End-To-End Routing for User Driven Network Recovery

Simon Delamare
Telecom ParisTech
46 rue Barrault, 75634 Paris Cedex, France.
Email: simon.delamare@telecom-paristech.fr

Michel Riguidel
Telecom ParisTech
46 rue Barrault, 75634 Paris Cedex, France.
Email: michel.riguidel@telecom-paristech.fr

***Abstract*—Dependable communications are needed by some users. When failures disrupt their communications, recovery mechanisms are needed to bypass them and preserve communications delivery. This article studies a recovery mechanism deployed by users themselves, which suits to their needs. We propose a system based on end-to-end routing in an overlay network to recover users' communications. We present this system, and measure its performances thanks to a software implementation. We show that our system is able to recover a communication in less than one second with low network resources consumption.**

***Index Terms*—Dependability, End-to-end, Failure, Overlay, P2P, Recovery, Routing.**

I. Introduction

Internet communications are likely to be disrupted by failures at any time. Recovery mechanisms are deployed in networks to recover users' communications when such events happen. These mechanisms are usually deployed by networks operators. When users communications are disrupted, they cannot predict how long will take the recovery process.

When communications dependability is needed by users, they would benefit to supervise recovery mechanisms used to protect their communications. This would let users to adjust the recovery process according to their dependability needs.

In this article, we study a network recovery system based on end-to-end routing in an overlay network. Our system performs routing operation on network end-nodes. When a failure disrupts a user communication, it is possible to divert it to an overlay path to bypass the failure.

Our system suits to users' dependability needs. The system deploys mechanisms for communication protection according to users' requirements. It tries to use the lowest network resources. If needed, our system is able to detect and recover failures disrupting users' communications in less than one second. The main contributions of this article are the description of the system conception, which includes inventive mechanisms such double delivery, and the study of its performances using a software implementation.

The remainder of this document is organised as follows: section 2 describes related works, section 3 presents our system and section 4 discusses the experimentation environment and results. Finally, we conclude and present future works.

II. Related work

A. *Network Failures and Recovery*

Users' communications can be disrupted for several reasons. For instance, a physical failure of a network device can prevent it to deliver communications. To mitigate failures impact on the users' communications, recovery mechanisms [1] are deployed. These systems divert communications to bypass the failing part of the network.

Routing protocols aim is to allow network nodes connectivity by computing routers' routing tables. Dynamic protocols, such Open Shortest Path First [2] or Intermediate System To Intermediate System [3] can detect network failures and modify routing tables to bypass them.

Other recovery mechanisms can be used in IP networks. For instance, the Multi Protocol Label Switching [4] provides several ways to recover from failures [1]. Several other mechanisms have also been proposed [5]–[8].

However, Internet end-to-end communications can be unavailable for several minutes [9], [10]. This may be caused by a failure that cannot be recovered by the mechanism deployed by the network operator. A network device misconfiguration can lead to similar consequences. The Border Gateway Protocol (BGP) [11], used to enable connectivity between Internet networks, is also known to need a significant time to recover from failures [12].

B. *Overlay Routing Systems*

Overlay networks are logical networks build on top of the existing network. Overlay networks are common in IP networks. For example, MPLS Virtual Private Network [13] uses a logical virtual private network on the top of the existing network. File sharing peer-to-peer systems [14] also build an overlay network between their users.

Several overlay systems are dedicated to communication routing [15]–[21]. Like standard routing protocols, their goal is to establish connectivity between overlay nodes. These systems aimed at solving BGP routing quality issues. Indeed, routes computed by BGP are not optimal in terms of performance because they are subject to administrative constraints.

Resilient Overlay Network [16] (RON) is an overlay routing system that computes routing tables using a link state algorithm. A RON node periodically disseminates routing messages to every other RON nodes. This causes it to not scale well. Some RON variants have been proposed, but do not solve this issue [17]. RON authors show that a communication can be recovered in about 20 seconds.

Others overlay routing systems [18], [19] are deployed on structured overlay network, supported by distributed hash tables [22], [23]. These systems scale well, but do not seem able to allow fast communication recovery.

Other overlay routing systems use source routing to deliver communications in the network [15], [20], [21]. To deliver communications through a succession of overlay nodes, these systems add nodes address to each packet routed in the overlay network. This can be used to bypass a failure disrupting the network. These studies mainly focus on the ability for overlay routing to bypass failures. They do not investigate how fast a communication can be recovered.

III. System Presentation

Our system objective is to increase communications dependability, on users demand and according to their needs. Users ask the system to manage their most critical communications only.

When a failure disrupts a communication, the system goal is to recover its delivery. The time needed for the recovery process, called recovery time, depends on users needs. The network resources consumption is as small as possible to satisfy these needs. Moreover, our system is able to manage any kind of IP communications, and does not need any configuration in the network.

The system uses an overlay network for communications routing. The number of overlay nodes in this network is not expected to be high. We consider that few tens of cooperating nodes is enough for our system to work. We also assume that all nodes in the system are not malicious and use it fairly. For example, a company with multiple connections to the Internet from several locations around the world can use this system to improve the reliability of its most critical communications.

A. System Conception

In this section, we describe the system conception, as illustrated by figure 1.

1) Overlay Network Organization: In our system, several nodes cooperate with each other to make an overlay network dedicated in communications dependability. Each node joins this network thanks to software running on their operating system.

Currently, two nodes must be members of the overlay network for their communications to be managed by the system. This constraint could be avoided by using Network Address Translation mechanisms, as done in other systems [17].

The overlay network topology does not follow any particular structure. Each node is able to reach one other directly. Unlike others overlay networks such RON, this is not an issue because in our system, the number of overlay nodes is small and they don't need to permanently probe each other.

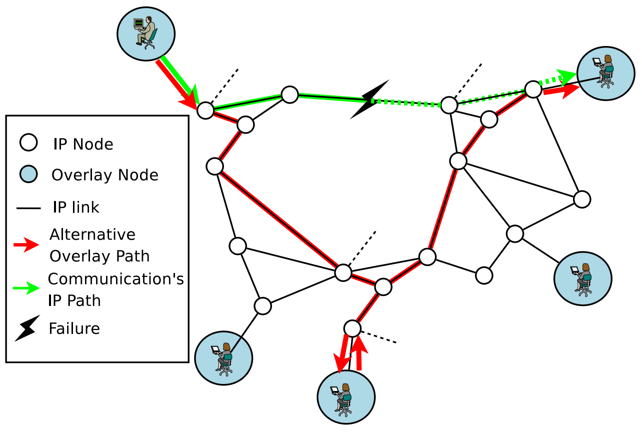

Fig. 1. How end-to-end overlay routing can recover communications disrupted by failures

2) Maximum Allowed Interruption Time: For a communication to be managed by our system, its originator must first declare its dependability need for this communication. This need is denoted as the Maximum Allowed Interruption Time (MAIT). This is the maximum time allowed for a node involved in the communication to not receive packets sent by the other node. This time is defined by users according to their dependability needs and the nature of the delivered application.

3) Usual Delivery of Communications: A communication is delivered by the regular IP path while no failure disrupts it. Our system deploys a failure detection mechanism to detect a failure that could interrupt the communication delivery. When the presence of a failure is suspected, the failure detection mechanism triggers a warning. Then, an alarm can be triggered if the presence of a failure is confirmed. Failure detection mechanisms will be described later in this section.

4) Alternative candidate paths: During the communication establishment, our system selects 8 alternative candidate paths. One of these paths will be used to deliver a communication if a failure disrupts the path usually used.

These paths are overlay paths composed of two successive Internet paths. The first one is the Internet path from the communication originator node to an overlay third-party node. The second path is the Internet path from this third-party node to the communication destination node. Currently, our system chooses third-party nodes randomly among overlay network nodes.

Using only 8 alternative paths passing by only one third-party node is supported by several previous works [20], [21], [24]. It has been shown that only one third-party node can be used by alternative paths to bypass most of failures, among those which can be bypassed with overlay routing in a given overlay network. Moreover, it has been demonstrated that only a few number of third-party nodes can be considered to bypass most of failure.

Once alternative candidate paths are selected, the system deploys failure detection mechanisms on them. However, the detection needs are lower for the alternative candidate paths than for the path used for communication delivery. Detection mechanisms must be configured accordingly. If a failure is detected on an alternative candidate path, it is dropped from the candidate list and a new path that uses another third-party node is selected.

5) Alternative Path Selection: When the failure detection mechanism deployed on the communication path triggers a warning, meaning that a failure is suspected of disrupting the communication delivery, an alternative path must be selected as soon as possible. For this purpose, the system asks to failure detection mechanisms deployed on alternative candidate paths to immediately check for failure presence. The system then selects the first alternative candidate path who has reported that no failure is disrupting it.

6) Double delivery: Once the alternative path has been verified and selected, the system starts delivering communications on the selected alternative path. It also deploys a failure detection mechanism on the selected alternative path. This mechanism is similar to the one deployed on the usual path used to deliver the communication prior to the warning triggering.

It is important to note that the communication delivery on the usual path is kept while its detection mechanism has not confirmed the failure by triggering an alarm. This period is called double delivery. Once the alarm is triggered, delivery on this path ends and its failure detection mechanism is stopped. But if the failure is invalidated, the communication delivery on the alternative path is cancelled and it becomes an ordinary alternative candidate path again.

7) Communication delivery on the alternative path: Interception of traffic and source routing are used to divert the communication and allow its delivery on the selected alternative path.

Packets created by source node applications are intercepted by our system. We use a special Linux firewall rule [25] to perform this interception. Each intercepted packet is entirely put in a new IP packet that uses the User Datagram Protocol (UDP). The IP destination address of this packet is set to the alternative path third-party node address. If double delivery is used, a special flag is set in the packet.

The system then sends this packet in the network, which delivers them to the third-party node. This node retrieves the original IP packet, puts it in a new UDP packet and sends it to the communication destination.

When the destination node receives this packet, it retrieves the original IP packet and delivers it to its applications. It also has to remember the alternative overlay path used to deliver this communication. It will thus be able to deliver its communication back to the source node through this path. The node also remembers if the double delivery flag is set in the packet, to perform double delivery for communications going back to the source.

8) Failure Detection Mechanism: Our system uses mechanisms to detect failure that could disrupt communication delivery. Detection uses special messages exchanged by nodes involved in the communication. A request/response pattern is used, as in the "ping" utility [26]. Our system does not

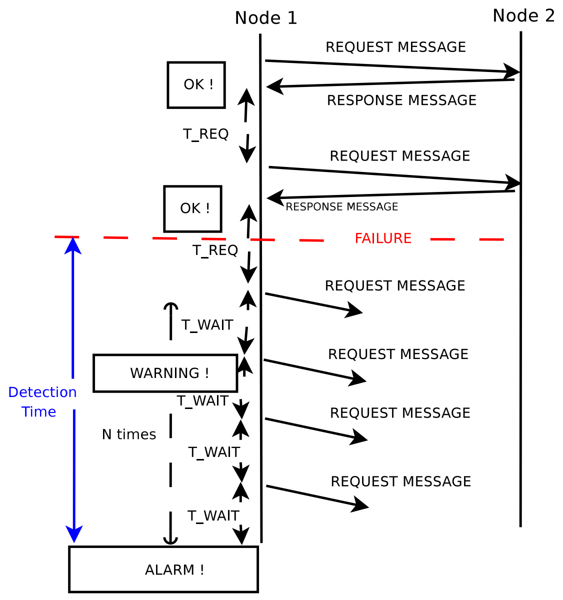

Fig. 2. The failure detection mechanism

depend on this kind of failure detection mechanisms, and other mechanisms could be used.

As shown in figure 2, a node periodically sends a REQUEST message to the other node who immediately responds with a RESPONSE message. After receiving a RESPONSE message, the delay before sending a new REQUEST message is denoted as T_REQ. After it sent a REQUEST message, a node waits during a period denoted as T_WAIT for the reception of a RESPONSE message. If no RESPONSE messages are received during this time, a warning is triggered, and the node sends a new REQUEST message. The alarm is triggered when a node has not received any RESPONSE message after sending N successive REQUEST messages.

These detection mechanism parameters are chosen to detect a failure under a defined time, called Maximum Detection Time (MDT). The table 1 shows the various parameters choices.

The MDT on the path used to deliver a communication must be chosen as a function of the needed MAIT for this communication. Indeed, once a failure is detected, the system still has to choose an alternative path and deliver communication to the destination node through it.

Since the overall operation time must be shorter than the MAIT, the failure MDT should be equal to the MAIT reduced by the time needed for alternative path selection and use. As this time cannot be predicted during communication establishment, we approximate it by two times the network round trip time delay to deliver communications between source and destination node. This time can be computed when nodes join the overlay network, for instance. Thus, with rtt the round trip time measured between the communication end nodes, the MDT is given by:

TABLE I
PARAMETERS OF THE FAILURE DETECTION MECHANISM, ACCORDING TO MAXIMUM DETECTION TIME (MDT). "VARIABLE" MEANS EACH TIME A RESPONSE MESSAGE IS RECEIVED, T_WAIT IS UPDATED WITH THE FOLLOWING FORMULA: $T\_WAIT \leftarrow rtt + 4.vrtt$, WHERE rtt IS THE AVERAGE ROUND TRIP TIME OF REQUEST/RESPONSE MESSAGES, AND $vrtt$ ITS STANDARD DEVIATION

Parameter / MDT (ms)	$MDT > 3500$	$1750 < MDT \leq 3500$	$MDT \leq 1750$
T_WAIT (ms)	1000	500	Variable
N	3	3	2
T_REQ (ms)	$MDT - 3000$	$MDT - 1500$	$max(0, MDT - 750)$

$$MDT = MAIT - 2.rtt$$

Failure detection mechanisms are also deployed on alternative candidate paths. Since no communications are delivered on these paths, detection time can be longer. We then use for these paths a Maximum Detection Time equals to 8 times the one computed for the path used to deliver the communication.

B. System Implementation

We implement our system using the Java programming language. The Linux operating system is also required to perform IP packet interception. Figure 3 shows the software architecture. The various software components are:

- SocketOverlay: This component sends and receives overlay messages, such as encapsulated IP packets or overlay management packets.
- SocketIP: This module intercepts packets from the network, to allow their delivery by overlay paths. It is also used to inject these packets from overlay paths to the network.
- Manager: This is the core of the software. It schedules the other components.
- ComProtector: Each of these components manages one communication. It manages the path used for communication delivery, the list of alternative candidate paths and the failure detection mechanisms.
- PathSelector: This module is used by ComProtector to compute the list of alternative candidate paths.
- IncidentDetector: This component implements the failure detection mechanism. It sends messages in the network to detect failures. The detection configuration is set by ComProtector.
- UserInterface: This component gives the software user the ability to start and stop the management of a communication by the system, and information on the system state.

IV. EXPERIMENTATION

In this section, we present our system experimentation. We used the software implementation presented above to perform this experimentation

A. Experimentation Setup

The goal of this experimentation is to measure two major performance criteria of network recovery systems: the recovery time and the network resources consumption. Other performance criteria will not be studied here, but will be discussed later in this section.

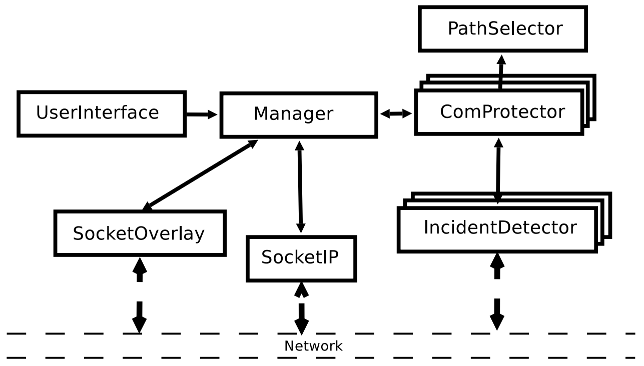

Fig. 3. Software implementation architecture

To evaluate our system performances, we used two different test beds. In the first test-bed, we use a virtualized network, using the OpenVZ software [27]. This network is composed of 30 virtualized computers, running the Linux operating system and connected to a virtual bridge. We used the Linux Traffic Control tool [28] to introduce delay in packets delivery between two network nodes. To set up the various network delays, we randomly positioned each node in a virtual plane. The minimal round trip time delivery delay between two nodes is chosen to be the Euclidian distance between these nodes so that the maximum distance is equivalent to a 500ms RTT. Each packet RTT is then associated to a random variable with a Pareto distribution using a shape parameter equals to this minimal delay, and a scale parameter equals to the tenth of the minimal delay.

The second test bed uses 8 computers connected to the Internet thanks to 5 different European Internet Service Providers.

To perform our experiments, we used the following test scenario: after a user has chosen is needed MAIT, a communication is established and its management by our system begins. After a random time, a failure is produced in the network to disrupt the communication path. Our system then tries to recover the communication delivery.

Scenarios used various MAIT values as well as different communication source and destination nodes. Several thousands of test scenarios have been measured in the virtualized test bed. A few hundreds test scenarios have been performed in the Internet test bed.

B. Experimentation Results

1) Recovery Time: We first study the system recovery time. The recovery time is the time between the beginning of the failure and the communication delivery recovery. Figures 4a and 4b show the distribution for the recovery time measurements, as a function of the Maximum Allowed Interruption Time (MAIT) needed by users.

For both virtualized and Internet test bed, our system recovery time is shorter than the MAIT, for MAIT longer than one second. This means that our system is able to recover communication delivery in one second if needed by users.

For shorter MAIT, results are mixed. If MAIT is one half second, the system is able to recover communications under this time in about half of the scenarios. Indeed, for MAIT inferior to one second, our system recovery times are distributed between 300 milliseconds and one second in the Internet test bed, and 100 milliseconds and one second in the virtualized test bed.

If MAIT shorter than one second is needed by users, our system cannot ensure that it will recover communications under this time. However, it can sometimes recover communications in few hundred of milliseconds.

In the Internet test bed, recovery times are less uniformly distributed than in the virtualized test bed. This could be explained by the greater variations of the network delay between nodes in the Internet network than in the virtualized one. Indeed, Internet delay cannot be expressed by Euclidian distance [29], as we did in virtualized network. However, these results seem close enough to validate experiments lead in the virtualized test bed. In addition, others results presented in this article concern the bandwidth consumption, which is not related to the network delay.

2) Bandwidth Consumption: In this section, we study the system network bandwidth consumption. There are four sources of consumption:

- Failure detection mechanisms.
- Packets headers added to allow communication delivery inside the overlay paths.
- Double delivery
- Overlay management

The latter will be neglected in our evaluation. As our system use a small number of overlay nodes, we consider that the resources needed to overlay management are low.

The network bandwidth used by failure detection mechanisms depends on their configuration. If the Maximum Detection Time (MDT) needed for a mechanism is short, the consumption will be high. For each communication, several mechanisms are used: one mechanism (or two during double delivery periods) to measure the path used for communication delivery, and eight to measure alternative candidate paths. As seen in section III-A8, the mechanisms' MDT are chosen according to communication's MAIT. Consequently their network consumption will depend on the communication's MAIT.

Figure 5 shows the bandwidth used by failure detection mechanism, as measured in various scenarios in the virtualized test bed. The bandwidth consumption decreases for MAIT longer than one second. With shorter MAIT, the bandwidth consumption stays approximately constant because failure detection mechanisms operate with the most "aggressive" configuration for these values.

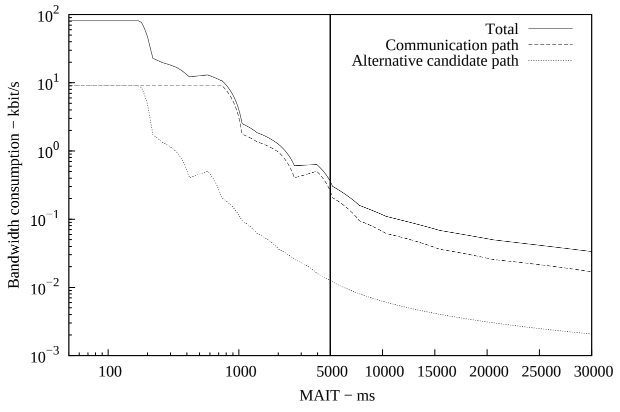

Fig. 5. Network bandwidth consumed by failure detection mechanisms, as function of the Maximum Allowed Interruption Time (MAIT). Total bandwidth is given for failure detection mechanisms deployed on one communication path and 8 alternative candidate paths

If the MAIT is equal to one second, the bandwidth consumption is 4 kbit/s. For MAIT of 5 seconds, it is 0.35 kbit/s. This consumption thus depends on the user dependability needs, and stays reasonable even for short MAIT.

Our system encapsulates IP packets in UDP datagram to allow their delivery in the overlay paths. This adds an overhead to each packet delivered this way in the network. In our current implementation, this overhead is equal to 52 bytes per packet. This value could be decreased by optimising the implementation.

Packets fragmentation could be another cause of bandwidth consumption. This happens when the total size of a new IP packet is superior to the Maximum Transmission Unit (MTU) of the network. To avoid fragmentation, applications should decrease the maximum packets size, to ensure it would stay inferior to the network Maximum Transmission Unit after packets encapsulation.

Figure 6 shows the bandwidth penalty caused by these various problems, with a network MTU equals to 15OO bytes.

Periods of double delivery will of course be costly in network bandwidth consumption. During these periods, communications are delivered two times in the network. It thus important to know how often double delivery occurs. Two cases may occur:

- The double delivery is caused by a warning triggered by the incident detection mechanism, whereas no failure is actually disrupting the communication. This case is called "illegitimate"

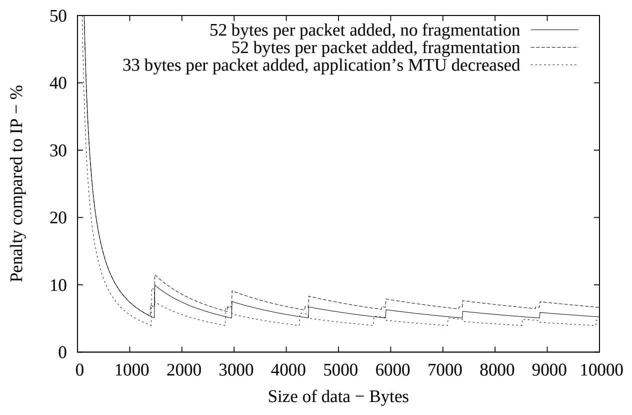

Fig. 6. Bandwidth consumption penalty caused by packet headers needed for overlay routing

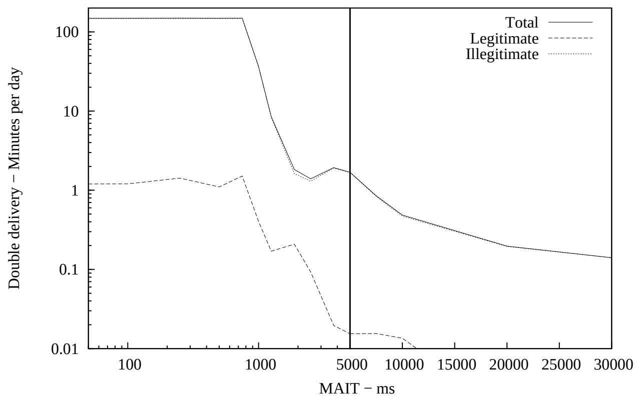

Fig. 7. Average duration of double delivery periods as a function of the Maximum Allowed Interruption Time (MAIT)

- The double delivery is caused by a warning triggered by the incident detection mechanism, and a failure is actually disrupting the communication. This case is called "legitimate"

Figure 7 shows the average duration of double delivery periods, each day, as a function of the MAIT needed by users. These results were inferred from measurements made in the virtualized test bed. The "legitimate" double delivery periods are negligible compared to "illegitimate" ones. With MAIT shorter than one second, there is on average more than 100 minutes of double delivery per day. This value then quickly decreases for longer MAIT: It is less than 2 minutes per day for MAIT of 2 seconds.

These results can be explained by the relationship between mechanisms configuration and the needed MAIT. Indeed, shorter is the MAIT and shorter is the mechanism Maximum Detection Time (MDT). The risk to trigger an "illegitimate" warning is higher if the MDT is short because mechanisms send more detection packets and they wait less time for their responses to arrive.

We studied the various sources of our system bandwidth consumption. As a summary, figure 8 shows the variation of the bandwidth consumed in the network, during a recovery scenario progress. In this scenario, the initial bandwidth needed by the communication is 50 kbit/s and the needed MAIT is 500 ms.

The overall bandwidth consumed by our system seems reasonable to us. Moreover, in this scenario, the MAIT is short and the communication bandwidth consumption is low. Expect during the double delivery periods, the various system components bandwidth consumption is low compared to the communication.

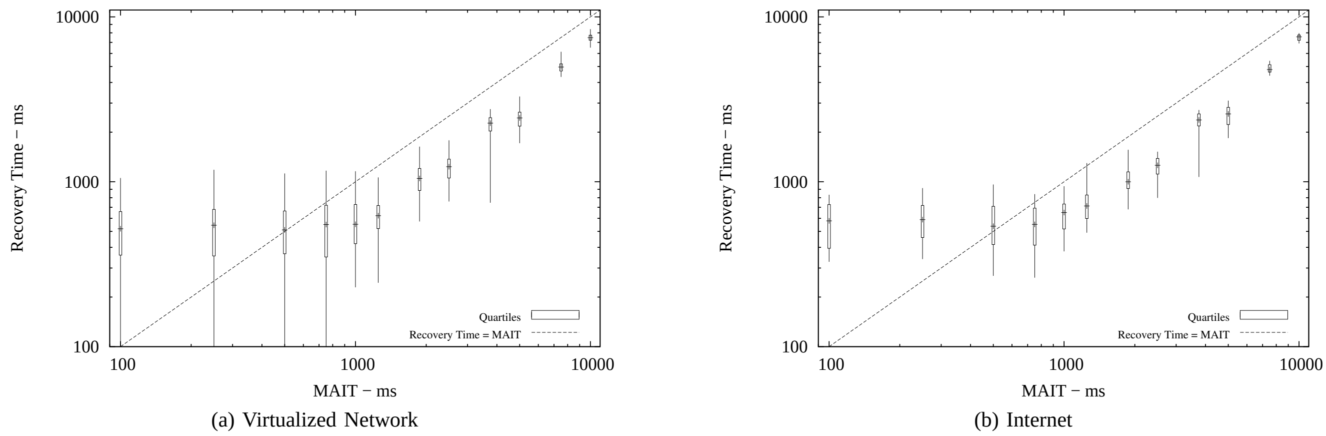

Fig. 4. Recovery time distribution as a function of the Maximum Allowed Interruption Time (MAIT). The recovery time distributions are described by quartiles box plot: The higher line shows the 25 % longer recovery times. The lower line shows the 25 % shorter recovery times. The box shows the 50 % residual recovery times. The cross shows the median recovery time. Recovery times above the Recovery Time = MAIT line denote a successful recovery

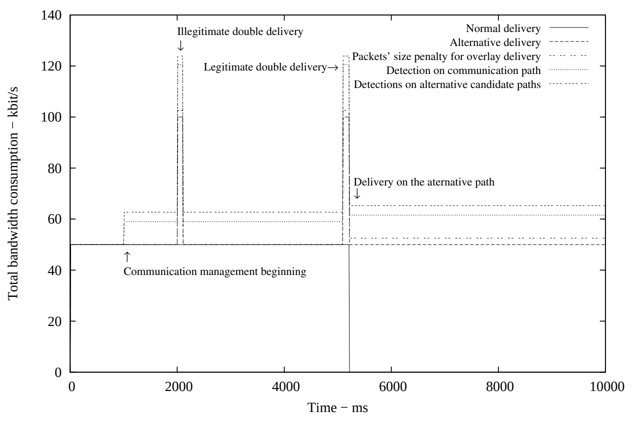

Fig. 8. Total consumption of network bandwidth during a recovery scenario, as a function of time

C. *Results Summary And Discussion*

We demonstrated that our system is able to recover communications in a time as short as half a second, with moderate network bandwidth consumption. As far as we known, it is the only one overlay routing system which is able to recover a communication that fast. If users do not need such fast recovery time, the system is able to configure itself to consume even less resources.

Another fundamental point for recovery systems performances is their ability to recover from a failure. This ability depends on the failure location and scope, as well as the overlay nodes position in the IP network for overlay-based systems. Depending on the studies and the system investigated, it has been demonstrated [10], [15], [16], [20] that an end-to-end communication disrupted by a failure can be recovered by an overlay routing system in one or two third of cases. We expect similar results with our system, and plan to achieve such measurement in a near future.

We consider that our system is effective to increase communications dependability when failures disrupt them. Our approach is based on end-to-end routing on an overlay network. It moves the network recovery task from networks' operators to users. This is especially relevant for unreliable networks.

V. Conclusion and future work

In this article, we introduced a network recovery mechanism based on overlay routing. The aim of this system is to increase end-to-end communication dependability, according to users' needs. This system can be used with any kind of communications in an IP network. We presented the system operating and we measured its ability to restore a communication disrupted by an incident under a maximum allowed time specified by users.

Our system uses overlay routing to recover communications disrupted by incidents. To deliver communications in the overlay, source routing is used. To allow fast recovery, several possible alternative paths are maintained and are ready to be used in case of primary path failure. In addition, failure detection mechanisms based on probe messages are used together with original processes such as double delivery and the failure detection warning trigger. This system can be used to increase dependability of all kinds of IP communications. We also provide a software implementation.

We studied our system ability to recover communications delivery in order to satisfy users' needs. For this, the communication recovery time must be lower than the Maximum Allowed Interruption Time, given by communication originator. We measured that in most of cases, it is possible to satisfy this constraint for MAIT higher than half a second. We demonstrated that the system network bandwidth consumption depends on the MAIT wanted by users. However, this consumption stays acceptable, even for MAIT lower than one second. Our system is effective to increase users' communications dependability, according to their needs.

We plan to improve this work by several ways. We are currently working on overlay routing system ability to recover from failure in large networks, such Internet. We want to optimize our system implementation. A more long-term task is integration of others overlay reliability mechanisms in our system.

References

[1] J. Vasseur, M. Pickavet, and P. Demeester, *Network recovery*. Elsevier, 2004, ch. 5.

[2] J. Moy, "Rfc 2328 : Ospf version 2," Internet Engineering Task Force, Tech. Rep., 1998.

[3] D. Oran, "Rfc 1142 : Osi is-is intra-domain routing protocol," Internet Engineering Task Force, Tech. Rep., 1990.

[4] E. Rosen, A. Viswanathan, and R. Callon, "Rfc 3031: Multiprotocol label switching architecture," Internet Engineering Task Force, Tech. Rep., 2001.

[5] M. Molnár and M. Tezeghdanti, "Reroutage dans ospf avec des chemins de secours," *Projet ARMOR, Rapport de recherche 4340*, 2001.

[6] D. Stamatelakis and W. Grover, "Ip layer restoration and network planning based on virtual protection cycles," *IEEE Journal on Selected Areas in Communications (JSAC)*, 2000.

[7] M. Medard, S. Finn, R. Barry, and R. Gallager, "Redundant trees for preplanned recovery in arbitrary vertex-redundant or edge-redundant graphs," *IEEE/ACM Transactions on Networking*, 1999.

[8] A. Kvalbein, A. Hansen, T. Cicic, S. Gjessing, and L. O., "Fast recovery from link failures using resilient routing layers," *10th IEEE Symposium on Computers and Communications*, 2005.

[9] N. Feamster, D. Andersen, H. Balakrishnan, and F. Kaashoek, "Measuring the effects of internet path faults on reactive routing," *Proceedings of ACM SIGMETRICS*, 2003.

[10] D. Andersen, A. Snoeren, and H. Balakrishnan, "Best-path vs. multi-path overlay routing," *Proceedings of the 3rd ACM SIGCOMM conference on Internet measurement*, 2003.

[11] Y. Rekhter, T. Li, and S. Hares, "Rfc 4271 : A border gateway protocol 4 (bgp-4)," Internet Engineering Task Force, Tech. Rep., 2006.

[12] V. Paxson, "End-to-end routing behavior in the internet," *Proceedings of ACM SIGCOMM*, 1997.

[13] E. Rosen, Y. Rekhter, and Cisco, "Rfc 2547: Bgp/mpls vpns," Internet Engineering Task Force, Tech. Rep., 1999.

[14] B. Cohen, "Incentives to build robustness in bittorrent," *Proceedings of the 1st Workshop on Economics of Peer-to-Peer Systems*, 2003.

[15] A. Collins, "The detour framework for packet rerouting," *PhD Qualifying Examination, university of Washington*, 1998.

[16] D. Andersen, H. Balakrishnan, M. Frans Kaashoek, and R. Morris, "Resilient overlay networks," *Proceedings of the 18th ACM SOSP*, 2001.

[17] R. Morris and A. Yip, "Natron: overlay routing to oblivious destinations," Massachusetts Institute of Technology, Tech. Rep., 2002.

[18] I. Stoica, D. Adkins, S. Zhuang, S. Shenker, and S. Surana, "Internet indirection infrastructure," *Proceedings of the 2002 SIGCOMM conference*, 2002.

[19] B. Zhao, L. Huang, J. Stribling, J. A., and J. Kubiatowicz, “Exploiting routing redundancy via structured peer-to-peer overlays,” *Proceeding of ICNP*, 2003.

[20] K. Gummadi, H. Madhyastha, S. Gribble, H. Levy, and D. Wetherall, “Improving the reliability of internet paths with one-hop source routing,” *Proceedings of the 6th conference on Symposium on Opearting Systems Design & Implementation (OSDI’04)*, 2004.

[21] C. Cheng, Y. Huan, H. Kung, and C. Wu, “Path probing relay routing for achieving high end-to-end performance,” *Proceedings of IEEE Global Telecommunications Conference (GLOBECOM’04)*, 2004.

[22] I. Stoica, R. Morris, D. Karger, F. Kaashoek, and H. Balakrishnan, “Chord: A scalable peer-to-peer lookup service for internet applications,” *Proceedings of the ACM SIGCOMM’01 Conference*, 2001.

[23] B. Zhao, J. Kubiatowicz, and A. Joseph, “Tapestry: An infrastructure for fault-tolerant wide-area location and routing,” *IEEE Computer*, 2001.

[24] Z. Li and P. Mohapatra, “The impact of topology on overlay routing service,” *Proceeding of INFOCOM : Twenty-third Annual Joint Conference of the IEEE Computer and Communications Societies*, 2004.

[25] R. Russell and H. Welte, “Linux netfilter hacking howto,” 2010, http://www.iptables.org/documentation/HOWTO/netfilter-hacking-HOWTO.html.

[26] “Linux man-pages project : Ping, send icmp echo request to network hosts.”

[27] Parallels, “Openvz,” 2010, http://wiki.openvz.org/.

[28] B. Hubert, T. Graf, G. Maxwell, R. Mook, M. Oosterhout, P. Schroeder, J. Spaans, and P. Larroy, “Linux advanced routing and traffic control,” 2010, http://lartc.org/.

[29] H. Zheng, E. Lua, M. Pias, and T. Griffin, “Internet routing policies and round-trip-times,” *Passive and Active Network Measurement*, 2005.