Large Scale Gigabit Emulated Testbed for Grid Transport Evaluation

Pascale Vicat-Blanc Primet, Ryousei Takano, Yuetsu Kodama, Tomohiro Kudoh, Olivier Glück, Cyril Otal

Large Scale Gigabit Emulated Testbed for Grid Transport Evaluation

P. Vicat-Blanc Primet*, R. Takano** ***, Y. Kodama**, T. Kudoh**, O. Gluck*, C. Otal*
*LIP, UMR CNRS-ENS Lyon-INRIA-UCB Lyon 5668, France
** National Institute of Advanced Industrial Science and Technology (AIST), *** AXE, Inc, Japan
{pascale.primet}@inria.fr

***Abstract*— Evaluating the performance of Grid applications running on high performance platforms interconnected by high speed and long distance networks with new transport services and protocols is highly required. This paper presents the eWAN integrated environment enabling large scale grid emulation at gigabit speed. It discusses features provided to control key characteristics (topology, round trip time, packet size, drop rate, link capacity) of an evaluation scenario. A method to increase the accuracy of rate control under various delay configuration is proposed and some experimental results are detailled.**

Keywords: grid computing, performance analysis, transport protocols evaluation, emulation.

I. Introduction

Being large scale complex distributed systems, grids are difficult to study only following a theoretical approach [1]. As a matter of fact, most of the research conducted in grids is currently performed using simulators, emulators or production platforms. The network, generally composed of high rate links, is a key component of the grid. The input/output data volume of a data grid may be in the order of terabytes and will likely reach petabytes in the near future. Transporting enormous quantities of data among grid nodes pose specific challenges on the transport protocol and its related mechanisms. To tackle these problems, solutions are explored to enhance the transport services and protocol [2]. Alternative technologies, such as load balancing, data caching, and data replicating, are proposed for grid data movement optimization. Provisioning end-to-end services with known and knowable characteristics is also on the agenda. Targeting bulk data transfer, the problem of network resource reservation [3] is studied within the grid scope. Transport solutions benchmarking is then of great importance for the overall system and for individual application performance evaluation.

Figure 1 gives an abstract view of a grid network. In this model, high performance local area networks are interconnected via a high speed long distance backbone with access links representing potential bottlenecks. This paper centers on transport protocol and service performance evaluation in the context of such high performance grids. It is associated with the DataGrid Explorer, a project of the Grid 5000 initiative [4], an experimental grid platform gathering 3000 processors over eight geographically distributed sites in France and the NAREGI project in Japan.

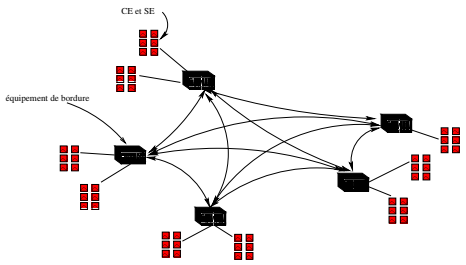

Fig. 1. Abstract view of a Grid

II. Simulation and emulation of Grids and their networks

A. Motivations

In large distributed systems, numerous parameters must be considered and complex interactions between resources make analytical modelling impractical. Thus simulators and real testbed are preferred. Simulators like Bricks or SimGrid, focus on a specific behavior or mechanism of the distributed system and abstract the rest of the system. Their fundamental advantage is their independence from the execution platform. Experiments conducted on real platforms [5] have demonstrated that when exposed to real life networks with large bandwidth, delay and packet losses, many transport protocols and consequently grid applications perform poorly. That shows the importance of real life testing. However, there are two strong limitations of real life platforms as experimentation tools 1) their low software reconfiguration capability and 2) the lack

of deep control and monitoring mechanisms for the users. Often, it is costly and difficult to reproduce behavior because the testbed is not a controlled environment and load conditions introduce lot of noise that is difficult to isolate.

Then, a strong need for emulation tools for testing application with different latency and load conditions emerged.

B. eWAN features

eWAN, the integrated environment we propose, combines a set of tools to build an emulated grid environment operating at gigabit speed. This software allows grid researchers to set up and use a scalable, flexible and controllable instrument. They have the possibility to experiment and evaluate any transport protocol or service in interaction with grid applications. Thus, distributed algorithms performing in local, metropolitan, wide area or transoceanic grid configurations can be compared. Such evaluation scenario examples are running an MPI benchmark or testing distributed file systems benchmarks at different scales and under various workloads.

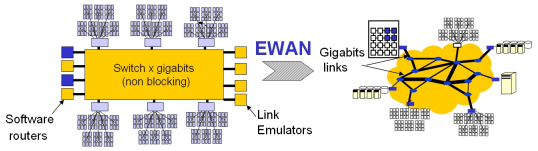

Fig. 2. Converting a cluster in a wide area grid network with EWAN

eWAN is designed to enable a large spectrum of topologies and configurations while achieving very high rates. eWAN provides features to control key characteristics of grid or transport protocol evaluation scenarii. As explained on the figure 2, an emulated grid environment is defined as an overlay network based on the local resources of a cluster. The main functionnalities that have been identified are:

- link emulation with key characteristics control: latency (from 1ms to 500ms) loss rate (with different distributions), capacities (from 10Mb/s to 10Gb/s).
- topology emulation (chain, star, ring, mesh, dumpbell, fish bone...)
- IP version (v4 and v6) and jumbo frame support
- traffic generation
- process application running
- traffic and performance monitoring and logging

Deployment

Scripts generation

Machine 0 IP : *140.77.12.61* emulate : Client c0
IP0: *192.168.4.2*

Machine 1 IP : *140.77.12.62* emulate : la1
IP0: *192.168.3.2*
IP1: *192.168.5.2*

```
ip address flush label eth*;
ifconfig eth0 140.77.12.62 netmask 255.255.255.0;
route add default netmask 0.0.0.0 gw 140.77.12.1 dev eth0;
ifconfig eth0:0 mtu 1500 192.168.3.2;
ifconfig eth1 mtu 1500 192.168.5.2;
if [ -z "\`cnistnet -Fd  2>/dev/stdout| grep command\`" ];
 then modprobe -r nistnet;
 modprobe nistnet;
 cnistnet -u;
 cnistnet -a 0.0.0.0 0.0.0.0  --delay 5 --drop 5 > /dev/null;
else echo NIST Net not available;
fi;
route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.3.1 dev eth0;
route add -net 192.168.6.0 gw 192.168.5.1 netmask 255.255.255.0 dev eth1;
```

Machine 4 IP : *140.77.12.65* emulate : Router rc0
IP0: *192.168.1.1*
IP1: *192.168.2.1*
IP2: *192.168.3.1*

Machine 5 IP : *140.77.12.66* emulate : Access Point p0
IP0: *192.168.1.2*
IP1: *192.168.4.1*

```
ip address flush label eth*;
ifconfig eth0 140.77.12.66 netmask 255.255.255.0;
route add default netmask 0.0.0.0 gw 140.77.12.1 dev eth0;
ifconfig eth0:0 mtu 1500 192.168.1.2;
ifconfig eth1 mtu 1500 192.168.4.1;
tc qdisc replace dev eth1 root tbf rate 100mbit latency 1ms burst 15400000;
route add -net 192.168.0.0 netmask 255.255.0.0 gw 192.168.1.1 dev eth0;
```

Machine 6 IP : *140.77.12.67* emulate : Access Point p1
IP0: *192.168.5.1*
IP1: *192.168.6.1*

Fig. 3. Configuration deployment

The eWAN's graphical user interface lets the user concentrate on the grid network topology and configuration description. Then eWAN software maps the specified configuration in a virtual network. It automatically generates the differents scripts required to activate the corresponding routing tables, rate controllers, link emulators and cross-traffic generators. Figure II-B is a view of the scripts automatically generated during the configuration phase. Finally, the scripts are executed by the components of the cluster and the emulated grid is initialised with the correct parameters.

III. Implementation, experimentations et results

To achieve correct performance and enable test at gigabit speed with minimum noise and overhead, the different functional entities are deployed on separate, non shared machines and networks. Compared to Emulab, based on resource sharing principle, the particularity of EWAN is to exploit within a limited time (one to few hours depending on the experiment needs) any cluster composed of several tens or more common PCs. Each function, link emulation, software routing for topology emulation, traffic generation and traffic sink, is implemented in an independant Linux box or by a dedicated hardware. One on the main issue of such

high performance environment is the high speed link emulation feature, which we detail in the next subsection.

A. Emulation and control of high speed links

For high speed link emulation, there are two possibilities: hardware assist solutions or software solutions. Hardware assist solutions, based on FPGA [6] or network processors [7] have been proposed. For example, GtrcNET-1 is a hardware network emulator developed by AIST [6] . It has a large scale FPGA (Field Programmable Gate-Array), four high speed memory blocks and four GbE ports. By configuring the circuits on the FPGA, various functions such as network emulation, traffic measurement, packet capturing and traffic generation can be achieved.

For network emulation, it can emulate a link with variable delay, bandwidth and frame loss rates at the wire speed of GbE. Since the emulation is fully implemented by hardware, the emulated delay is clock-accurate and the error is less than 1 μs. The emulated delay is independent from the incoming traffic bandwidth or packet length. GtrcNET-1 can also continuously monitor the bandwidth of the traffic going through the emulating link every millisecond. AIST also developed 10GbE network testbed GtrcNET-10. It also supports accurate delay emulation at the wire speed of 10GbE.

On the other hand, many software emulators exist. They do not use any specific hardware support to introduce a delay or losses: they just intercept packets and store them in waiting queues for applying a software processing specified by the user. The most popular emulators are Dummynet [8], NIST Net [9] and Netem [8]. Dummynet is a standard part of FreeBSD and is implemented as part of the packet filtering mechanism. Dummynet does packet filtering on output. But it is completely self-contained and is not easily extended.

NIST Net is a Linux kernel extension that provides complex delay, loss, and other emulation options. NIST Net is running in LINUX kernel 2.4 but is not available in kernel 2.6. NetEm is a recent enhancement of the traffic control facilities of Linux that allows adding delay, packet loss and other scenario's. NetEm has been largely improved in last few month and is available in kernel 2.6. It is very easy to deploy as it is implemented as a queuing discipline in Linux and can be activated by the LINUX tc utility. The main drawback of software network emulators is that the emulated delay is not stable because of indeterministic software scheduling. The delay variation may becomes large when the traffic is high or multiple functions are run simultaneously. The other limitations of these software emulators are mainly their performance (emulate a long delay at very high throughput requires a high-speed CPU and a lot of memory) and their accuracy (timer precision, necessity to use a real-time OS for avoiding periodic tasks to be late). On an other hand, hardware assist solutions are much accurate but are expensive hardware solution and their usage may be limited to few links. For the design of a complex high speed emulated network, we choose to let the user creates an overlay network composed of software routers interconnected with hardware (when available) or software link emulator in a local cluster. As most grid researchers use Linux kernel and in order to facilitate the dynamic deployment of an eWAN configuration, we concentrate on Linux solutions: NIST net was integrated in the first version of eWAN and NetEm is used in the current version. To provide a tradeoff between cost and precision, the eWAN software combines software emulators that enable the configuration of complex high performance topologies and GtrcNET-1 for high precision experiment and emulator calibration.

IV. Instrument calibration

Linux is not a real-time system and this provides some constraints on the performance of a real-time emulator such as NetEm. Kernel timers are limited by the system time tick rate of 1000Hz (1ms) on Linux 2.6. The Linux 2.4 kernel uses a slower 100Hz clock (10ms). Therefore NetEm can not be used to emulate relatively short delay networks of less than 1ms as we observed.

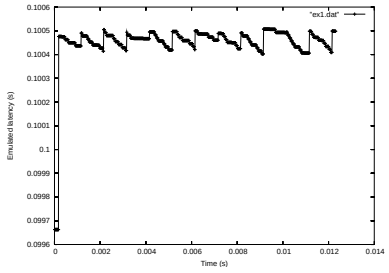

Fig. 4. Emulation of a 1Gb/s link with 100ms latency with netem

This problem is not unique to NetEm, the rate control disciplines also suffer when running over high speed links. It is not possible to limit a 10Gigabit network to 100Mbit with accuracy without higher resolution timers. NIST Net gets around this by programming one of the alternate time

sources available on the PC architecture to provide a high speed clock. This has a performance impact because of the high interrupt load.

As eWAN may be deployed on any cluster, microbenchmarks have been define to analyse performance sensitivity of the instrument deployed on a given hardware configuration. For this measurement, for example GtrcNET-1 can measure latency of a network path at the precision of less than $1mu$s by inserting GtrcNET-1 at the both end of the path under test. It is important the users know in advance circumstances where the emulation breaks down. The microbenchmark are run before experiments to calibrate the network configuration , enabling the control and the debug the performance results after the tests.

A. eWAN calibration methodology

Lot of parameters are interacting and affecting the performance of distributed applications and transport protocols. There are parameters related to the hardware, parameters related to the system version and configuration. There are parameters related to the network hardware. Then the protocol itself and the protocol configuration play an important role. Finally, in the case of emulated network like eWAN, parameters related to the emulation tools and to traffic control tools have to be considered. The methodology we adopt to evaluate transport protocols and communication software is first to measure the hardware and system parameters and detect any local bottleneck. We then adjust the best hardware and system configuration that allows correct performance analysis. Indeed, in some case, bandwidth control is required. The table I gives a list of hardware parameters and table II gives a list of system parameters that have an impact on the communication performance when running the eWAN software. Tables III gives the Iperf end to end bandwidth we obtained in two different clusters with different NICs.

eWAN tools for link emulation, topology emulation, bandwidth limitation may also have a strong impact on the results. For example, for topology emulation we are using virtual interfaces and software routing. In all our clusters, we observed that these virtual interfaces, enabling the assignment of two or more IP address to a physical interface, do not slow the flows. However, we see the hardware configuration may have strong impact on the software routing.

B. Accurate rate limitation under various delay configuration

When benchmarking transport protocols or grid applications, it is often usefull to play both with latency and rate. We studied the behavior of NetEm with rate limitation under different delay values. Moreover, we observed an uncorrect behavior for rate limited flows that does not appear with GtrcNET-1 . As NetEm buffer packets are delayed, the profile of the entering flow may be altered, creating burst at the output of the link emulator. To preserve the initial controlled rate, the software packet pacing mechanism has been associated with the NetEm module. Precise Software Pacer (PSPacer) [10] achieves precise network bandwidth control and smoothing of bursty traffic without any special hardware. PSPacer adjusts the interval between outgoing packets precisely by inserting *gap packets* between the outgoing packets. IEEE 802.3x PAUSE frame is used as a gap packet. Pause frames are discarded at the input port of switches or routers and real packets go through keeping uniform intervals.

With the PSPacer, it is possible to provide bandwidth control and smoothing for each of 100 or more connections by use of commodity PC. In the case of GbE, the transmission bandwidth can be set for a range from 8 Kbps to 930 Mbps for each of IP communication flows, and the packet transmission interval can be controlled at an accuracy of time for 1-byte data transmission (8ns).

The PSPacer is installed as a loadable kernel module for the Linux platform, to be ready for being introduced independent of the device driver. No modification to the applications which use the network are required. PSPacer also supports IPv6 and jumbo frame. The psp queing discipline associated with netem preserve the rate limitation as demonstrated in the figure 6.

V. PROTOCOL BENCHMARKING WITH EWAN

Evaluating the performance of new TCP proposals is not easy. One principal difficulty is the lack of an agreed methodology. This has become a hot research topic. [11] proposes and analyses a set of performance measures (fairness, efficiency, overhead....) . A new group at the IRTF is working on new model for TCP evaluation [12]. It has been also argued that most existing work fails to control for variations in performance associated with differences in network stack implementation that are unrelated to the congestion control algorithm. The eWAN environment in the Grid5000 project makes it possible to deploy dynamically new protocol

Cluster	AIST@Tsukuba	GRID5000@Lyon	GRID5000@Orsay
CPU	AMD Opteron 246 2.0GHz dual	XEON 2.8GHz dual	Opteron 2GHz dual
Memory	6GB	2GB	2GB
IO Bus	PCI-X 133MHz/64bit	PCI 66MHz/32bit	PCI-X 66MHz/32bit
NIC	Broadcom BCM5704	Broadcom, tg	Broadcom, tg
Switch	Cisco Catalyst C3750G-24T-S	Cisco Catalyst 6506	Cisco Catalyst 6509 (with sup 720)

TABLE I

HARDWARE PARAMETERS

Cluster	OS version	Kernel	Traffic control tools	Bechmarking tool
AIST@TSUKUBA	FedoraCode 3	kernel 2.6.14.3	iproute2 TBF + PSP	iperf 2.0.2
GRID5000@Lyon	DebianCode 3	kernel 2.6.11	iproute2 TBF	iperf 2.0.2
GRID5000@Orsay	DebianCode 3	kernel 2.6.11.12	iproute2 TBF	iperf 2.0.2

TABLE II

MAIN SOFTWARE PARAMETERS

	Lyon ETH1-ETH1	Lyon ETH2-ETH2	Orsay ETH1-EHT1
TCP	941 Mbits/s	762 Mbits/s	941 Mbits/s
UDP	957 Mbits/s	957 Mbits/s	957 Mbits/s

TABLE III

EXAMPLE OF IPERF BASIC TEST ON DIFFERENT INTERFACES IN LYON AND ORSAY GRID5000 CLUSTERS

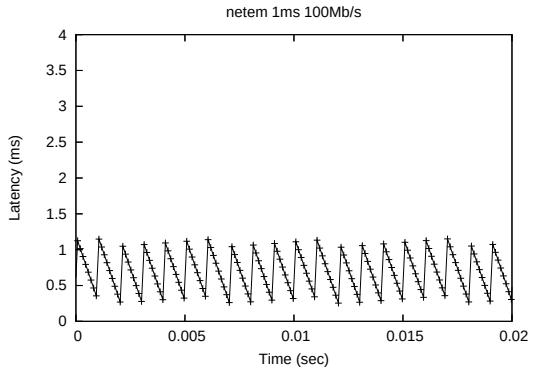

Fig. 5. netem1 alone with 1ms latency and 100Mb/s limited rate

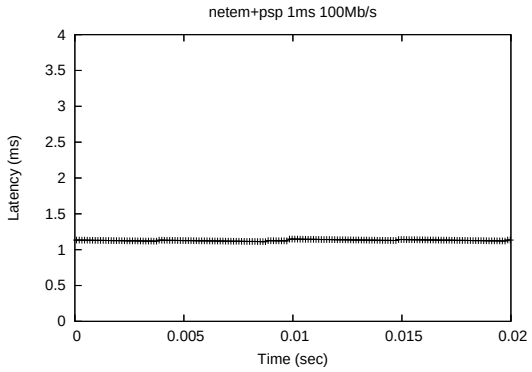

Fig. 6. netem combined with psp

stacks and to define custom configurations. Indeed, the Grid5000 [4] software offers the raw access to a set machines and an automatic kernel images deployment tool that makes it possible to change protocol stacks very easily and at large scale. In the next subsection we detail an experiment that has been conducted in the AIST supercluster and in the Grid5000@Lyon cluster.

A. Experiment description

The experimental system is composed of two workstations of the same LAN connected to the same access link and two workstations located in two different sites. The provided bandwitdh in the wide area network is assumed to be dedicated and well provisionned. The bottleneck we consider in this case is the access link in the site comprising the sources. The RTT for all connections is around 100ms. The system we consider provides two types of transport service. The first transport service accepts and starts immediately all transfer requests as they arrive. The second transport service, records transfer requests and schedules them in order to provide a deterministic transfer time by insuring

bandwidth protection. Standard TCP protocols are used.

Recall the notations for the input of the problem:

- set of requests $\mathcal{R} = \{r_1, r_2\}$, with $bw(r) = 1Gb/s$ as the bandwidth demanded by request $r \in \mathcal{R}$.
- an ingress points $\mathcal{I} = \{i_1\}$, with $B_{in}(i) = 1Gb/s$ as the capacity (i.e., bandwidth) of the access point $i \in \mathcal{I}$.
- each request $r \in \mathcal{R}$ has a **release** time $rt(r) = 0$ and a **deadline** time $dt(r)$ say $60s$. The time window of request r is then $[rt(r), dt(r)]$ say $[0, 60]$;
- each request $r \in \mathcal{R}$ has its volume $vol(r)$ say $500GBytes$;
- each request has a maximum acceptable rate $max\text{-}rate(r)$ say $1Gb/s$, meaning that the application is not able to send data at a rate higher than this value.

For each submitted request r, the transport service gives a transfer planning defined as:

- a boolean $satisfied(r)$, set to 1 if and only if the request can be scheduled within the required time-window, and if this is the case:
- a starting time $t_s(r)$
- a rate $scheduled\text{-}rate(r)$
- a finishing time $t_f(r) = t_s(r) + \frac{vol(r)}{scheduled\text{-}rate(r)}$

At the beginning of the experiment, each sending node is ready to send 500GBytes from memory to memory to an other site. In the first scenario, the transport service TS_1 starts all transfers at the same moment using the first scheduling policy (fair sharing). In the second scenario, the transfers are scheduled using the second transport service TS_2, (here we assume a scheduling algorithm that simply and greedily serialize the requests). We consider two objective functions: the first is to minimize the maximum transfer duration

$$f_1 = min(max(t_f(r) - t_s(r)))$$

and the second is to maximize the network utilization ratio of the bottleneck (access link) within the time period $[min_{t_s(r)}, max_{t_f(r)}]$.

$$f_2 = \frac{\sum_{r \in \mathcal{A}} scheduled\text{-}rate(r)}{B_{in}(i)}$$

.We expect the TS_1 service will allocate 500Mb/s to each request and will maximize f_2, while TS_2 service will allocate 1Gb/s to each request and will optimize f_1. We try to prove this theoretical result on our emulated testbeds.

B. Experiment analysis

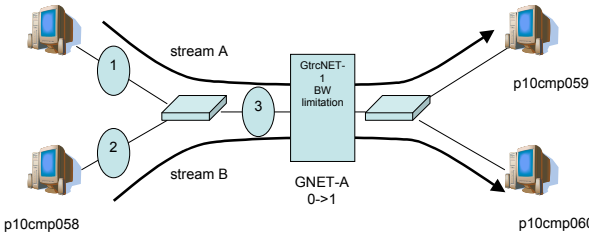

Fig. 7. Simple topology for transport service performance evaluation

We conducted this simple experiment in the GRID5000@Lyon cluster and then AIST Supercluster with our link emulation and traffic control tools to explore the problem of protocol benchmarking in an emulated environment. Figure 7 describes the topology we used. We choose two protocols: BIC and RENO, two bandwidth control methods: the standard Linux token bucket queuing discipline and the PSPacer. The goal was to characterize the behavior of the different protocols and mechanisms under various interaction patterns : one flow, two flows generated by the same machine, two flow generated by different machines. We set up and instrumented a simple dumpbell topology with a rate controlled bottleneck. The experiment in Lyon was very disappointing as for TS_1 and TS_2 the flows obtained the same ridiculous Iperf throughput of 24Mbits/s with the BIC and RENO protocols. The monitoring tools we are using in the AIST Supercluster enable a fine observation of the flow behavior. We are able to understand congestion situations that were unexplained in the first cluster with accurate time diagrams . Figure 8 show the impact of the latency, the socket buffer size. Figure 9 show the impact of the source generator. Figure 10 shows the impact of the bandwidth limitation tool. This explain the problem we encountered in Lyon with two independant sources, RENO and token bucket limitation. Figure 11 shows the impact of the bottleneck congestion level. In all these experiments we observed a very good Jain's fairness index of about 0.99. The token bucket rate limitation approach seems to

be very penalizing in very large bandwidth-delay product environments. The PSPacer, acting as a leaky bucket rate limiting tool performs much better in such conditions. The theoretical results were here experimentally demonstrated. We demonstrate also that controlling flow rate could be of great value when the source access rate (here 400Mbits/ or 490Mbits/) are in same order of the bottleneck capacity (700Mbits/s).

VI. Related works

EWAN is an experimental software for testing and validating new protocols or distributed applications. We can distinguish two large classes of network emulation [13]: the network emulation which allows simulated components to communicate with protocols implemented in the real world and the network emulation within a software environment, an extension of the network emulation, which allows to directly execute a real protocol. We list two kinds of network emulators:

- emulators of network links which allow to emulate a network cloud seen as a set of emulated links: like Dummynet [8] or NIST Net [9].
- emulators of virtual networks which allow to simulate/emulate a network cloud in real time and to inject real traffic in it: VINT/nse [14], IP-TNE [15], Virtual Routers [16], MicroGrid/MaSSF [17], ModelNet [18], PlanetLab [19], Emulab/Netbed [20], WAN in LAB [1].

EWAN belongs to this second class but uses existing network link emulators.

VINT/nse [14], based on the `ns` simulator, uses real-time events processing: a specific interface capture packets in real-time, input them in `ns` which process the packets and inject them in the network using *raw sockets*. IP-TNE is a network emulator using parallel simulation for achieving a better scalibility: everything is executed on the same computer within a real-time environment. The *Virtual Routers* emulate a network thanks to virtual routers implemented as linux processes in user space ; IP packets go through the virtual routers which communicate together via UDP or IPC (within the same node). Modelnet [18] allows the user to design a complete network topology but it requires modifications in the FreeBSD operating system. As well as EWAN, ModelNet assigns specific processing to different nodes of the emulator (border nodes which execute the user application,

[1] `http://netlab.caltech.edu`

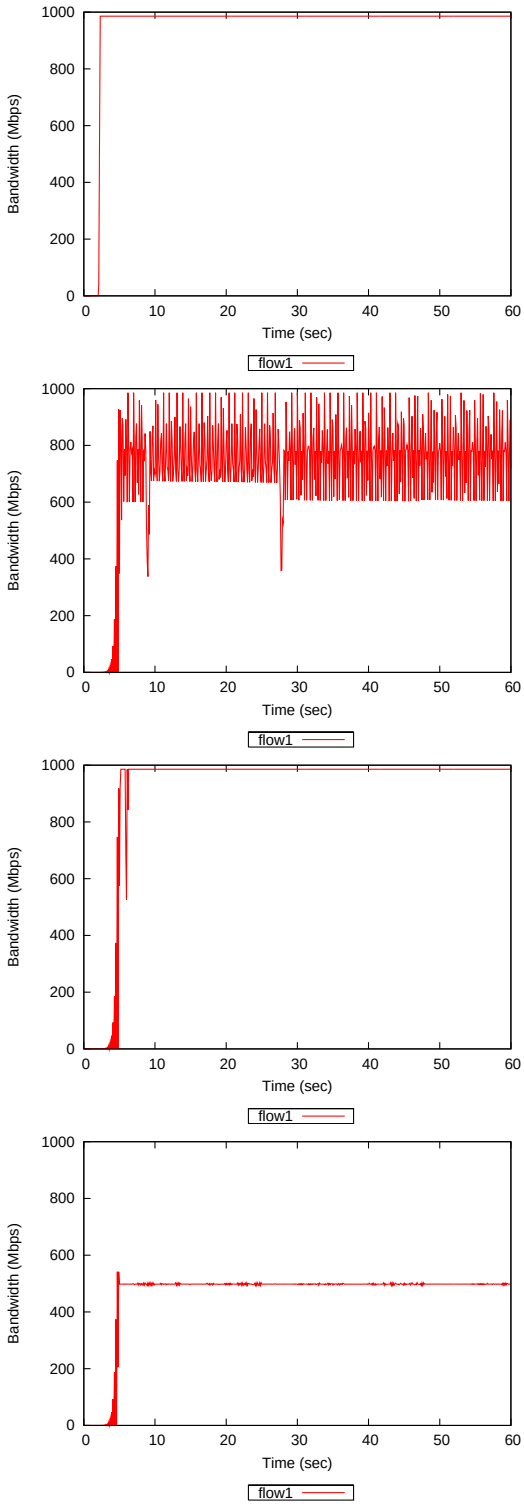

Fig. 8. Basic measurements with one flow: 1) RENO and 10ms and socket buffer of 16MB, 2) RENO and 100ms and socket buffer of 16MB, 3) RENO and 100ms and socket buffer of 25MB, 4) RENO and 100ms and socket buffer of 25MB and token bucket rate limitation of 490Mbits/s

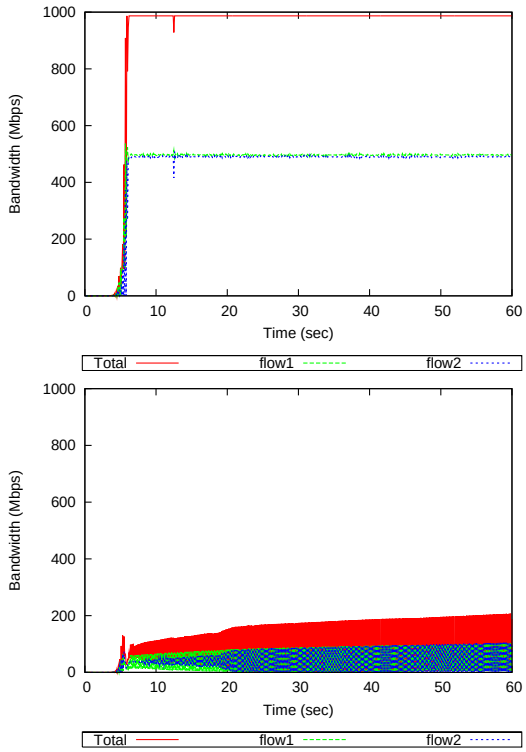

Fig. 9. 2 flows generated by the same machine, 2 flows generated by two independant machines

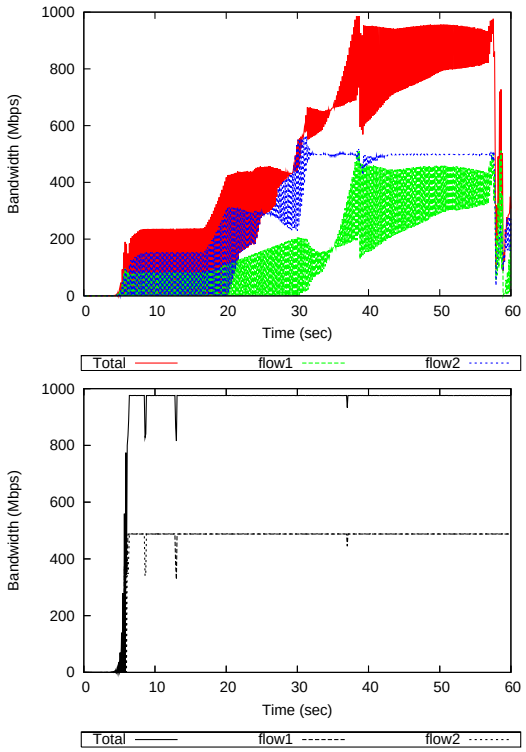

Fig. 10. Influence of the rate limitation tool1) Token bucket based rate limitation, 2) PSPacer based rate limitation

core network routers which emulate the virtual network cloud, ...). Emulab [20] , uses Dummynet, ns and Vlans to supply a fully-configurable network environment. The user can define via a Web page a virtual network topology and the characteristics of the network nodes of the emulator. Emulab [20] emulates complex networks with multiple machines and flows using Dummynet. The main difference between Emulab and eWAN is that Emulab aim at time sharing the plateform for hosting many users. Consequently, high performance network protocol evaluation may be hard to do on such uncontrolled plateform. EWAN is designed over a cluster composed of several tens or so common PCs to build a flexible and high performance grid network cloud: every cluster node, according to its charateristics, is assigned to only one specific function (link emulator, core access router, core router, traffic generator, ...) thus achieving the whooly high speed network cloud emulation.

WAN in LAB [2] projects design a real network confined in a laboratory to study network problems using few computing nodes, real routers and optical switches and very long optical fibers. WAN in LAB enable accurate experimentation but is much more costly than our eWAN environment.

[2] http://netlab.caltech.edu

VII. Conclusions

From TCP protocols to resource management strategies, grids pose new requirements on networking technologies. This article proposed a flexible and easy to use tool for grid network emulation that enable transport protocol developers as grid application designers to test and evaluate their model and software under various controlled conditions. We have detailled the main components of the environment and the calibration task. The eWAN software integrates hardware and software link emulators and software routers. The usage of an easy to use software rate controller (PSPacer) may help in improving accuracy of the instrument. Simple transport solutions evaluation in eWAN emulators deployed in the Grid5000 and NAREGI contexts have been proposed to demonstrate the usage and the potential of our integrated software.

VIII. Acknowledgment

This work has been funded by the French ministry of Education and Research, INRIA, and CNRS, via ACI GRID's Grid5000 project and ACI

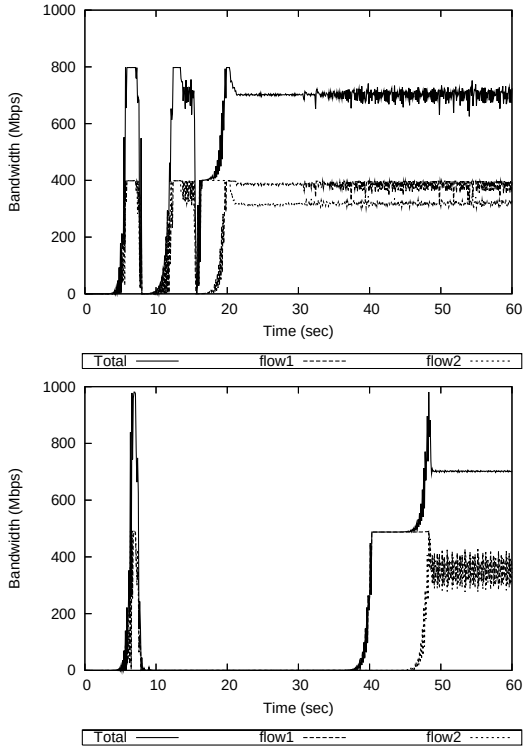

Fig. 11. Influence of the bottleneck congestion level: 1) 10%, 2) 20% of a 700Mbit/s bottleneck

MD's DataGridExplorer, a grant from the Ministry of Education, Sports, Culture, Science and Technology (MEXT) of Japan through the NAREGI (National Research Grid Initiative) Project and the PAI SAKURA 100000SF with AIST-GTRC.

REFERENCES

[1] F. Berman, G. Fox, and A. J. Hey, *Grid Computing: Making The Global Infrastructure a Reality*, W. S. in Communications Networking and D. Systems, Eds., 2003, iSBN: 0-470-85319-0.

[2] M. Weltz, E. He, P. Vicat-Blanc Primet, and al., "Survey of protocols other than tcp," GGF Informational Document, url = http://forge.gridforum.org/projects/dtrg-rg/document/draft-ggf-dtrg-survey-0/en/1/draft-ggfdtrg-survey-1.pdf, Global Grid Forum, Tech. Rep., April 2005, submitted as GFD document.

[3] L. Marchal, P. Primet, Y. Robert, and J. Zeng, "Optimizing network resource sharing in grids," in *IEEE GLOBECOM'05, USA*, Nov. 2005.

[4] F. Cappello and al., "Grid5000: A large scale, reconfigurable, controlable and monitorable grid platform," in *GRID2005 worskhop of the IEEE SuperComputing Conference*, Nov. 2005.

[5] J.-P. Martin-Flatin and P. Vicat-Blanc Primet, *Special issue "High Performance Networking and Services in Grids: the DataTAG project*, Elsevier, Ed., 2005, vol. 21, no. Issue 4.

[6] Y. Kodama, T. Kudoh, T. Takano, H. Sato, O. Tatebe, and S. Sekiguchi, "Gnet-1: Gigabit ethernet network testbed," in *In Proceedings of the IEEE International Conference Cluster 2004*, San Diego, California, USA, September 20-23 2003.

[7] K. Nakauchi and K. Kobayashi, "Studying congestion control with explicit router feedback using hardware-based network emulator," in *Proceedings of 3rd International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet'05)*, Lyon, France, Feb. 2005.

[8] L. Rizzo, "Dummynet: A simple approach to the evaluation of network protocols," *ACM SIGCOMM Computer Communication Review*, vol. 27, no. 1, pp. 31–41, January January 1997. [Online]. Available: citeseer.ist.psu.edu/rizzo97dummynet.html

[9] M. Carson and D. Santay, "Nist net: a linux-based network emulation tool," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 111–126, July July 2003.

[10] R. Takano, T. Kudoh, Y. Kodama, M. Matsuda, H. Tezuka, and I. Ishikawa, "Design and evaluation of precise software pacing mechanisms for fast long-distance networks," in *Proceedings of 3rd International Workshop on Protocols for Fast Long-Distance Networks (PFLDnet'05)*, Lyon, France, Feb. 2005.

[11] D. L. Yee-Ting Li and R. N. Shorten, "Experimental evaluation of tcp protocols for high-speed networks," 2005.

[12] F. Sally, "Tools for the evaluation of simulation and testbed scenarios," in *http://www.ietf..org/irtf/draft-irtf-tmrg-tools.txt*, Sept. 2005.

[13] K. Fall, "Network emulation in the VINT/NS simulator," in *In Proceedings of the fourth IEEE Symposium on Computers and Communications*, Red Sea, Egypt, July 1999, pp. 244–250. [Online]. Available: citeseer.ist.psu.edu/fall99network.html

[14] L. Breslau, D. Estrin, K. Fall, S. Floyd, J. Heidemann, A. Helmy, P. Huang, S. McCanne, K. Varadhan, Y. Xu, and H. Yu, "Advances in network simulation," *IEEE Computer*, vol. 33, no. 5, pp. 59–67, May 2000. [Online]. Available: citeseer.ist.psu.edu/breslau00advances.html

[15] R. Simmonds and B. W. Unger, "Towards scalable network emulation," *Computer Communications*, vol. 26, no. 3, pp. 264–277, February 2003.

[16] F. Baumgartner, T. Braun, E. Kurt, and A. Weyland, "Virtual routers: A tool for networking research and education," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 127–135, July July 2003. [Online]. Available: citeseer.ist.psu.edu/654202.html

[17] X. Liu and A. Chien, "Traffic-based load balance for scalable network emulation," in *In Proceedings of the ACM Conference on High Performance Computing and Networking (SC2003)*, Phoenix, Arizona, November 2003.

[18] A. Vahdat, K. Yocum, K. Walsh, P. Mahadevan, D. Kostic, J. Chase, and D. Becker, "Scalability and accuracy in a large-scale network emulator," in *In Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, December 2002.

[19] B. Chun, D. Culler, T. Roscoe, A. Bavier, L. Peterson, M. Wawrzoniak, and M. Bowman, "Planetlab: An overlay testbed for broad-coverage services," *ACM SIGCOMM Computer Communication Review*, vol. 33, no. 3, pp. 3–12, July July 2003.

[20] B. White, J. Lepreau, L. Stoller, R. Ricci, S. Guruprasad, M. Newbold, M. Hibler, C. Barb, and A. Joglekar, "An integrated experimental environment for distributed systems and networks," in *In Proceedings of the 5th Symposium on Operating Systems Design and Implementation (OSDI)*, December 2002, pp. 255–270.