Fault-Tolerant Model for Deterministic IP Communication

Anupama Potluri, Sylvia Grace, Rajeev Kumar, A.G.K.Suman
Department of Computer and Information Sciences
University of Hyderabad.
apcs@uohyd.ernet.in

Abstract

Previous research in real-time communication over IP networks, does not give much attention to fault tolerance issues. We, in our work, have tried to add a fault tolerance capacity to schemes for real-time IP network communication methods. This is intended to improve reliability and ensure high availability. Here by a fault we mean the failure of a link or the failure of an intermediate node or port. Even in wired, highly reliable networks, there are chances of some failures, which cause packet loss and jitter (delay variation). Fault tolerance deals with such negative conditions and tries to make end-to-end availability close to 100%. We use the idea of bandwidth provisioning [2] and combine it with route pre-computation to guarantee determinism. In addition to that in our scheme we improve bandwidth utilization by fault provisioning on critical nodes and re-computation of paths along with very few redundant transmissions.

Introduction

Most approaches to providing Quality of Service (QoS) in IP networks are fundamentally based on the management of already-queued packets. During congestion such approaches experience degradation of QoS due to the variable queuing delays and possible overflows in buffer queues. To overcome this issue, there are primarily two different models proposed previously: a sequencing model [1] and the bandwidth provisioning model [2]. Here we briefly review these works before suggesting our approach to solve the problem.

Packet Sequencing Model

One approach to deal with this problem is to eliminate resource contention [1]. This is done by precisely scheduling, *a priori*, the arrival and service time intervals of packets at each network node. Thereby ensuring that two packets are never using the same switch resources at the same time. In this model, packet switching is unaffected by network load. For more details, see [1] and [3].

However, there are certain drawbacks with the approach proposed by Moore and Siller [1]. Firstly, this completely violates the philosophy of IP packet switching and reduces to a circuit-switching like behavior. Secondly, the paper does not talk about

failure conditions and the authors do not make it clear how determinism is maintained in cases of failures. One solution rather loosely suggested by the authors is to send packets redundantly on a few alternate paths to ensure availability [3]. Such a solution would be wasteful of bandwidth. On the other hand, they do not propose any mechanism to detect either link or remote node/port failures nor do they suggest any mechanism to handle such situations deterministically without redundant transmissions.

Bandwidth Provisioning Model

Another approach is to use bandwidth provisioning models as in integrated services for IP. The bandwidth needed at each node can be caluclated based on [2]. In this paper, the notion of effective bandwidth is defined for a source with a variable bit rate as "the minimum amount of bandwidth required to satisfy its QoS." Here a worst-case representation of traffic is described using a constraint function that has an envelope on the upper bounds of the arrival of packets. However, this paper does not talk about the process by which a path is chosen so that the QoS is satisfied. Their work is also silent about the issues related to path-failures.

Fault Tolerant Model

In this section, we propose and describe a model for a deterministic response from IP for different flows based on their QoS requirements. While we use the bandwidth provisioning model briefly discussed above, however, this bandwidth provisioning is done on a path that is chosen to satisfy the constraints of a deterministic response. We also discuss various failure conditions and propose methods to handle these failures. The proposed model is described with a network as shown in Figure 1.

Definitions and Assumptions

The network in Figure 1, has a source **S** which wants to communicate with a destination **D** and has certain QoS requirements where we choose delay as our network metric.

We assume that the network has at least two independent paths between source and destination nodes that satisfy the QoS requirements, without which there is no fault tolerance. We also assume that all the ***k*** alternate paths that are identified for the QoS requirements are all not congested at the same time.

A ***deterministic response*** is defined as ***d+*** where ***d*** is the delay while, is the maximum permissible jitter.

Primary Path is the path from source to destination that has the minimum delay that satisfies the QoS requirement of delay ***d***.

Transmission Path is defined as one path (from ***k***) that can satisfy the bandwidth provisioning requirements and hence is used for actual data plane operations.

A critical node is the one which has alternate paths emerging from it in the forward direction but nodes on the path downstream of it, do not.

Procedure
{Begin procedural description}

STEP 1
Precompute k paths that satisfy the QoS delay requirement through a topological sort of the network [5].

STEP 2
Provision the bandwidth on the best path using the effective bandwidth model [2]. A well known provisioning protocol such as RSVP [4] may be used.

STEP 3
To detect remote node/port failures, we use a new protocol, the fault-tolerant control protocol (FTCP), that sends 'hello' and 'ack' packets between successive nodes of the path once the flow is provisioned. Without this, any failure in the network will result in loss of deterministic response, since the existing control plane protocols take too long to detect a failure.

STEP 4
In the event of failure, the previous critical node on the path is informed so it can do transmission on an alternate path, as shown in Figures 2-4. The source is also informed through a control plane protocol that is piggybacked on the 'hello'/'ack' packets so that it can start the provisioning of the bandwidth on the alternate path. While the alternate path is being provisioned, the source also does redundant transmission on the alternate paths to ensure that the delay jitter remains within acceptable bounds. Once the alternate path is setup, the current transmission path is torn down, i.e., it deletes the provisioning of bandwidth on the nodes of the path. This is shown in Figure 5 below.

STEP 5
Once the source is finished with its transmission, it tears down the current transmission path.
{End}

Description

The source node **S** does a topological sort over the entire network to pre-compute the **k-best possible paths** in terms of delay jitter requirements identified earlier for the particular flow based on its QoS requirements.

The constraints that determine the selection of these k-paths are that :

1. If P_p is the primary path and its delay is represented as $D(P_p)$, then, the last path, P_k , with delay $D(P_k)$, is the path which is the upper bound on the delay, i.e., $D(P_k) = D(P_p) + d$.
2. There are at least two independent or node-disjoint paths that satisfy the above constraint.
3. The k-paths are as independent as possible with respect to the number of common nodes to ensure that there are no single points of failure.

After these k-paths are selected by step 1, bandwidth provisioning is done on the primary path. If the primary path is unable to satisfy the bandwidth requirements, say, because of congestion, then, the same procedure is repeated on the other independent path. This is done until a path satisfies the bandwidth requirements. Let us say that this path, which is the transmission path is P_t.

Once the transmission path is set up, the FTCP is started on this path to ensure the detection of remote node/port failure in such a way that the packet can be switched on to the alternate path while ensuring that the delay jitter is within the bounds as defined in the beginning of this section. The node n_{i-1} sends a hello packet to the node n_i which returns an ack in response. Let us say that the time between successive 'hello' and 'ack' packets is time m. If after n attempts, there is no 'ack' packet received from the remote node, it is assumed that the remote node/port is down. The values for m and n have to satisfy the following constraint:

$$D(P_t) + mn \leq D(P_i) + d$$

where, $D(P_i)$, is the delay of the alternate path, P_i , chosen to replace the current transmission path.

As soon as an error is detected, an error control packet is sent to the source node to start the initiation of bandwidth provisioning on an alternate path. The previous critical node is also informed of failure in the path. The critical node that has received the control packet then initiates transmission on other paths emanating from it. Since it is possible that there were packets in transit that got lost because of the failure of a node on the critical path, the critical node adds some buffering to re-transmit these buffered packets.

The source node, at the same time, does a redundant transmission of the packets that are yet to leave it, on all the alternate paths chosen in step 1. Once the new transmission path is set up, the redundant transmission is stopped. Also, the old transmission path is torn down.

Analyzing Failures

In this section, we analyze all possible failure conditions that might occur in the network. There are three possible failures that can occur in any network – *link failure, port* and *node failure*.

In case of a link failure, the previous node can easily detect the failure. On detecting the failure, it sends a control packet back to the previous critical node using the bandwidth provisioned for control messages. The critical node, in response to this control message, sends the rest of the packets arriving at it, including the buffered ones, redundantly in the alternate paths, thereby ensuring deterministic communication.

Remote node and port failures are detected using *hello and ack packet* mechanism. Every node sends a hello request to its immediate next node to test whether the latter has failed or not. If it has failed, the former does not receive any response in some time interval. If the former node is a critical node, it immediately initiates transmission on an alternate path from it, while it sends a control packet back to the source to notify it of the failure.

In all these cases of failure, the source node tries to setup an alternate primary path from the alternate ***k-1*** paths that were precomputed initially while also doing redundant transmission over these paths. The original primary path is torn down by deleting the provisioning of bandwidth on the nodes of this path.

Conclusion

This paper talks about the fault tolerance method in IP networks. The fault tolerance model deals with some negative conditions, which are frequently encountered, in real-time data communication. The method tries to provide end-to-end availability and determinism for real-time data very close to 100% while utilising the network resources well at the same time. The method uses very few redundant transmissions, which is an improvement over previous QoS methods where a lot of redundancy is used to guarantee availability. So, an efficient utilisation of network resources is also taken care of. Though the basic idea itself takes care of determinism and availability, we have also discussed some of the optimisation techniques, which can be used to increase the performance.

Since this paper talks about in-advance bandwidth provisioning, so the transmission will not be affected at all by any congestion on the primary path. While there is a very very small probability that all the ***k***-alternate paths will be congested at the same time. If that is the case, some jitter will be inevitable. There is also a

possibility of the next node going down right after responding to the hello request from the previous node. But probability of this is almost negligible.

Future Work

There are issues in the present scheme regarding the packets in transit and the duplicate/out-of-sequence packets arriving at the destination. The present scheme needs to be augmented with sequence numbering methods to be able to handle these issues to achieve complete fault-tolerance. We could also utilize the cycles in the network topology to find alternate paths from critical nodes which satisfy the QoS requirements so that packets in transit can be switched on alternate paths from the critical nodes while also informing the source node about failures. We also need to simulate the protocol after further development to analyse how it works.

References

1. "Availability of End-to-End Ideal QoS in IP Packet Networks" by Sean S.B. Moore and Curtis A. Siller Jr. *Computer Communications* (in Press)
2. "An Effective Bandwidth Model for Deterministic QoS Guarantees of VBR Traffic" by Hala M. Mokhtar, Rubem Pereira and Madjid Merabti. Proceedings of the Eighth *IEEE International Symposium on Computers and Communications* (ISCC'03).
3. Packet Sequencing: A Deterministic Protocol for QoS in IP Networks" by Sean S. B. Moore and Curtis A. Siller Jr. *IEEE Communication Magazine*. October 2003.
4. "Resource ReSerVation Protocol (RSVP) -- Version 1 Functional Specification" by Bob Braden, Lixia Zhang, Steve Berson, Shai Herzog and Sugih Jamin, *RFC 2205,* Network Working Group, IETF, September 1997.
5. "Introduction to Algorithms", T. Cormen, C. Leiserson, and R. Rivest*, MIT Press*, Cambridge MA, 1990.