Middleware for Timely and Reliable Communication over the Internet

Helton Miranda
helton.miranda@tecnico.ulisboa.pt

Instituto Superior Técnico
(Advisor: Professor Miguel Correia)

Abstract. This document addresses the problem of providing timeliness and reliability assurances for control traffic and file transfer over wide area networks. This is an important issue mainly for critical infrastructures and cloud infrastructures that rely on messages being delivered within the deadline so that they are able to function correctly. In this document it is proposed a solution to timely deliver messages by exploring the benefits of overlay networks and multihoming while also providing reliability assurances in the delivery. The reliability assurance is obtained by dependably adapting to the current environment and proactively handling network failures. Also the work describes the existent solutions that aim to provide these assurances identifying their strengths and weaknesses. Finally it is described how the proposed solution will be evaluated.

1 Introduction

Recently a lot of applications and information infrastructures are being deployed over the Internet. One of the reasons is that it reduces the cost of deployment because it is not needed to build a private network to connect them. One example is cloud infrastructures, as they are distributed over the world and therefore they resort on the Internet to interconnect them. Also critical infrastructures are being deployed over the Internet. They can be viewed as systems that have a great impact in our society. One example is the system responsible for the distribution of electricity.

As these systems are deployed over *wide area networks* (WANs), they need to be synchronized in order to function correctly. Therefore they often have timeliness requirements. This means that certain messages have to be delivered within a particular deadline. An example is data centers that need to exchange application control traffic to ensure a consistent view of their resources, e.g., Google's Megastore coordination service access [16]. Normally they rely on protocols that can assure these requirements most of time, e.g., TCP or UDP over IP. Although some of the systems are able to support violations in the requirements, there are those that have stringent requirements. Normally these requirements can not be assured over best effort approaches. These requirements are needed mostly by critical infrastructures as they require that the messages are delivered in time so

that they function correctly. Violations of time properties in these systems can cause big damages, e.g., power outages.

However meeting deadlines in WANs is a difficult task. Mainly because as these systems are distributed over WANs, they are susceptible to events which may compromise their operations. Two examples of these type of events are: failures in the network backbone and malicious faults. Network failures in IP backbone can lead to disruptions on the applications by making the network unstable over a period of time [38]. One example of malicious faults are *distributed denial of service* (DDoS) attacks which tend to generate a lot of traffic and cause congestions thus increasing the delay and packet loss rate of the messages exchanged between the infrastructures. There are several works in this field, as described in [49], but there is no final solution.

The objective of this work is to provide a middleware that timely delivers messages over WANs in a reliable way. The intention is not to offer the fastest way to deliver a message but to deliver it in time. We intend to provide these assurances both to control traffic and file transfer messages.

The majority of existent solutions do not provide these timeliness and reliability assurances. For example overlay solutions only aim to improve reliability by detecting and avoid faulty and congested paths. The ones that are able to provide these assurances rely on high quality links with high available bandwidth. Normally these type of networks are very expensive.

Therefore in this work we propose the development of a solution that is able to deliver the messages within their deadline in a reliable way. By exploring the benefits of overlay networks and multihoming it is possible to select one or more paths to deliver the message in time. The novelty is in the algorithms that will be developed to provide more reliability in the delivery. This is done by adapting the system to the environment that the underlying network provides and by proactively handling network failures. This imply that the network has to be periodically monitored in order to obtain sufficient data.

However this is not an easy task. One challenge that rises is how to efficiently select the most suitable paths to deliver the message knowing that WANs are unpredictable. Another challenge is how to relate the overlay paths with the physical paths. This is a big issue because for example two distinct overlay paths could share several physical links.

With this solution critical infrastructures and applications over WANs can see their timeliness and reliability requirements been assured with high probability. This solution will not require any changes in the underlying network and in the applications.

This report is organized as follows. Section 2 briefly describes the goal, the requirements and the expected results of this work. Section 3 presents an overview of solutions that are related to this work. Section 4 presents the proposal of the solution and Section 5 describes how the results are going to be evaluated. Finally Sections 6 and 7 present the scheduling of the future work and conclusions, respectively.

2 Objectives

This work addresses the problem of providing message latency and reliability assurances for control and file transfer traffic in WANs.

The goal of this work is to design and implement a practical solution to achieve timely and reliable communication with high probability in WANs, such as the Internet, at application level without requiring any changes in the network.

Therefore the proposed solution should have the following requirements:

- It should deliver the messages in a timely manner;
- It should provide that property both for small and large messages;
- It should provide reliability in the delivery assuring that the message will be delivered according to some probability;
- It should be cost sensitive, avoiding sending non-necessary packets like flooding.

The approach for this work is based on the assumption that at least one path is fast enough to deliver messages in time.

This work will produce the following results: i) specification of the algorithms for delivering the message within the deadline in a reliable way; ii) implementation of a prototype of this system; iii) extensive experimental evaluation with the prototype.

3 Related Work

The goal of the majority of the solutions in this field is to improve the system availability and performance in terms of delay or bandwidth consumption in WANs. Others approaches rely on dedicated networks to link their facilities in order to provide the same goals. This section addresses these approaches in order to understand what are the main challenges for this work.

Section 3.1 presents the existent solutions to obtain timeliness and reliability assurances. In the end it presents a summary of the solutions describing their advantages and disadvantages. Section 3.2 describes some algorithms that are used to estimate the following network parameters: latency; available bandwidth; packet loss; and path diversity. Section 3.3 addresses the algorithms for adapting timeout values and predicting the network channel state. These type of algorithms improve the reliability assurance. The last section presents a discussion about these works comparing them and showing which algorithms are the most suitable for this work.

3.1 Timely Communication in Wide Area Networks

Today there is a significant number of applications that require certain level of *quality of service* (QoS) in WANs. The QoS is specified in terms of throughput, bandwidth, packet loss ratio and latency. However todays WANs can not provide these QoS requirements because it is best effort. Therefore applications

rely on techniques to maintain the required degree of QoS. This section addresses these techniques describing their advantages and disadvantages taking into consideration the requirements for this work.

3.1.1 ATM One of the firsts approaches to deliver QoS in WANs was Asynchronous Transfer Mode, designated as ATM [18, 27]. ATM is a transmission, packet switching and multiplexing technique designed for a network that support both high-throughput data traffic and real-time, low-latency content such as voice and video. In [48] it was described that this technique can take advantage of bursty services while guaranteeing acceptable performance for continuous-bit-rate services. This is possible because ATM splits the usable bandwidth into small and fixed size units designated as cells and these cells are allocated to services on demand.

ATM defines four type of QoS classes and each one establishes the required performance and traffic parameters like *cell loss ratio* (CLR) or *cell delay variation* (CVD). The first class, default class or stringent class, requires small buffers with 100 cells. In this class the CLR has to be very low, almost to none. The second class, denominated as tolerant class, requires large buffers which is at least 1000 cells. The third class, bi-level, also requires small buffers, however the maximum CLR required is higher than in the stringent class. The last class, U class, has no requirements.

In order to optimize the selection of the performance and traffic parameters ATM provides multiple services categories which are CBR, rt-VBR and nrt-VBR, ABR and UBR. CBR, constant bit rate, is used in connections that need fixed rhythm. Rt-VBR, real time variable bit rate, is used for applications sensible to delays like video or voice. Non real time variable bit rate or nrt-VBR is used for applications which generate traffic bursts however are not sensible to delay. UBR, named as unspecified bit rate, is a best effort type of service which is similar to what the Internet provides. Finally ABR, designated as available bit rate, is used for applications that can adapt their traffic accordingly to the network conditions.

Although ATM can support different type of traffic classes thus providing the required QoS to applications it is a complex framework. One of the main reasons for not being massively used was the difficulty of deploying its interfaces to end hosts applications. Another issue was the high cost of the ATM network adapter in comparison with the Ethernet network adapter as referenced in [43]. Due to these issues ATM nowadays is used mainly in the core of the networks because of its unique advantages in providing the required QoS.

3.1.2 QoS in Distributed Multimedia Applications over the Internet Multimedia application is a combination of multiples forms of communications like text, sounds, videos or graphics. Most of these applications like *voice over IP* (VOIP), *television over IP* (IPTV) or *video on demand* (VOD) are sensitive to end-to-end latency because if it is higher than a certain value the users may notice it. Nevertheless, latency is not the only network parameter that affects

multimedia applications. This issue leads to a bad user experience with the application which is an important factor in this case. Therefore they usually require a minimum QoS in terms of packet loss, latency and others parameters from the underlying network. Recently most of this type of applications are being executed over the Internet in order to reduce their costs like IPTV for example. However the Internet provides only best effort services, so these applications have to employ techniques that help to assure the required QoS. Nevertheless these mechanisms do not succeed every time. This subsection is going to address some of these techniques that are used.

As the Internet itself does not support QoS requirements there were proposed some techniques to expand it so that it could support multimedia applications for example. [29] described the following techniques: admission control, policing, priority scheduling of packets and isolation. Admission control is a technique that compares the service QoS requirements and the available resources and based on that decides whether to accept or reject the service request. Another technique is policing which determines if a packet is obeying the *service level agreement* (SLA) negotiated between the origin and the network and if not it may drop that packet. Routers employs scheduling algorithms that share the available bandwidth in a fair manner or taking into consideration the priority of the variety of traffic classes present in the queues. Using a priority scheduling the queues are served in order of their priority. Isolation is the ability to isolate each flow from the others guaranteeing that they do not interfere with each other. Clark et al. described a schedule algorithm, WFQ [24], which was able to isolate the flows from each other providing a specific share of the available bandwidth to each flow. These techniques are the building blocks used in QoS approaches [29].

One of the solutions for supporting QoS requirements over the Internet was an *integrated services model* (IntServ) [22], with the goal of providing QoS to individual flows on the Internet. The model framework consists of four components: admission control, packet scheduling, classifier and reservation setup protocol. The idea was to have different service classes with different traffic characteristics that could match the application QoS requirements. IntServ establishes virtual paths for each flow and sets up the required resources in the path with the aid of RSVP [69], which is a reservation protocol that reserves a share of the resource for a service. According to [29] this solution was not scalable because the reservation mechanism was for each flow thus the router has to maintain the reservation for each flow that passes through it.

The *differentiated services model* (DiffServ) [19] was proposed to address the limitations of IntServ. Instead of providing QoS for each flow it provided for aggregates of flow by using 2 bits in the IP packet header to assign each flow to an aggregate behavior. Each aggregate received a different treatment in the network. DiffServ reserves resources for classes instead of flows thus removing the issue of scalability identified in the IntServ model. As the classification is based on the packet there is no need for a prior resource reservation thus eliminating the waiting time observed in the IntServ. According to [29] DiffServ does not provide

end-to-end QoS assurances to the traffic in the Internet. Also implementing DiffServ can be difficult because it requires to define a large set of values for non specified parameters.

Recently there are new techniques that are being applied to improve the network availability and performance. [44] presented a protocol to detect faults in a bidirectional path between two forwarding engines, part of the router responsible for forward incoming packets. Their goal was to provide a short duration detection of failures in the path between two forward engines by periodically exchanging *bidirectional forward detection* (BFD) packets and if one stops receiving the packets over a certain period it assumes that the other node has failed. It was made to be used over any protocol layer and over any media. Another technique is the statefull switchover [63]. It allows a redundant engine to take over if the primary fails thus improving fault resistance. By maintaining the user session information during a switchover the backup engine is able to continue to forward the traffic thus improving the network availability.

One of the main issues with the existent routing protocols is that they converge only in seconds which is a lot of time considering the number of packet loss in that link. IP fast reroute techniques [59] can solve this problem by providing convergence in less than one second. One approach is to have pre-computed backup paths so that when a failure is detected a backup path is used right away. The backup path has to be loop free by using a mechanism like split-horizon [67]. If is not possible to choose a loop free path it can be applied the NotVia mechanism which is more complex and requires new advertisement to be built. There are recent studies about using disjoint topologies technique [30, 15].

3.1.3 Overlay Networks Overlays Networks are virtual networks built on top of other networks, for example a virtual network built on top of the Internet, as shown in Figure 1. Each node in the overlay network is a computer and they cooperate for the same purpose using application layer protocols. They have the ability to increase network resilience, which is the the ability to maintain a certain level of QoS in the network even in the presence of faults or changing conditions. This is done by allowing nodes to send packets to indirect paths instead of the direct ones provided by the underlying network protocols. By increasing the resilience the network is able to maintain the required QoS thus it is capable of providing timeliness assurances. There are several approaches regarding overlay network routing protocols but in this work only those that select paths in order to provide reliability assurances will be addressed.

Several overlay network routing algorithms choose the paths that provide better end to end performance metrics. However there is no overlay solution that aims to provide timeliness guarantees by selecting a suitable path accordingly to the deadline. Most of them aim to increase the resilience and the performance by quickly detecting a link failure or by using packet redundancy to mask network faults for example.

In [7] Akella et al. evaluated an overlay routing technique where each node had a complete view of the network thus they were able to select the best overlay

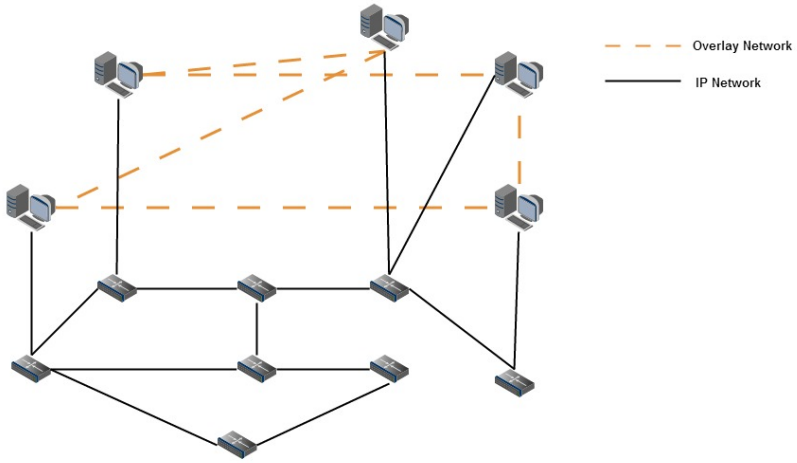

Fig. 1. Overlay Architecture example

paths in terms of performance and availability. They observed that this technique could improve *round-trip time* (RTT) between 20% to 70% compared to using direct BGP [56] routes over a single *internet service provider* (ISP). Therefore it could offer better performance than the direct path provided by the Internet.

The authors of [14] described two approaches that took advantage of existing path redundancy: mesh routing based on packet replication and reactive routing based on adaptive path selection. In the mesh routing approach a node sends a message through multiple paths at the same time. The goal is that the message should arrive at the destination through one of the paths even if there are link failures. In the reactive routing approach the nodes periodically probe their links to determine the required network metrics allowing to select paths that have the best probability to deliver the message.

One example of the mesh routing approach was presented in [61]. The approach consisted in creating a mesh based overlay network where every node was connected to n other nodes, parent nodes, thus forming a high redundant overlay network. By having this type of network, a node could use multiple paths simultaneously to multicast a stream of data. The novelty in this solution was the use of a protocol which could reliably reconstruct a sequenced packet stream by receiving packets from multiple senders. By using this type of protocol, the receiver could detect that a packet was missing in the same way as TCP, by verifying the *acknowledgment* (ACK) numbers. When a certain time passes and it still did not receive the missing packet, the node would request the packet through all the senders it received from. This type of failure recovery has good results when there is a burst of packets to transmit because it can detect that a loss occurred faster. [61] evaluated this solution setting up an experimental procedure and concluded that a node can experience a loss rate of 5% even if the parents are experiencing a loss rate of 45% individually. The main issue of

this solution is that the failure recovery is effective only when the detection time is very low, meaning it works fine for burst of packets but not for the control traffic type for example.

RON [12] is a reactive routing approach. It was described as an architecture that allowed nodes to detect and recover from path failures within seconds as compared to existent mechanisms in the underlying network that can recover only after a few minutes. In RON the nodes periodically probed their links and exchanged the measurements to other nodes in order to build a routing table which allowed to choose the best path available. By monitoring periodically each link a node could detect rapidly a link failure, consequently updating its routing table and then the node would disseminate this information to the others. RON used a exponential weighted moving average to calculate the expected latency in each link, similar to TCP. The loss probability is measured by calculating the two-way packet loss and then use the half for the one way packet loss probability because it is assumed that the paths are symmetrically. Assuming that paths are symmetric may not lead to the best results and it has been shown that Internet paths tend to be asymmetric [51, 34]. The authors of [12] also made some evaluations and they observed that RON took on average 18 seconds to detect and recover from faults. However this type of solution has a considerable issue which is that it is not scalable due to the aggressive probing and monitoring. Normally it works for a network comprised of 50 to 60 nodes. Although this solution presents a high resilience for the network it does not take into considerations the deadline of the messages thus it can not provide timeliness assurances.

There are other approaches using overlay networks to guarantee some level of QoS. The goal of OverQoS [62] is to provide QoS over overlay networks by offering statistical loss and bandwidth guarantees between the nodes. It assumes that the path is predetermined and fixed. The solution provides these guarantees by first determining the level of quality of service that can be provided through that path and then uses a hybrid solution for error control combining FEC and ARQ to actually provides the assurances. FEC, forward error correction, is a mechanism that exploits redundancy in the information transmitted. FEC is mostly used when retransmissions are costly or nearly impossible. ARQ, automatic repeat query, is the opposite of FEC, as the receiver uses *acknowledgments* (ACKs) to notify that it received the message. In ARQ if the sender does not receive the ACK over some time, it retransmits automatically the message. This solution was built to improve QoS for streaming applications that work on top of overlay networks. This means that it achieves good results only for stream traffic type. In [11] it was described an architecture to improve the performance of VOIP applications built on top of overlay networks, denominated as High Quality VoIP Streams. As this kind of application depends on the latency and jitter provided, it was propose to use UDP. Therefore this solution tends to suffer form packet losses and network failures, as UDP is best effort. This solution involved a protocol that chooses the best path taking into considerations both the latency and loss rate in each overlay link, designated as expected latency cost function. The authors of [11] also presented a protocol responsible for re-

covering packets only if there was a chance of delivering it in time due to the application specification. This protocol tried to recover packets locally, meaning that the recovery could be done at each intermediary node instead of being done at the end-hosts. This protocol is similar to the one used in [61], but the main difference is that when the receiver detects that a packet is missing it triggers the retransmission procedure immediately. So the only variable for the failure recover procedure is the detection time. Because this protocol is more effective when the detection time is shorter, the nodes aggregate multiple voice streams and then forward them in one connection. Therefore they are able to reduce the detection time. One problem with this solution is that each node needs a lot of computational power to calculate the expected latency cost function of all the links of a path.

3.1.4 Multihoming Multihoming can be defined as a computer connected to more than one external access link which belongs to different Internet providers or the same, as illustrated in Figure 2. The main advantage is obviously the increasing of the resilience in the network. Recently there have been studies about multihoming improving the network performance. In [6], the authors wanted to quantify the extent to which subscribers could leveraged connections to multiple network providers in order to improve the performance in a enterprise perspective and content provider perspective. The analysis was based on data collected from large data sets consisting of measurements taken at servers and measuring nodes of Akamai's *content distribution network* (CDN). They concluded that choosing the right upstream provider was crucial to improve the performance. Choosing the wrong provider could lead to a performance twice as bad as the optimal choice.

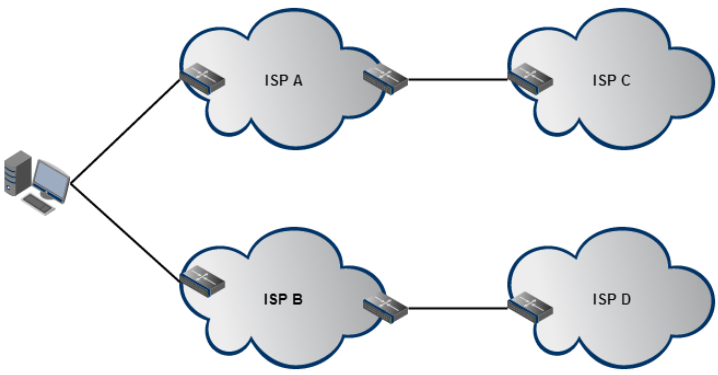

Fig. 2. Multihoming example

The best approach for choosing the most suitable provider is to probe each upstream link in order to know which can provide a better performance. Allowing the client to choose the best path to send messages and to which interface they should receive the messages was defined in [7] as *intelligent route control*. The authors of [8] evaluated several schemes that could extract the performance benefits of using multiple providers by offering routing control capabilities to clients. This work addressed how to monitor the links, how to choose the best path and how to direct the traffic over selected providers. One of the conclusion was that the route control schemes described could improve significantly the performance of client transfers up to 25%. Also they concluded that both active and passive measurement-based offered significant performance benefits in the range of 15% to 25% when compared to using the single best performing ISP provider.

However there are some considerations that have to be addressed in order to implement a multihoming solution. RFC 6418 [20] describes the main issues found when a node is attached to multiple provisioning domains or set of configuration information like gateways, DNS, prefixes and the correspondent interfaces. For each active interface the node can receive configuration information from different mechanisms such as DHCPv4, DHCPv6 or PPP. The problem is how to merge them or how to select the correct configuration when communicating. One example described in [20] was that when the configuration overlapped it could contribute to the node sending the data to the wrong destination. Other main issues are related to session management, routing, name resolution, address selection and security.

RFC 6824 extends TCP protocol to explore the multiples paths between peers, Multipath TCP [31]. The goal was to improve the throughput and resilience while also maintaining the same type of service that TCP offers. This solution operates at the transport layer and aims to be transparent to both higher layer and lower layer. To create multiple paths between two peers Multipath TCP establishes several TCP sessions, each associated to a different host address. Each TCP session is called a *subflow*. In the beginning the hosts exchange informations in order to know if they support Multipath TCP or not. This means this solution is backwards-compatible with TCP. This initial exchange also allows the hosts to establish additional subflows in the future. The send strategy used by the host is defined by a local policy. Multipath TCP also provides data retransmission by sending it through the original and a backup subflow.

Such improvements provided by multihoming intelligent route control can help achieve the goal of providing timeliness and reliability assurances. However an issue is that the client can not control the entire path to the destination, he controls only the access links. This problem can be solved with overlay networks by choosing indirect paths to reach some destination as stated in the previous subsection. In [7], the authors made comparisons between overlays, multihoming and k-overlays which is a combination of multihoming and overlay networks in terms of RTT, throughput and availability. Between overlays and multihom-

ing, they observed that in many cities as the number of providers increased, the RTT obtained by multihoming was shorter than overlays. The same applied to throughput as multihoming provided 2% to 12% better than overlays. When combining both overlay with multihoming, they observed that the benefits where marginally better than with just multihoming. However in terms of availability the overlay presented a 100% availability while mulithoming not. In conclusion overlays combined with multihoming can offer better performance and resilience in the network than using just multihoming or overlays. Also with this combination is possible to provide a quick failure recovery.

One example of this type of approach, combination of multihoming with overlay networks, is MONET [13]. The goal was to improve client availability to web sites by one more nine or better through the combination of multihoming with cooperative overlay network of peer proxies. This degree of magnitude is related to the percentage of available time that service providers offer to clients. The authors of [13] described that normally this value is around 95% to 99%. This approach created many potential paths between clients and web sites and relied on a scalable algorithm to select a good path. MONET consists of a set of web proxies deployed across the Internet which serve as conduits for client connections to web sites. When a client sends a request, the MONET proxy first checks if it has the object in its cache. If not it will check if it already has an open connection to the web site and if so it uses it to obtain the object. Otherwise it will use the waypoint selection algorithm to obtain an ordered list of the paths to the site and connects to the servers in the specified order. Each element of this list has a delay which specifies the time to wait until the correspondent path should be probed again. This algorithm seeks to order these paths accordingly to their degree of success. Occasionally MONET uses paths that are not the best to check if their quality has changed. In order to know when to do a retransmission in case of failure, MONET uses an algorithm to adapt the delay between requests depending on the network conditions. Although this algorithm does not have an high precision it is lightweight and can be implemented without increasing the computational requirements. The authors concluded that the waypoint selection algorithm performed almost as well as sending requests to all available interfaces, flooding. Although this system achieves most of the requirements stated for this work, its goal is to improve the resilience and performance meaning that it does not take in to considerations the message deadline in order to select a good path. Also it can not offer a good reliability in terms of delivering message in time because the algorithm used to adapt the timeout is not very accurate.

3.1.5 Data Center Networks Guaranteeing timeliness assumptions is an important objective in todays data centers. It is a big issue because they tend to host multiple applications and displaying latency to the end users promotes a bad user experience, which leads to loss of revenue. The problem is that the hosted applications tend to generate different type of flows at the same time. Therefore it becomes difficult to manage all those flows without compromising packet loss and consequently missing deadlines. One of the solution is to over-

provision the network bandwidth by providing far more bandwidth than the necessary. Another approach is to implement a deadline-aware network that allocates the required bandwidth for each flow, if available.

Wilson et al. proposed a solution to maximize the application throughput by maximizing the number of flows that are completed within the deadline [68]. Assuming that at the flow time initiation the correspondent size and deadline were available, the protocol tried to allocate in each router along the path the needed bandwidth in a greedy approach. This means that each router would try to satisfy as many flows as possible and the rest would be given a minimum bandwidth. Also in [68], they evaluated this solution using a small testbed. The conclusion was that their solution supported 28 concurrent senders while assuring that at least 99% of the flows completed in time in a moderate deadline, around 30 ms. Although this solution achieves good results in terms of providing timeliness assurances, it mandates that the network elements, like switches, have to be changed in order to support the allocation. Also it does not take the value of the deadlines in the bandwidth allocation procedure. In fact [64] described that this solution allocated the bandwidth for some of requests with longer deadlines before the ones that had smaller deadlines.

D2TCP is a distributed and reactive deadline-aware protocol which provides timeliness assurances and also alleviates the network congestion [64]. This protocol adds deadline awareness while also providing TCP properties like the reactiveness for allowing end-hosts to correct the over-subscription of the network. This solution also avoids network congestion issues by using an algorithm to prioritize the flows accordingly to their deadlines. To evaluate the protocol the authors did a small scale real test and simulations for large scale, and they concluded that the protocol reduced missed deadlines compared to D3 [68] by 50%. This protocol can co-exist with existent transport protocols like TCP.

Another type of approach to provide timeliness assurances was described in [41]. The authors adopted a *software defined network* (SDN) architecture for the interconnection of their data centers. A SDN allows the separation of the control plane from the data plane for the network management thus simplifying networking to enable rapid deployment of new network control services. The timeliness assurances can be provided by the centralized traffic engineering service because it is responsible for the allocation of the bandwidth among services basing on their priority. The authors of [41] observed that this solution enabled them to deploy substantial cost-effective WAN bandwidth, running many links at near 100% for extend periods thus providing high resilience in the network.

3.1.6 Discussion In conclusion, none of the approaches described in this section are able to guarantee the timely and reliability assurances in the delivery of messages over the Internet while meeting the proposed requirements.

Table 1 compares some of the solutions described in this section according to the requirements that were stated earlier because none of them were evaluated with the same benchmark. The solutions are compared in terms of: providing timeliness assurances, providing resilience over the network and providing a quick

failure recovery. The last term means that if they can detect quickly if a failure occurred and they are able to recover from it.

As the reader can see, the data centers approaches are able to provide timeliness assurances. However they rely on having a dedicated underlying network that is able to provide high available bandwidth which is not the case for this work. Normally building these type of network have a high cost which only big companies are able to afford it, for example Google which uses B4 [41] to connect their data centers.

ATM [18, 27] could provide timeliness assurances by allowing the allocation of fixed cells on demand. ATM defines four type of classes for QoS and each one establishes the required metrics. Also it provides service categories. These categories allow the users to select specific combinations of network parameters that are adequate to the intended service. However due to its complexity, ATM is not highly used today by end-host applications.

Multimedia applications have some of the requirements proposed for this work, mainly the timeliness assurances. To provide these assurances normally they rely on mechanisms like Bidirectional Forward Detection [44] and statefull switchover [63] to achieve it. But these mechanism are not reliable in the delivering within deadline because they are not able to quickly converge in case of a failure in the network. Also they do not adapt to the conditions that the underlying network is providing. DiffServ [19] is an interesting solution but it still has some issues to resolve as described in [29]. For example as DiffServ does not reserve the path therefore packets could be lost in case of network congestion. There are new works that intend to reduce the convergence time by having pre-computed backup paths such as disjoint topologies technique [30, 15].

Multihoming and overlay routing approaches can only provide resilience in the network and quick failure recovery by relying on mechanisms to detect a failure and select a backup path quickly. The issue with multihoming approaches is that although they add redundancy in the paths it is not possible to control the entire path to reach a destination. The client only chooses the access link and the rest is chosen by the by the network-level routing protocols, namely BGP. The only way to control the entire path is to modify BGP, more precisely changing the path selection algorithm to take into account other network parameters such as latency or packet loss rate. However this is a difficult task because the network providers normally have agreements between them about the traffic exchanged. Overlay routing can solve this issue by providing the client an indirect path to reach the destination, but it does not add the same degree of redundancy as multihoming.

Therefore none of the existent solution can provide all of the requirements proposed in this work, namely none of them is deadline aware for network control traffic and for file transfer. Nevertheless there are some good lessons can be taken from these works as most of the presented solutions improved the resilience or the performance over WANs.

Table 1. Comparison of the existent solutions according to the proposed requirements

Solution	Approach	Timeliness	Resilience	Quick Failure Recovery	Reference
ATM	QoS/ATM	yes	no	no	[48]
DiffServ	Resource Allocation	yes	no	no	[19]
MTCP	Multihoming	no	yes	yes	[31]
MONET	Multihoming	no	yes	yes	[13]
Mesh XML	Overlay	no	yes	yes	[61]
RON	Overlay	no	yes	yes	[12]
D2TCP	Data Center	yes	no	yes	[64]

3.2 Network Parameters Estimation

In oder to assure some level of QoS to the clients, most ISPs monitor the network by measuring the parameters like latency, bandwidth, jitter or packet loss in a consistent basis. By knowing what is the current condition of the network they apply some techniques, for example the ones discussed in Section 3.1.2 in order to provide the required QoS to the clients. Therefore it is important to estimate those parameters with high precision and low overhead. This section is going to address the network parameters and the tools to estimate them that are relevant for this work. Only tools that measure these metrics on an end-to-end perspective will be addressed.

3.2.1 Round-Trip Delay Network delay is an important indicator of the current state of the underlying network. It shows the time that a packet spends in the path until it arrives at the destination node. One type of network delay is the round-trip delay which can be defined as the time spent since the sender sent a request until receiving the correspondent response. According to [10], the round-trip delay is useful for applications that do not perform well if the delay is above some threshold. Also values above the minimum delay provide an indication of the congestion present in the path. There are two ways of measuring round-trip delay: active measurement and passive measurement. In active measurement a host is periodically probing the network while in passive measurement the host relies on data to be transmitted in order to obtain the round-trip delay. Therefore the passive approach is only effective if it is generated a substantial amount of traffic.

There are several approaches to estimate this metric. They differ in the types of packets that are used and the protocol layers where the measurements are made. One of the most popular is the *ping* tool which takes advantage of the request and reply mechanism defined in the ICMP [54]. It estimates the delay by measuring the time that the exchange of echo request and echo reply packets take. It is very popular because it is already included in the IP and is available for a long time in the majority of the operating system. However there is a major problem with this approach. According to [53] it was observed some unexpected

measurements using ping. One explanation was that the load-balancing mechanism implemented by routers were affecting the flow that the packets would take. This mechanism were implemented by routers that supported ECMP [37] by applying hash functions over the packet's header. Another approach is to use the TCP three-way handshake by calculating the time between the SYN and SYN/ACK packets.

One different approach to measure the round-trip delay between two hosts is measuring the RTT between their authoritative name servers. This approach was proposed in [32], with the goal of creating a tool that is scalable and does not need active cooperation from the two end-hosts. It assumes that most of the end-hosts in the Internet are close to their respective authoritative name servers. The authors gathered data from a list of servers previously studied and observed that 76% of web servers and 79% of Napster clients had at least one authoritative name server that was close to them.

Nevertheless all of these approaches have one thing in common which is that they use timestamps to mark when the packet was sent and when it was received. [10] discussed that one issue of measuring round-trip delay is that the clock should be synchronized since the request was sent until the response arrives. During this time if some event discontinued the clock it could lead to an inaccurate estimation. Also the authors discussed that using software clocks could cause some level of uncertainties because they only marked the times in the application level. Therefore they proposed that it should be used wire times, time in the network interface.

3.2.2 One-Way Delay In todays Internet the response path may be different from the request path, thus measuring the RTT does not provide the accuracy that some applications need as they are more concerned about the forward delay. Most applications calculate the *one-way delay* (OWD) by taking the half of the measured RTT. This is equivalent to assume that the delay is symmetric, meaning that both forward and backward delay have the same value. [51] wanted to quantify the delay asymmetry because of the path asymmetry existent on the Internet. After analyzing several measurements made by *owping* [3] which is a one-way delay estimation tool, and comparing them to the correspondent RTT, they observed that delay asymmetries definitely exist in the Internet. For example the RTT of a measurement was 150 ms and the respective forward delay was only 60 ms. Also they concluded that delay asymmetry implies that there is a path asymmetry but not vice-versa.

To the best of our knowledge all approaches for measuring OWD accurately require synchronized clocks. However synchronizing the clocks is not an easy task. RFC 2679 discusses some challenges to face related to clock synchronization like clock uncertainty, precision and skew, when calculating OWD [9].

There are two approaches to synchronize clocks in order to calculate the OWD. The first one involves that the hosts have to synchronize their clocks directly, accessing each other or using an external node. One example is the owping. Before sending the packet the host node marks its current time on the

packet. When the destiny receives the probe packet it subtracts its current time with the time on the packet thus obtaining the OWD. For each measurement the hosts exchange 10 probe packets. Owping uses NTP [47] to synchronize the hosts clocks. Before starting any measurement owping obtains the clock drift, the difference between host's and NTP's clock, from the NTP daemon on the system.

The other approach is by deploying proper infrastructures in the nodes which run a dedicated software. One example is Surveyour [42], which deploys measurement infrastructures at the sites and these infrastructures all have their clock synchronized. Each infrastructure performs its own measurements and stores in a central data base. Surveyour uses GPS for clock synchronization. GPS is an alternative for NTP but it requires a proper hardware.

3.2.3 Available Bandwidth Estimating accurately the available bandwidth of a path can benefit applications that require this parameter, like file transfer or interactive. It benefits because the applications can choose the paths that can support the size of their packets that they are about to send. As this work also pretends to assure timeliness in file transfers, measuring the available bandwidth in each path is important. [55] defined the available bandwidth as the unused or spare capacity of the link during a certain time period.

Iperf [1] is the easiest and most used tool. Its goal is to measure the achievable throughput using parallel TCP connections using TCP or UDP data streams. According to [39] the available bandwidth in a path could be estimated by measuring the achievable throughput.

Nevertheless there are other approaches. The authors of [55] described that there are two main techniques to estimate the available bandwidth: *self-loading periodic streams* (SLoPS) and *train of packet pairs* (TOPP).

In the SLoPS technique the sender sends k number of packets with the same size to a receiver in a rate r. It relies on the measurement of the one way delay of theses packets to verify if the delay is increasing or not, and if so it means that the rate is bigger than the available bandwidth in the path. The goal is to bring the rate close to the available bandwidth using an iterative algorithm. One example that uses this technique is Pathload [40].

In the TOPP technique, the sender sends each time a pair of packet probes to the receiver with an initial delay between them. In the receiver if the packets do not arrive with the same delay between them, TOPP assumes that the send rate is higher than the available bandwidth. The difference between these two techniques is that TOPP increases the rate linearly to reach the available bandwidth. One example that uses this technique is ABwE [50].

[60] made comparisons between these tools in terms of accuracy and overhead introduced. They observed that ABwE was the least accurate however it introduced the lowest overhead, almost none. Between Iperf and pathload, the authors observed that pathload was more accurate and introduced less overhead than Iperf. This analysis was made using data with realistic workload characteristics and also they used generated traffic from their lab, and they compared

the estimated values with the values obtained by MRTG which is a network monitoring tool.

3.2.4 Packet Loss Packet loss can impact application's performance whether they use any transport layer protocol by degrading their quality of service. For example when using UDP, excessive packet loss will impact negatively the user experience in applications like VOIP or video conferencing.

There are several tools that estimate the packet loss over a path. Ping and Iperf are two of the most used due to their simplicity. Ping uses ICMP packets as probes and it sends them at a fixed time interval. The loss is detected when the ICMP echo response does not arrive in a certain time period. Iperf uses the same methodology but the difference is that it uses UDP packets instead of ICMP packets.

One interesting finding is that packet loss also presents asymmetries over a certain path. In [21], it was observed that like the delay, the forward loss rate was not equal to the backward loss rate mainly due to the path asymmetry that the Internet provides. Sting [58] is a one-way packet loss rate estimation tool. It explores the TCP's error control mechanism to estimate the packet loss rate in both forward and reverse path. However Sting requires changes in the TCP so that it could provide accurate estimations.

Another interesting property of packet loss is that it presents some dependence pattern. In fact [21] concluded that the probability of a packet to be lost in a path was higher if the previous one was also lost.

3.2.5 Path Diversity The diversity of the multiple paths between two end-hosts impacts the performance and the availability in the network. For example [33] observed that although multihoming or overlay networks could provide significant gains in the performance and availability, their effectiveness depended of the natural diversity of the redundant paths between the end-hosts and in the ability to select the most uncorrelated paths. They concluded that to improve these metrics the overlay networks or multihoming should be topology-aware.

There are several solutions to estimate the correlation between the multiple paths available. The easiest approaches rely on using tools like *traceroute* [46] to determine the collection of routers that are present in a particular path. The same result can be accomplished using the using *lft* tool [2]. The difference between them is that lft determines the collection of *autonomous systems* (ASs) present in the path instead of the IP addresses. In this type of approach the correlation between the paths can be calculated by using the number of the ASs or IP sub-networks that are present in both paths. Most applications use these approaches because it is easy to deploy and to compute the correlation value.

However there are other approaches which use complex computations to determine the correlation between two paths also adding new metrics like the path history or link failures. One example is the solution presented in [70], which exploits the paths availability history in order to select topological disjoint paths. They proposed a new mechanism called *availability history window*

(AHW) which records the failures that occurred in the path over a time interval and from which the correlation of two paths can be determined. From the AHW they can determine that two path are highly correlated if they tend to fail at the same time interval. With this type of scheme they were able to detect and turn away from failure correlated paths.

In [25] the authors tried to find a backup route that minimizes the joint path failure probability between the backup and the primary path. They proposed this scheme because the overlay paths that share the same physical link would present a highly correlated failure probability. Therefore they are able to select a backup path which share the least failure probability in the overlay path but also sharing the least physical link in the network.

3.3 Dependability in Message Delivery

Achieving a defined level of dependability in the delivery of the messages over the Internet is a hard challenge due to the unpredictable environment which the Internet is. One way of providing dependability is to have a way to adapt the time bounds, timeout values that triggers the retransmission process, accordingly to the environment provided by the underlying network. Another solution tends to be by proactively monitoring the network channels in order to know in advance if some channels may fail or not. By resorting to these type of approaches a system can behave correctly even if the environment is uncertain.

3.3.1 Dependable Adaptation in Uncertain Environments In order to provide timeliness assurances systems have to adapt to the conditions that the underlying network is providing in order to preserve the required QoS. One way to provide dependability in the delivery is to timely detect failures, in this case timing failures, in order to trigger the retransmission mechanism while there is still time to deliver the message. This means that the failure detector has to be reliable in such way to detect the failures in time and also to be sure that the failure really occurred.

Most of the solutions presented in Section 3.1 have a mechanism to rapidly detect timing failures. For example in [11] and [61] it was detected that a failure occurred when the receiver did not receive the packet n while receiving packet $(n + 1)$. This means that if the receiver does not receive the following packet of the sequence it does not detect the failure, thus this mechanism is efficient only when there is a stream of packets going through the network.

[13] described two mechanisms for triggering the retransmission protocol, *rttvar* and *k-means clustering*. The approach of these mechanisms is to set up a delay threshold between the requests that should adapt to changes in the network conditions. The k-means clustering scheme collects connect time samples and groups them into k clusters. This computation can be done periodically or each time that a connection failed or succeeded. The rttvar scheme is inspired on the TCP retransmission timer computation described in [52]. Basically for each RTT sample taken, the delay threshold for the next requests will be updated

computing the average linear deviation of the delay until the path is probed again. It is simpler to compute the delay threshold using this scheme than using k-means clustering but the downside is that is less accurate than k-means clustering.

The main challenge to provide this type of dependability is how to adapt the delay threshold to changes in the network conditions without compromising the failure detector accuracy.

[65] presented the *Timely Computing Base* (TCB) that addresses this problem. In this work, the authors explored the concept of coverage which is the degree of correspondence between the system timeliness assumptions and what the environment can guarantee. Also they defined the coverage of a timely property as the probability of that property holding over an interval of reference. Basically the idea is to compute a time bound that assures that a timely property is guaranteed with some probability p. By using these concepts, the authors were able to compute the probability distribution function, *pdf*, which represents the distribution of the timing variable with a known error, *pdev*. They defined that an application whose bounds, delay threshold, can be changed dynamically belongs to the *time-elastic* class which is one of the classes that TCB can support. The authors of [23] applied these concepts to build a dependable QoS adaptation mechanism. The idea is to generate several pairs of (bound, coverage) to specify the QoS requirements, meaning that each time bound holds with certain coverage. By using these pairs, the system can select the new bound when the network conditions change in order to maintain the same coverage.

This means that *dependable adaptation* can be defined as ensuring that effective coverage of a system property will stay close to the assumed coverage and the difference is bounded.

[28] improved the approach in order to maintain correctness of the system properties after adaptation. With this improvement, the system was able to select optimistic time bounds when the environment was stable and select a conservative bound when the environment was unstable, but still providing dependable adaptation. In this solution after selecting a distribution that characterizes the current phase, they calculated the time bound values using six type of estimators: Conservative, Exponential, Shifted Exponential, Pareto, Weibull and Uniform. As their goal was to improve safe bounds, the one with the lowest value would be chosen.

In conclusion, the objective of dependable adaptation is to select the adequate time bound, so that the effective coverage of a property will be bounded to some specific value [28].

3.3.2 Network Channel State Prediction Systems that proactively handle failures can be more dependable because if they know a situation or event in advance they can trigger countermeasures in order to prevent the failure to occur or to repair it. [66] describes two types of prediction related to the time horizon, short-term prediction and long-term prediction of events. The short-term prediction of events is easier and more successful than the long-term prediction.

With short-term prediction systems can prevent or limit the damage that can be caused by a near future failure. [57] describes that proactive fault management consists of four steps:

- 1. The online failure prediction which collects a sample of measured data and outputs a decision;
- 2. The diagnosis phase, which is triggered by the online failure prediction to find the error;
- 3. The action scheduling which takes the decision and selects the actions that should be used for countermeasure;
- 4. The execution of those actions.

Thus in order for proactive management to be efficient, the online failure prediction model should be accurate.

Online failure prediction models are based on the runtime monitoring of the system parameters during a small interval to predict whether a failure might occur or not in the near future. Figure 3 illustrates how the online failure prediction works. At time t it is predicted that a failure might occur from $\Delta t1$, the lead time. The failure is predicted by analyzing the measured data during Δtd. The prediction is valid for the interval Δtp, designated as prediction period. The interval Δtw is the warning time which is the time that the system needs to react or prepare for the upcoming failure. This time needs to be smaller than the lead time in order to prevent or countermeasure correctly the future failure that was predicted.

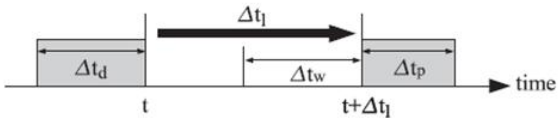

Fig. 3. Online Failure Prediction definition

There are several approaches and algorithms for online failure prediction. [45] described some metrics to qualify and compare the approaches. These metrics are mostly used by information retrieval algorithms but they can be applied to online failure prediction algorithms. To compute these metrics the authors of [57] based on the contingency table, which described the concepts of true positive, false positive, true negative and false negative. *True positive* (TP) indicates that the algorithm predicted a failure and it occurred within the prediction period. *False Positive* (FP) indicates that the algorithm predicted a failure but it did not occurred within the prediction period. *True negative* (TN) expresses that the algorithm did not predict a failure and no failure occurred within the prediction

period. *False negative* (FN) expresses that the algorithm failed to predict a true failure. The most relevant metrics are precision, recall, F-measure and accuracy.

Precision is the proportion of correctly predicted failures against all the failure predictions made:

$$precision = \frac{TP}{TP + FP} \tag{1}$$

Recall is the proportion of the correctly predicted failures against all the failures that occurred:

$$recall = \frac{TP}{TP + FN} \tag{2}$$

Precision and recall measure the ability of a system to correctly predict the failures. However by improving one the other decreases. Another metric is the F-measure which is the harmonic mean of precision and recall:

$$F\text{-}measure = \frac{2\ .\ precision\ .\ recall}{precision + recall} \tag{3}$$

Accuracy is the ratio of all correct predictions made to the number of all predictions that were made and not made:

$$accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{4}$$

The authors of [57] proposed to divide the approaches into five major branches. These branches differ in the type of input data that is measured. This section is going to address only two of those branches: symptoms and detected errors approaches. The other branches are not addressed because they can not be related to this work.

Symptom based prediction addresses the manifestations of faults that do not cause failures right away. Normally they are referred as side-effects of an error. The main challenge here is the symptoms are hard to detect. One example is a memory leak in the system because it does not cause any error until there is no more memory available. Nevertheless a memory leak can be detected by monitoring the memory usage and if it shows an increasing value it may be due to leaks. This concept can be applied to network by relating symptoms to the available bandwidth in the path, because if this parameter is constantly decreasing it may be a consequence of network congestion in the links of the path. The same thing can be applied to the path latency. There are four types of approaches in this branch which are: function approximation, classifiers, system models and time series analysis.

The goal of function approximation approaches is to select a function that closely matches a target value given an input data. The output of this function can be the probability of a failure occurrence or the future values of computing resources. One example of this approach is linear regression, which adapts the parameters of a linear function in order to the resulting curve of the function

matches the measured data. This approach can be used to predict for example the number of calls in a telecommunication operator or its availability. By applying this type of technique, the authors of [35] were able to model and forecast the call availability and failures of an industrial telecommunication system.

Classifiers can also be used to predict failures. Using classifiers, the system classifies the current state as a failure prone or not state. The classifier has to be trained first by collecting samples of failure prone state data and correct state data. Then it computes a graph which has a point that separates the failure prone states from the correct states. In runtime, given the current state the classifier checks whether this state is on the part of the graph which correspond to the failure prone state or to the correct state. In order to improve its accuracy, it is need a large amount of data for the training phase.

System models are similar to classifiers but the difference is that for the training phase they only need the data that corresponds to the correct state. During the runtime if the current state does not match the correct state, it is assumed that is a failure prone state. Although this method requires less data to be trained it has a higher false positive rate than classifiers. The authors of [36] proposed an intelligent system for failures detection that learns the normal behavior of the network and any deviation from this behavior are detected. The information is gathered and combined in the probabilistic framework of a Bayes network. The main advantage of this solution is that it could detect unknown faults. Also this type of approach is used in Network Intrusion Detectors in order to detect recent types of intrusions.

The last approach is based on time series analysis which treats a sequence of measured data as time series. This type of approach is mostly used to predict the future progression of the series in order to estimate the future value of the parameter. One example of its applicability is to compute the time for the resource exhaustion. Also it can be used to predict if the time series will violate a certain threshold.

Detected error reporting based prediction uses error reports to predict an upcoming failure. Basically it analyses all the errors that occurred during a time interval to predict a failure. The main difference in relation to symptoms is that the input is event-based instead of periodically monitoring the parameters. According to [57], this branch can be divided into 5 approaches. One of them is based on classifiers which have been discussed above. The other ones are: Rule-Based Systems, Co-occurrence, Pattern Recognition and Statistical Tests. Rule-Based Systems predicts an upcoming failure if at least one of the defined conditions are met. The conditions or rules are defined through error reports. Co-occurrence predicts that detected errors that occurred together are likely to occur again. Pattern recognition computes a ranking value to a sequence of errors reported based on the similarity to patterns that are prone failures and patterns that represent the correct state. Then through a classification based on the rankings, it is made a prediction. Finally Statistical Tests, like the name says, applies statistical tests to the reported errors to predict an upcoming failure.

These approaches rely on a training data set of samples in order to train the algorithms in order to detect failures in run time. [45] describes them as supervised machine learning techniques, because they involve a training phase.

This kind of approach can be applied in a network context to parameters like those discussion in Sections 3.2.1-3.2.4. They can be used to:

- Predict which network channel is in a dangerous state and may fail in the future;
- Predict the future values of the latency, available bandwidth and packet loss;
- Predict which routers will be used on a path in the future.

3.4 Summary

Realizing that there is no existent solution that is able to provide timeliness and reliability assurances while also providing dependable behavior, this work pretends to fulfill those requirements. The first step is to monitor the network parameters. Section 3.2.2 described the parameters that will be measured and the correspondent approaches. Choosing the best approach is important because it could impact negatively the network and in the worst case causing failures.

Table 2 compares some of the approaches used for estimating the delay in a path indicating: their target, type of network delay, the approach used, and presenting their main advantages and disadvantages. OWD offers more advantages mainly because it is possible to estimate the forward delay on a path which is very important for applications like VOIP. However is very challenging to estimate the OWD accurately because the clocks have to be synchronized. Between the two approaches described, estimating the OWD by accessing the hosts is less expensive than using a dedicated infrastructure but it needs that both hosts clocks are synchronized meaning that it is an intrusive approach. The most used approach is through using ICMP packets, as ping does, but as explained in Section 3.2.1 it is not very accurate. King [42] is an interesting approach because it does not need direct interaction between the hosts, but it relies that the DNS servers used are located near the hosts.

Table 2. Comparison between Delay estimation tools.

Solution	Target	Approach	Advantages	Disadvantages	Reference
Ping	RTT	Host interaction	Implicit in IP	Accuracy	[4]
King	RTT	DNS interaction	Accuracy	DNS located near the host	[32]
OwPing	OWD	Host interaction	Forward Delay	Clock Synchronization	[3]
Surveyour	OWD	Infrastructure	Forward Delay	Dedicated Infrastructure	[42]

Another network parameter described was the available bandwidth on path. There are three types of approach and in this case all of them were tested using

the same benchmark which helps the comparison. Table 3 compares the three approaches, presenting one example for each of them. Iperf is very accurate but it is sensible to packet loss meaning that it can loose the accuracy if it occurs packet loss during the estimation. Pathload [40] is an interesting solution but its accuracy depends on estimating the OWD accurately. ABwE [50] provides less overhead for the network but it is not accurate.

Table 3. Comparison between Available Bandwidth estimation tools.

Solution	Approach	Advantages	Disadvantages	Reference
Iperf	Parallel TCP	High Accuracy	Sensible to Packet Loss	[1]
Pathload	SLOPS	High Accuracy	Relies on OWD	[40]
ABwE	TPP	Less Overhead	Low Accuracy	[50]

Packet Loss estimation was also described. Table 4 compares the approaches previously described in terms of their advantages and disadvantages. The easiest way of estimating it accurately is using ICMP packets because they are already deployed in the IP. Therefore the majority of systems use it. However they can be easily filtered by firewalls. Iperf is easy to use but adds more overhead on the network. Nevertheless there are some more complex approaches that take in consideration the path asymmetry problem because like for delay, the degree of packet loss can be different for each way in a path. Sting is one example of this approach. It estimates both reverse and forward packet loss rate by exploring the TCP's error control mechanism. However the TCP has to be modified in order to obtain accurate results.

Table 4. Comparison between Packet Loss Rate estimation tools.

Solution	Approach	Advantages	Disadvantages	Reference
Ping	ICMP	Implicit in IP	Can be filtered	[4]
Iperf	UDP	Easy to use	High Overhead	[1]
Sting	One-Way Packet Loss	Forward Packet Loss Rate	TCP modification	[58]

Estimating the Path Diversity is important for systems that rely on retransmitting messages over a backup path, because they want to choose paths that are not correlated. Table 5 compares the approaches indicating their respective advantages and disadvantages. Traceroute is widely used because it is already implemented in IP. Lft is an alternative for traceroute. These two are different because they are IP aware and AS aware, meaning that it searches for IP addresses and ASs over that path respectively. Traceroute is less accurate than the other solution because a single router has multiple IP addresses. Lft provides low granularity because there are AS that have a large number of routers. Therefore

it may be possible to discover just one AS in path. There are some other complex approaches like the one presented in [70] which correlates paths using their history of link failures. These approaches are more accurate than just using IP addresses or AS providers discovered in the paths to correlate them. However they rely that the failures have to be captured accurately in order to provide accurate results.

Table 5. Comparison between Path Diversity estimation tools.

Solution	Approach	Advantages	Disadvantages	Reference
Traceroute	IP addresses	Implicit in IP	Low Accuracy	[46]
Lft	AS	Easy to use	Low Granularity	[2]
AHW	Path History	High Accuracy	Clock synchronization	[70]

Finally the last requirement for this work is that it should provide a correct behavior even if the environment is changing over the time. One way is to dependably adapt the timeouts for the timing failures accordingly to the network state. Most applications use a fixed value thus they do not take advantage of what the underlying network is providing or when the conditions degrade they loose the dependable behavior. The authors of [23] described a dependable model that adapts to the environment which relies on the concepts of coverage and built a probability density function that describes the probability distribution of the current state. By using this function it is possible to know which bound should be used to trigger the retransmission depending on the coverage needed. Another way is to proactively handle failures by predicting when they will happen in the near future. To make this possible, techniques like classifiers, system models, time series analysis, function approximation and so on are used. These techniques differ on the inputs used and on the type of prediction they provide. Classifiers and System Models are best used to predict the state of a target. They can be used to know the opinion of a sentence for example. Time series analysisand function approximation are best used to predict a value in the near future.The accuracy of these techniques relies on the amount of data that are used in the training phase. This phase is important because it allows the tunning of the algorithms thus improving the accuracy.

4 Solution

As discussed in Section 3.1 none of the current solutions is able to achieve all of the goals defined for this work. Mainly the majority of them do not provide timeliness assurances in the delivering of messages in the network. In order to provide the timeliness assurances this system is going to be deadline-aware, meaning it will select a suitable path that is able to deliver the message before the deadline while also selecting an adequate backup channel if occurs packet loss during the transmission in the primary path.

This work is going to extend an algorithm called *Just-In-Time Routing* (JITeR) [17], whose goal is to provide the timeliness and reliability assurances for short messages. The design rationale of JITeR, described in [26], was driven by the architecture of modern wide area Information Infrastructures, denominated as IIs, where these infrastructures are distributed over several facilities following a WAN-of-LANs model. That system is based on overlay networks combined with multihoming thus improving the network resilience that multihoming is able to provide while also having more possible paths to reach a destination. The nodes in the overlay network were connected to each other thus forming a one-hop based scheme which there is at most one intermediate hop to reach a certain destiny.

JITeR proactively monitored the network to assess the degree of independence between the channels and to measure the network latency of each one. These measurements were important because they were used to guide the routing decisions. Each node maintained information about the network latency of every overlay channel by measuring the RTT of their direct channels and then sharing them to the other nodes. It assumed that the latency was symmetrical thus establishing the *transmission time* (TXT) as the half of the RTT. Until the channel was not probed, JITeR used an algorithm to extrapolate the latency similarly to the one used in TCP, the exponential weighted moving average. The RTT was measured using UDP packets. The correlation degree between the channels was computed using the Sorensen-Dice coefficient. The routes were obtained using the traceroute tool.

When a node wanted to send a message the routing algorithm selected a base channel which allows the most possible retransmissions before the deadline was reached. This algorithm also provided some level of load balancing resulting that the best channels were available for the messages that had the shortest deadlines. Then it selected B backup channels based on their TXT and their degree of correlation with the base channel and between themselves. The value B was predefined and always constant.

4.1 Proposal

JITeR provides the timeliness and reliability assurances. Nevertheless it has room for improvements. We intend to extend JITeR in order to improve the reliability in the delivery and provide these assurances for file transfers.

We intend to build a module responsible for adapting the timeout bound accordingly to what the environment is giving. By exploring the concepts of the approach described in [23] we intend guarantee that the message will be delivered in time with according to a certain coverage even if the environment is unpredictable. With this module it is possible to provide a dependable adaption to the solution.

To predict whether a network channel will fail it is intended to use a combination of the classifier and time series analysis approaches. A classifier will be used to estimate whether the channel is in a failure-prone state or not. If the channel is not in a failure-prone state it will be used a time series technique

to predict the future values of the parameters in the correspondent channel. The decision of using time series instead of function approximation is that the analysis is made over samples that were measured over an interval thus treating the samples as a time series. It is intended to use a cross validation method in order to choose which type of classifier or time series is more suitable for this work. This method uses to data sets. The first data set which has known data is used to train the algorithms. After the training phase it is used the second data set with unknown data to test the algorithms. A specific challenge is to predict medium term periodic oscillations that sometimes affect the network, e.g., with a weekly periodicity.

Also we intend to design a more accurate path correlation algorithm to improve the selection of backup paths. To improve the accuracy we intend to combine two approaches that explore the path diversity. The first one is to list the network elements that are present in the paths and the other approach is to explore the path's failure history. Therefore our algorithm will gain the benefits of both approaches.

4.2 Network Monitoring

To support all these new components we intend to use some of the existent works, mainly to estimate the network parameters described in Section 3.2.2. The way JITeR estimates the OWD is very inaccurate because it was demonstrated that the Internet presents path asymmetry. Instead of computing the half of the RTT estimated we will use an approach similar to the one used in owping tool. The proposed solution intends to extend the measurement module to estimate as well the available bandwidth and the packet loss ratio over each path. To estimate the available bandwidth it will be used the pathload tool because although it adds more overhead than other tools it is highly accurate and for this work accuracy is one of the most important aspects. For packet loss ratio it will be used a solution that uses UDP packets.

To estimate the correlation between paths JITeR uses the traceroute tool. The main issue with this approach is that the same router can have different IP addresses thus affecting its accuracy in determining the paths correlation. The lft tool instead of searching for the IP addresses identifies the *autonomous systems*. Therefore by using lft it is possible to increase the accuracy. However using lft is not enough as it does not present high granularity. Therefore in order to improve the path correlation module accuracy it is going to be used the paths link failure history, similar to the work described in [70]. The challenge here is to find a way to combine both of these approaches in order to compute a final value that is able to describe the correlation between a pair of paths.

4.3 Architecture

The system architecture, illustrated by Figure 4, will be composed by:

– A measurement module which is responsible for estimating the delay, available bandwidth, packet loss ratio and correlation parameters (set of ASs and their link failure history);
– A bound estimation module which is responsible for computing the current bound according to the environment and the required coverage;
– A prediction module which is responsible for predicting whether a channel is in a prone-failure state or not and responsible for predicting the parameters future values;
– A channel correlation module that computes the correlation between channels based on their correlation parameters;
– A path selection module which base on the deadline select the channels that can deliver in time while also being the least correlated between them;
– A channel matrix which stores all the channels of the network with their respective parameters; a communication module which is responsible for the communication with the network.

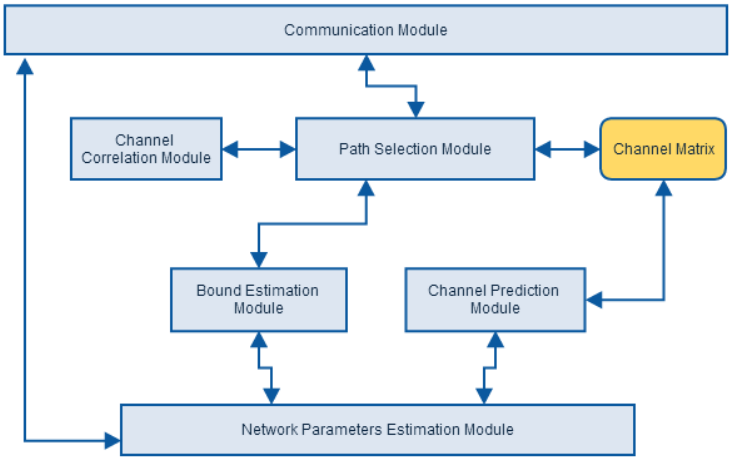

Fig. 4. System Components Architecture

5 Validation

The goal of this work is to provide timeliness and reliability assurances while providing a dependable behavior in the delivering of messages over the Internet. The validation of this solution is to build a prototype and process experimental evaluations. The evaluations are going to be done over PlanetLab [5] which is useful to test the prototype because it provides the possibility to test its

scalability and to run experiments with control over each node. The evaluation will be focused on the following aspects:

- Success Rate of messages delivered within the deadline, testing with variations on the deadline;
- Success Rate of file transfers completed within the deadline, testing with variations on the deadline;
- Comparison with other overlay strategies: *flooding scheme* and *primary and backup scheme* in terms of delivery within the deadline success rate and cost.

6 Scheduling of Future Work

Future work is schedule as follows:

- January 10 - March 29, 2014: Implementation of the proposal.
- March 30 - May 3, 2014: Experimental evaluation of the results.
- May 4 - May 25, 2014: Write a paper describing the work.
- May 26 - June 15, 2014: Finish the Writing of the dissertation.

7 Conclusion

This report analyzed several works that aim to provide timeliness and reliability assurances over WANs. It was observed that the majority of the solutions could not provide timeliness assurances. The ones that could relied on high quality links. This analysis also showed that some of the solutions were able to adapt to the conditions of the network but not reliable in a reliable way.

This analysis motivated us to propose a solution that was able to provide these requirements without any changes in the applications or the underlying network. In order to implement the proposal it is necessary to monitor the network to acquire data in order to help improve the reliability in the delivery. Therefore this work described several approaches to estimate the required network parameters and comparing them. With this comparison it was possible to select the most suitable approaches to be used in the solution. Finally the challenges to face in order to built such type of solution were identified.

References

[1] Iperf. Available at http://sourceforge.net/projects/iperf/.
[2] Layer four traceroute (lft). http://pwhois.org/lft/.
[3] Owping: One-way ping. http://e2epi.internet2.edu/owamp/.
[4] Ping. Available at http://ping.eu/ping/.
[5] Planetlab: An open platform for developing, deploying and accessing planetary scale services. https://www.planet-lab.org/.

[6] Aditya Akella, Bruce M. Maggs, Srinivasan Seshan, Anees Shaikh, and Ramesh K. Sitaraman. A measurement-based analysis of multihoming. In *Proceedings of the SIGCOMM*, pages 353–364, 2003.

[7] Aditya Akella, Jeffrey Pang, Bruce M. Maggs, Srinivasan Seshan, and Anees Shaikh. A comparison of overlay routing and multihoming route control. In *Proceedings of the SIGCOMM*, pages 93–106, 2004.

[8] Aditya Akella, Srinivasan Seshan, and Anees Shaikh. Multihoming Performance Benefits: An Experimental Evaluation of Practical Enterprise Strategies. In *Proceedings of the USENIX Annual Technical Conference, General Track*, pages 113–126, 2004.

[9] G. Almes, S. Kalidindi, and M. Zekauskas. A One-way Delay Metric for IPPM (RFC 2679). IETF Request For Comments, September 1999.

[10] G. Almes, T. Li, and S. Hares. A Round-trip Delay Metric for IPPM (RFC 2681). IETF Request For Comments, September 1999.

[11] Yair Amir, Claudiu Danilov, Stuart Goose, David Hedqvist, and Andreas Terzis. An Overlay Architecture for High-Quality VoIP Streams. *Proceedings of the IEEE Transactions on Multimedia*, 8(6):1250–1262, 2006.

[12] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Robert Morris. Resilient Overlay Networks. In *Proceedings of the SOSP*, pages 131–145, 2001.

[13] David G. Andersen, Hari Balakrishnan, M. Frans Kaashoek, and Rohit N. Rao. Improving Web Availability for Clients with MONET. In *Proceedings of the NSDI*, 2005.

[14] David G. Andersen, Alex C. Snoeren, and Hari Balakrishnan. Best-path vs. multi-path overlay routing. In *Proceedings of the Internet Measurement Conference*, pages 91–100, 2003.

[15] Alia Atlas, R Kebler, M Konstantynowicz, G Enyedi, A Császár, and Mike Shand. An Architecture for IP/LDP Fast-Reroute Using Maximally Redundant Trees. *draft-ietf-rtgwg-mrt-frr-architecture (work in progress)*, 2011.

[16] Jason Baker, Chris Bond, James C. Corbett, J. J. Furman, Andrey Khorlin, James Larson, Jean-Michel Leon, Yawei Li, Alexander Lloyd, and Vadim Yushprakh. Megastore: Providing Scalable, Highly Available Storage for Interactive Services. In *Proc. CIDR*, pages 223–234, 2011.

[17] Alysson Bessani, Nuno Neves, Paulo Veríssimo, Wagner Dantas, Alexandre Fonseca, Rui Silva, Pedro Luz, and Miguel Correia. Just in time routing.

[18] Uyless Black. ATM: FOUNDATION FOR BROADBAND NETWORKS. 1995.

[19] S. Blake, D. Black, M. Carlson, E. Davies, Z. Wang, and W. Weiss. An Architecture for Differentiated Services (RFC 2475). IETF Request For Comments, December 1998.

[20] M. Blanchet and P. Seite. Multiple Interfaces and Provisioning Domains Problem Statement (RFC 6418). IETF Request For Comments, November 2011.

[21] Michael S Borella, Debbie Swider, Suleyman Uludag, and Gregory B Brewster. Internet packet loss: Measurement and implications for end-to-end QoS. In *Architectural and OS Support for Multimedia Applications/Flexible*

Communication Systems/Wireless Networks and Mobile Computing., 1998 Proceedings of the 1998 ICPP Workshops on, pages 3–12, 1998.

[22] R. Branden and D. Clark. Integrated Services in the Internet Architechture: an Overview(RFC 1633). IETF Request For Comments, July 1994.

[23] Antonio Casimiro and Paulo Veríssimo. Using the Timely Computing Base for Dependable QoS Adaptation. In *Proceedings of the SRDS*, pages 208–217, 2001.

[24] David D. Clark, Scott Shenker, and Lixia Zhang. Supporting Real-time Applications in an Integrated Services Packet network: Architecture and Mechanism. In *Proceedings of the SIGCOMM*, pages 14–26, 1992.

[25] Weidong Cui, Ion Stoica, and Randy H. Katz. Backup Path Allocation Based on a Correlated Link Failure Probability Model in Overlay Networks. In *Proceedings of the ICNP*, pages 236–245, 2002.

[26] Wagner Saback Dantas, Alysson Neves Bessani, and Miguel Correia. Not quickly, just in time: Improving the timeliness and reliability of control traffic in utility networks. *networks*, 9:12, 2009.

[27] M. de Prycker. *Asynchronous Transfer Mode: Solution for Broadband ISDN*. Prentice Hall, 3rd edition, 1995.

[28] Monica Dixit, Antonio Casimiro, Paolo Lollini, Andrea Bondavalli, and Paulo Veríssimo. Adaptare: Supporting automatic and dependable adaptation in dynamic environments. *Proceedings of the TAAS*, 7(2):18, 2012.

[29] Mohamed A. El-Gendy, Abhijit Bose, and Kang G. Shin. Evolution of the Internet QoS and support for soft real-time applications. *Proceedings of the IEEE*, 91(7):1086–1104, 2003.

[30] Gábor Enyedi, Péter Szilágyi, Gábor Rétvári, and András Császár. IP Fast Reroute: Lightweight not-via without additional addresses. In *Proceedings of the INFOCOM 2009, IEEE*, pages 2771–2775, 2009.

[31] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. TCP Extensions for Multipath Operation with Multiple Addresses (RFC 6824). IETF Request For Comments, January 2013.

[32] P. Krishna Gummadi, Stefan Saroiu, and Steven D. Gribble. King: estimating latency between arbitrary internet end hosts. In *Proceedings of the Internet Measurement Workshop*, pages 5–18, 2002.

[33] Junghee Han and Farnam Jahanian. Impact of Path Diversity on Multihomed and Overlay Networks. In *Proceedings of the DSN*, pages 29–38, 2004.

[34] Yihua He, Michalis Faloutsos, Srikanth V. Krishnamurthy, and Bradley Huffaker. On routing asymmetry in the Internet. In *Proceedings of the GLOBECOM*, page 6, 2005.

[35] Guenther Hoffmann and Miroslaw Malek. Call availability prediction in a telecommunication system: A data driven empirical approach. In *Reliable Distributed Systems, 2006. SRDS'06. 25th IEEE Symposium on*, pages 83–95, 2006.

[36] Cynthia S. Hood and Chuanyi Ji. Proactive Network Fault Detection. In *Proceedings of the INFOCOM*, pages 1147–1155, 1997.

[37] C. Hopps. Analysis of an Equal-Cost Multi-Path Algorithm (RFC 2992). IETF Request For Comments, November 2000.

[38] G. IANNACCONE, C. CHUAH, R. MORTIER, S. BHATTACHARYYA, and C. DIOT. Analysis of link failures in an IP backbone. In *Proceedings of the IMW*, 2002.

[39] Manish Jain and Constantinos Dovrolis. End-to-end available bandwidth: measurement methodology, dynamics, and relation with TCP throughput. In *Proceedings of the SIGCOMM*, pages 295–308, 2002.

[40] Manish Jain and Constantinos Dovrolis. Pathload: A measurement tool for end-to-end available bandwidth. In *Proceedings of Passive and Active Measurements (PAM) Workshop*, 2002.

[41] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jon Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. B4: experience with a globally-deployed software defined WAN. In *Proceedings of the SIGCOMM*, pages 3–14, 2013.

[42] Sunil Kalidindi and Matthew J Zekauskas. Surveyor: An infrastructure for internet performance measurements. In *Proceedings of INET'99*, 1999.

[43] Charles Kalmanek. A retrospective view of ATM. *ACM SIGCOMM Computer Communication Review*, 32(5):13–19, 2002.

[44] D. Katz and D. Ward. Bidirectional Forwarding Detection(RFC 5880). IETF Request For Comments, June 2010.

[45] Bing Liu. *Web data mining: exploring hyperlinks, contents, and usage data*. Springer, 2007.

[46] G. Malkin. Traceroute Using an IP Option (RFC 1393). IETF Request For Comments, January 1993.

[47] David L. Mills. Internet time synchronization: the network time protocol. *Proceedings of the IEEE Transactions on Communications*, 39(10):1482–1493, 1991.

[48] Steven E Minzer. Broadband ISDN and asynchronous transfer mode (ATM). *Communications Magazine, IEEE*, 27(9):17–24, 1989.

[49] Jelena Mirkovic and Peter Reiher. A Taxonomy of DDoS Attack and DDoS Defense Mechanisms. *ACM SIGCOMM Computer Communication Review*, 34(2):39–53, 2004.

[50] Jiri Navratil and R Les Cottrell. ABwE: A practical approach to available bandwidth estimation. In *Proceedings of the 4th Passive and Active Measurement Workshop (PAM)*, 2003.

[51] Abhinav Pathak, Himabindu Pucha, Ying Zhang, Y. Charlie Hu, and Zhuoqing Morley Mao. A Measurement Study of Internet Delay Asymmetry. In *Proceedings of the PAM*, pages 182–191, 2008.

[52] V. Paxson. Computing TCP's Retransmission Timer (RFC 6298). IETF Request For Comments, June 2011.

[53] C. Pelsser, L. Cittadini, S. Vissicchio, and R Bush. On the suitability of ping to measure latency, May 2013. https://ripe66.ripe.net/presentations/128-130513.tokyo-ping.pdf.

[54] J. Postel. Intenet Control Message Protocol (RFC 792). IETF Request For Comments, September 1981.

[55] Ravi Prasad, Constantinos Dovrolis, Margaret Murray, and KC Claffy. Bandwidth estimation: metrics, measurement techniques, and tools. *Network, IEEE*, 17(6):27–35, 2003.

[56] Y. Rekhter, T. Li, and S. Hares. A Border Gateway Protocol 4 (BGP-4) (RFC 4271). IETF Request For Comments, November 2006.

[57] Felix Salfner, Maren Lenk, and Miroslaw Malek. A survey of online failure prediction methods. *Proceedings of the ACM Computing Surveys*, 42(3):10, 2010.

[58] Stefan Savage. Sting: A TCP-based Network Measurement Tool. In *Proceedings of the USENIX Symposium on Internet Technologies and Systems*, 1999.

[59] M. Shand and S. Bryant. IP Fast Reroute Framework (RFC 5714). IETF Request For Comments, January 2010.

[60] Alok Shriram, Margaret Murray, Young Hyun, Nevil Brownlee, Andre Broido, Marina Fomenkov, and Kimberly C. Claffy. Comparison of Public End-to-End Bandwidth Estimation Tools on High-Speed Links. In *Proceedings of PAM*, pages 306–320, 2005.

[61] Alex C. Snoeren, Kenneth Conley, and David K. Gifford. Mesh Based Content Routing using XML. In *Proceedings of the SOSP*, pages 160–173, 2001.

[62] Lakshminarayanan Subramanian, Ion Stoica, Hari Balakrishnan, and Randy H. Katz. OverQoS: An Overlay Based Architecture for Enhancing Internet QoS. In *Proceedings of the NSDI*, pages 71–84, 2004.

[63] Cisco Systems. *Cisco IOS Software Configuration Guide*. 2011. `http://www.cisco.com/en/US/docs/switches/lan/catalyst6500/ios/12.2SY/configuration/guide/stateful\_switchover.pdf`.

[64] Balajee Vamanan, Jahangir Hasan, and T. N. Vijaykumar. Deadline-aware datacenter TCP (D2TCP). In *Proceedings of the SIGCOMM*, pages 115–126, 2012.

[65] Paulo Veríssimo and Antonio Casimiro. The Timely Computing Base Model and Architecture. *Proceedings of the IEEE Trans. Computers*, 51(8):916–930, 2002.

[66] Ricardo Vilalta, Chidanand V Apte, Joseph L. Hellerstein, Sheng Ma, and Sholom M. Weiss. Predictive algorithms in the management of computer systems. *Proceedings of the IBM Systems Journal*, 41(3):461–474, 2002.

[67] D. Waitzman, C. Partridge, and S. Deering. Distance Vector Multicast Routing Protocol (RFC 1075). IETF Request For Comments, November 1988.

[68] Christo Wilson, Hitesh Ballani, Thomas Karagiannis, and Antony I. T. Rowstron. Better never than late: meeting deadlines in datacenter networks. In *Proceedings of the SIGCOMM*, pages 50–61, 2011.

[69] Lixia Zhang, Steve Deering, Deborah Estrin, Scott Shenker, and Daniel Zappala. RSVP: A new resource reservation protocol. *Network, IEEE*, 7(5):8–18, 1993.

[70] Xin Zhang and Adrian Perrig. Correlation-resilient path selection in multi-path routing. In *Proceedings of the Global Telecommunications Conference 2010*, pages 1–6, 2010.