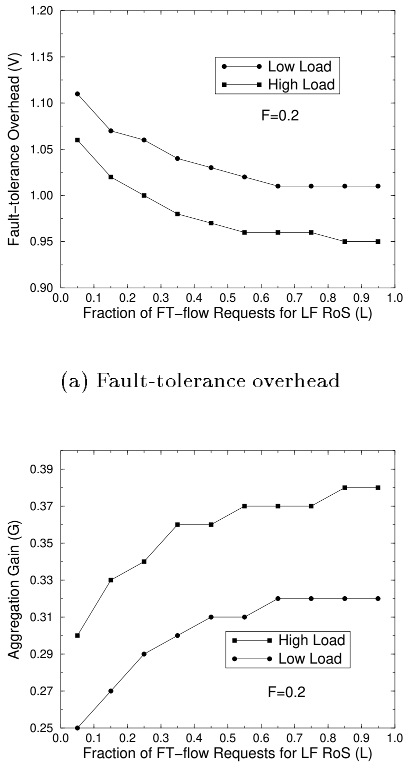
Figure 11: The performance of RAFT in UUNET as a function of the primary load P.



(a) Fault-tolerance overhead

(b) Aggregation gain

Figure 12: The performance of RAFT in UUNET as a function of the fraction of flows that request fault-tolerance (F).



(a) Fault-tolerance overhead

(b) Aggregation gain

Figure 13: The performance of RAFT in UUNET as a function of the fraction of FT-flows that request the LF RoS class (L).

is in the 20 to 40% range, is between 25 and 40%. The fault-tolerance overhead varies between 95 and 115%, meaning that a FT-flow costs 95 to 115% more than a regular flow. Since the path ratio of this network is 1.5, the cost of a FT-flow with the SUM scheme would be 150% more than the cost of a regular flow.

The effect of F. The fault-tolerance overhead and the aggregation gain curves are shown in Figure 12 as a function of the fraction F of flows that request fault-tolerance. In these graphs, $L = 50\%$. Two curves are shown in each graph: one for low network load conditions and one for high network load conditions. In the low load case, the acceptance ratio for all the different types of flows is larger than 97.5%, the primary load P is about 20%, and the total network load T varies between 25 and 40%. In the high load case, the acceptance ratio for all the different types of flows is larger than 70%, the primary load P varies between 30 and 40%, and the total network load T varies between 45 and 60%. Observe that as F increases the fault-tolerance overhead decreases and the aggregation gain increases. This is because as the number of FT-flows increases, there are more chances for resource aggregation. Also note that the aggregation gain increases rapidly with F when F is very low, while it almost saturates for F larger than 30%. This shows that even when the fraction of FT-flows is small (say 20%), the relative resource gain of using RAFT is comparable to the case where all the flows request fault-tolerance.

The effect of L. The fault-tolerance overhead and the aggregation gain curves are shown in Figure 13 as a function of the fraction L of FT-flows that request the LF RoS class. In these graphs,

$F = 20\%$. As before, two curves are shown in each graph: one for low network load conditions and one for high network load conditions. In the low load case, the acceptance ratio for all the different types of flows is larger than 96%, the primary load P is about 25%, and the total network load T is about 32%. In the high load case, the acceptance ratio for all the different types of flows is larger than 70%, the primary load P is about 45%, and the total network load T is about 52%. An interesting point in these graphs is that as the fraction L becomes larger than 50%, i.e., more than half of the FT-flows request the LF RoS class, the fault-tolerance overhead decreases at a slower rate. A similar observation can be made for the aggregation gain curves. This can be explained as follows: when most of the FT-flows belong to the LNF RoS class, the secondary reservations are mainly determined from the Node-FMTs. On the contrary, when most of the FT-flows belong in the LF RoS class, the secondary reservations are mainly determined from the Link-FMTs. The different slopes in the graphs of Figure 13 are caused because the rate with which $\Phi^S_{p,node}$ is decreased when L is increased is larger than the rate with which $\Phi^S_{p,link}$ is increased. The practical implication of this discussion is that it is beneficial for the network to have a pricing scheme for the LF and LNF RoS classes that adjusts L to around 50%. A higher L does not provide significant aggregation gain improvements, while it reduces the income from the higher-priced LNF FT-flows.

5.5 Bottleneck points

Large internetworks often have 'bottleneck points'. Consider for example an American network that is connected with two undersea links to a European network. Any FT-flow that passes from one network to the other will use both links, because of its two disjoint paths. These FT-flows have a common primary path link, and consequently, the aggregation gain for the undersea links will be zero. In general, although the average aggregation gain can be significant, there may be ports, normally located close to bottleneck points, where the aggregation gain will be very small or even zero.

6 Related Work

There is a large number of fault management mechanisms for telephone networks. The goal of fault recovery is to restore a large fraction of calls in usually less than two seconds. The restoration is performed at the lower levels of the transport hierarchy, such as at the SONET/SDH switching layer, and is based on self-healing rings, spare trunks, mesh restoration with crossconnects, and careful dimensioning of the network [15]. Similar methods have been proposed for ATM networks [1]. In most of the cases the restoration is performed at the *virtual path* (VP) level [16], and consequently it provides the same RoS to all connections of the VP. The idea that several VPs can share the resources of a backup VP, depending on their routing and assuming a certain fault model, has appeared also in [12]. This work, however, differs from RAFT because it is based on static resource allocation decisions in the network design phase (a similar analysis appears in [17]). Other reactive, both local and global fault management mechanisms have been studied in [4]. Several important issues, such as the locality of re-routing and the timing characteristics of the re-routing attempts have been addressed through extensive simulations. Due to the reactive nature of these schemes however, the probability of dropping a flow after a failure can be significant, and the recovery time can be several seconds, especially when the network is heavily loaded.

A global and proactive scheme based on *dispersity routing* and *forward error correction (FEC)* has been presented in [3]. A FT-flow is setup in multiple disjoint routes, called subchannels. Some subchannels transfer application packets, while the rest transfer FEC packets. If some subchannels (up to a certain number) fail, the flow survives because of the redundancy of the FEC. The main benefit of this approach is that the recovery time is practically zero, and that it does not require any significant network support. The bandwidth overhead for providing fault tolerance is reduced when more subchannels are used, given that the number of FEC subchannels remains constant. This scheme performs better than RAFT in terms of the cost of fault tolerance for non-coincident component failures when more than four or five subchannels are used. In many networks however, it may be hard to find so many disjoint routes between two hosts.

A global and proactive scheme that is similar to RAFT, called *Backup Channel Protocol* (BCP), has been recently suggested [9]. BCP supports different RoS classes, some of them for multiple failures, and it uses several backup channels (secondary paths) for every FT-flow. The method used for resource aggregation is based on a heuristic algorithm. A difference with RAFT is that BCP is a statistical scheme, in the sense that there is a certain probability of dropping some FT-flows after a failure, even if the failure conforms to the network fault model.

7 Conclusions

We have proposed a proactive and global fault recovery scheme for integrated services networks. RAFT can coexist with other fault recovery schemes that provide different reliability classes at a higher or lower cost. For example, applications with very strict RoS requirements can use a scheme such as [3], while applications that do not have special reliability requirements can use a dynamic re-routing scheme such as the one that is currently provided by RSVP [5]. We have shown that for the class of fault recovery schemes that use a primary and a secondary path for each FT-flow, RAFT decreases the secondary path reservations, reducing in this way the overhead for providing fault tolerance.

The presentation and evaluation of RAFT in this paper has been limited to unicast flows. Further work is required for the case of multicast FT-flows. A possible way for providing fault-tolerance to a multicast FT-flow is to have a secondary path for every branch of the multicast tree. The resource aggregation in this case may be larger because the secondary reservations of a subtree can be aggregated with the primary reservations of another subtree, within the same multicast session.

Acknowledgements
The authors are grateful to Kewal K. Saluja who co-authored an earlier version of this paper. We would also like to thank Anindo Banerjea, Larry Landweber, Brad Thayer, as well as the anonymous reviewers for their thoughtful comments.

References

[1] J. Anderson, B. T. Doshi, S. Dravida, and P. Harshavardhana, "Fast restoration of ATM networks," *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 1, pp. 128–138, January 1994.

[2] A. Banerjea, *Fault Management for Realtime Networks*, PhD thesis, Department of Computer Science, University of California, Berkeley, December 1994.

[3] A. Banerjea, "Simulation study of the capacity effects of dispersity routing for fault tolerant realtime channels," in *Proceedings SIGCOMM Symposium*, pp. 194–205, August 1996.

[4] A. Banerjea, C. J. Parris, and D. Ferrari, "Recovering guaranteed performance service connections from single and multiple faults," in *Proceedings IEEE Globecom*, pp. 162–168, November 1994.

[5] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin, *Resource ReSerVation Protocol (RSVP) - Version 1, Functional Specification*, September 1997. RFC 2205.

[6] D. D. Clark, "The Design Philoshopy of the DARPA Internet Protocols," in *Proceedings SIGCOMM Symposium*, pp. 106–114, 1988.

[7] D.Sidhu, S.Abdallah, and R.Nair, "Finding disjoint paths in networks," in *Proceedings SIGCOMM Symposium*, pp. 43–51, 1991.

[8] R. Guerin, S. Kamat, and S. Herzog, *QoS Path Management with RSVP*, March 1997. Internet Draft: draft-qos-path-mgmt-rsvp-00.txt (work in progress).

[9] S. Han and K. G. Shin, "Fast restoration of real-time communication service from component failures in multi-hop networks," in *Proceedings SIGCOMM Symposium*, September 1997.

[10] P. L. Higginson and M. C. Shand, "Development of router clusters to provide fast failover in IP networks," *Digital Technical Journal*, vol. 9, no. 3, pp. 32–41, 1997.

[11] K. Ishida, Y. Kakuda, and T. Kikuno, "A routing protocol for finding two node-disjoint paths in computer networks," in *Proceedings*

IEEE International Conference on Network Protocols, pp. 340–347, 1995.

[12] R. Kawamura, K. Sato, and I. Tokizawa, "Self-healing ATM networks based on virtual path concept," *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 1, pp. 120–127, January 1994.

[13] K. R. Krishnan, R. D. Doverspike, and C. D. Pack, "Improved survivability with multi-layer dynamic routing," *IEEE Communications Magazine*, pp. 62–68, July 1995.

[14] D. Kuhn, "Sources of failure in the public switched telephone network," *IEEE Computer*, pp. 31–36, April 1997.

[15] J. C. McDonald, "Public network integrity - avoiding a crisis in trust," *IEEE Journal on Selected Areas in Communications*, vol. 12, no. 1, pp. 5–12, January 1994.

[16] K. Murakami and H. S. Kim, "Virtual path routing for survivable ATM networks," *IEEE/ACM Transactions on Networking*, vol. 4, no. 1, pp. 22–39, February 1996.

[17] P.A.Veitch, D.G.Smith, and I.Hawker, "A comparison of pre-planned routing techniques for virtual path restoration," in *Performance Modelling and Evaluation of ATM Networks*, D. D. Kouvatsos, editor, volume 2, Chapman and Hall, 1996.

[18] D. K. Pradhan, *Fault-Tolerant Computer System Design*, Prentice Hall, 1996.

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

[20] S. Shenker, C. Partridge, and R. Guerin, *Specification of Guaranteed Quality of Service*, September 1997. RFC 2212.

[21] UUNET, "UUNET Technologies,", *http://www.uunet.net/*, December 1997.

[22] Q. Zheng and K. G. Shin, "Fault-tolerant real-time communication in distributed computing systems," in *Proceedings 22nd International Symposium on Fault-Tolerant Computing*, pp. 86–93, July 1992.