A Network Evaluation for LAN, MAN and WAN Grid Environments

Evgueni Dodonov[1], Rodrigo Fernandes de Mello[1], and Laurence Tianruo Yang[2]

[1] Universidade de São Paulo
Instituto de Ciências Matemáticas e de Computação
São Carlos – SP – Brazil
[2] Department of Computer Science, St. Francis Xavier University
Antigonish, NS, B2G 2W5, Canada

Abstract. The performance of network protocols on different usage scenarios differs significantly, making the protocol choice a difficult question. This had motivated a work that aims to evaluate the TCP, UDP and Sendfile (a POSIX-defined zero-copy TCP access technique) protocols on LAN, MAN and WAN environments, in order to find the most adequate configuration for each protocol. The protocols were evaluated on default configurations, without any application-specific optimizations.

1 Introduction

The increasing usage of grid networks [17] has motivated academic research and commercial projects. Most of these projects are directed to Grid environments for general purpose computing, such as Globus [24], X-Grid [6] and ProGrid [9]. Usually these projects are adopted on widely available hardware and, in most cases, use the default network and system settings.

The nature of the grid networks presumes the existence of a huge number of independent computing resources, each one composed by different processing capacity hardware, and located on several regions, offering different network latency accesses. In this manner, the grid environments must take into account the heterogeneous nature of the environment.

Work have been conduced to evaluate the features of heterogeneous environments and propose new techniques and models for process scheduling [5, 12–14, 22], load transfer [11, 18] and network modeling [7, 10, 23, 25]. Some of such work, such as from Mello and Senger [11, 13], were developed and applied on grid environments, in order to to evaluate the influences of grid applications on actual networks. This brings up the need to determine when applications should be distributed over grids and at what amplitude. When a distributed application does not involves excessive communication, it is possible to distribute it over networks with non-uniform latencies. Although, when the information exchange is frequent, it is better to allocate nearby computing resources, with low network latencies.

This brings an important subject of how it is possible to determine the nearness of resources and when it is acceptable to distribute applications over distant networks.

Based on these needs this paper used the Round-Trip Time (RTT) methodology [19] – which determines the time elapsed to send and receive a message of a fixed size – to evaluate the behavior of some protocols on different network scales. The network evaluated in this work belong to different network types: LAN – a local area network from the University of São Paulo (USP); MAN – a metropolitan area network infrastructure between USP and Federal University of São Carlos (UFSCar); and WAN – a wide area network between USP and St. Francis Xavier University in Canada. Based on these evaluations this paper aims to define good communication rules that should be applied when implementing grid applications. The transfer rate, another evaluation technique, was adopted to analyze the best message size on each network environments.

This paper is divided into the following sections: 2) related work; 3) network evaluations; 4) experiments; 5) conclusions and references.

2 Related Work

Several researches have been conducted in the area of network analysis and evaluation. Most of network evaluations use theoretical models and simulations, with few practical considerations. The most widely used practical network evaluations are performed using measure-based evaluations, such as the Round-Trip Time [19], combined with more specialized and complex methods like the Gaussian Approximation Allocation [8] and Mean-Value Analysis [1].

Besides the measure-based network evaluations, a network can also be described by analytic models [3, 4, 21]. Among the work related to the analytical network evaluations are the ones by Kherani and Kumar [20], Altman *et al.* [3], Eltetö *et al.* [15], Hockney [19], Iannello *et al.* and Alves and Mello [16].

Kherani and Kumar [20] and Altman *et al.* [3] conduced a behavior evaluation of TCP networks by using analytical models based on two different approaches. In [20], it was introduced a model for network performance prediction based on packet loss probability and network load. The paper analyzes a whole network with all simultaneous connections in order to predict their interactions. The second work [3] used stochastic models for throughput analysis of the random arrival of TCP connections. The authors analyze the network considering the bottleneck probability for each network node. The model is formulated as two equivalent solutions: a fixed point and a nonlinear programming formulation. These work, however, do not consider practical experiments to ensure results.

Eltetö *et al.* [15] analyzed both persistent (ftp-like) and non-persistent (web-like) connections. In this work they introduced a practical analysis framework for TCP networks with three sub-models: the first incorporates a detailed TCP analysis; the second is oriented to network modeling and the third analyzes the dynamics of parallel non-persistent TCP connections. Based on this framework it is possible to estimate the average behavior of network, such as the average queue length and average RTT.

In [19], Hockney proposed a parameterized model to evaluate the elapsed time to transmit messages over networks. On this model one computer, named sender, sends a message of a certain size to another one, named receiver. As the message arrives the receiver replies it. The time consumed on this send/receive operation is called elapsed time or *round-trip time (RTT)*.

Alves and Mello [16] performed a performance analysis on different network protocols, including TCP, UDP, MPI, PVM and GAMMA protocols. The authors evaluated a LAN (local area network) environment using the Hockney [19] and LogP [2] models. In this work, the protocols were evaluated for different message size (up to 20Kb of data). The RTT, GAP, sending and receiving overhead and the network interface latency values were analyzed.

3 Experimental Environment

This paper presents the evaluation of protocols from the TCP family on different network scales. The evaluations were performed using the Linux operating system on kernel versions 2.6 and 2.4 kernel. For the final experiments, the latest 2.6 family of Linux kernels was considered.Only default protocol configurations were adopted, with no specific protocol optimizations nor tweaking. The evaluated protocols were TCP, UDP and Sendfile over LAN, MAN and WAN networks.

The LAN network evaluation was performed on two Athlon XP 2800+ machines with 1Gb of RAM and Broadcom Gigabit Ethernet cards on a separate switched network.

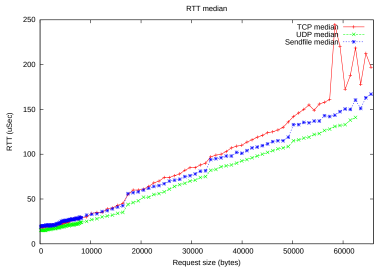
The MAN evaluation was performed using one Athlon XP 2800+ machine with 1Gb of RAM and a Athlon XP 1600+ machine with 256Mb of RAM, on a fast ethernet connection routed by 8 intermediate routers. The physical distance between the machines was approximately $3km$.

Then WAN evaluation was conduced using one Athlon XP 2800+ machine and a Sun UltraSparc cluster composed by five nodes. The Athlon machine was located at ICMC in Brazil, while the Sun machine was located at St. Francis Xavier University in Canada.

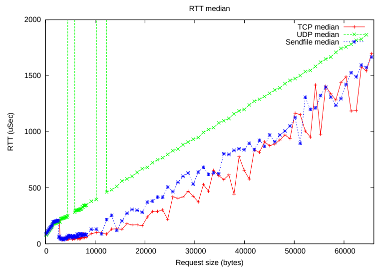
Before the experiments, the expected behaviors of network and directly connected machines were evaluated. The expected network behavior was evaluated using a loopback connection, having all message passing conduced in memory with no physical data transmission. The directly connected network evaluation was performed on one Athlon XP 1600+ machine and one Sempron 2200+ machine, interconnected by a cross-over cable.

4 Network Evaluation

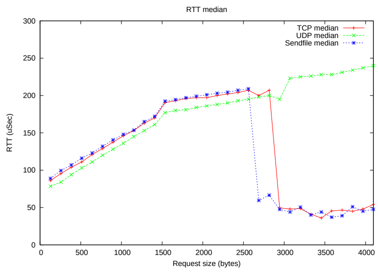
In order to evaluate the network performance for different network classes, a benchmarking application was adopted. This benchmarking application was chosen for the experiments in order to maintain compatible results with previous works [16] Existent performance evaluation applications, such as *iperf* and *netperf*, were not employed as the goal of the experiment was to evaluate the default network and system configurations. The benchmarking application evaluated the time elapsed to transmit and receive equal amount of data, counting the minimum, maximum and medium time taken for



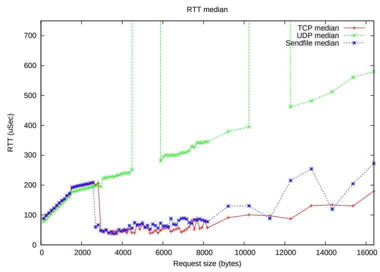
(a) RTT – loopback network evaluation



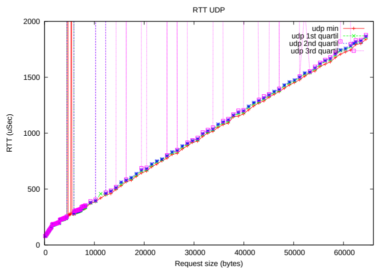
(b) RTT for Gigabit Ethernet LANs



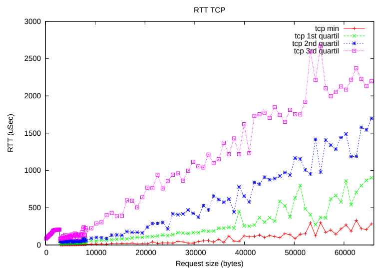
(c) Gigabit LAN – 4KB zoom



(d) Gigabit LAN – 16KB zoom



(e) UDP for LAN networks



(f) TCP for LAN networks

Fig. 1. Evaluation results, part I

this operation among with quartiles distribution [26]. The times distribution was also considered in order to provide a better evaluation of the protocol behaviors. The experiments are shown in the following sections.

4.1 Round-Trip time evaluation

Expected protocol behavior evaluation The expected behavior of the protocols is shown on Fig. 1 (a). This experiment was performed over a loopback connection on a Athlon 2000+ computer running 2.6.10 Linux kernel. In-memory copying was employed for all network operations, having no Internet connection at all. The obtained median results allowed us to confirm that UDP protocol always get the best times, while all other protocols have a constant behavior during the benchmark. The median values were adopted in order to provide a better visualization of the results, without extremely high and low times that could be very distant from other values.

Local area network using gigabit connection The intention of the next experiment was to evaluate the protocol behaviors on a Gigabit LAN. The overall performance for all evaluated protocols is shown on Fig. 1 (b). The results are shown using the RTT median response time. It was possible to observe that UDP takes an extremely long time to round-trip a message of sizes around 5 and 11 Kb. Another interesting observation is that the UDP curve is constant, even on cases where TCP and Sendfile present slow behavior, being the upper transfer delay limit.

Another interesting and unexpected detail is the huge performance improvement for TCP-based protocols on cases where message size exceeds the maximum transmission unit (MTU) size of the Ethernet network. As the amount of data exceeds approximately 2800 bytes (excluding the underline protocol overheads), being two times the MTU size, the TCP-based protocols present a significant performance gain. This behavior can be observed more closely on Fig. 1 (c) (zooming the Fig. 1 (b)) where the TCP protocol improves up to 400% when the message size exceeds 2800 bytes. For the Sendfile protocol, this improvement appears earlier, at approximately 2600 bytes. The UDP protocol, on its turn, keeps its elapsed time constantly growing for all message sizes.

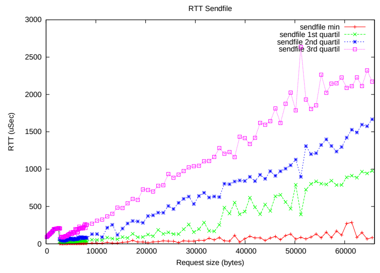
Analyzing further the network behavior for messages of up to 16Kb in size, as shown on Fig. 1 (d), it is possible to observe that the UDP protocol, beside showing a very high latency for some message sizes (messages between 5 and 6Kb in size and messages between 10 and 12Kb in size), manages to obtain the worst results among all evaluated protocols.

On Fig. 1 (e) the first, second and third quartil of RTT for UDP protocol are shown. The quartiles allow a better results visualization, allowing to see the amount of results that happen in up to 25% cases (1st quartil); in up to 50% cases (2nd quartil, or median value) and in up to 75% cases (3rd quartil). Thus, it is possible to see not only the average protocol performance but the overall protocol performance distribution. Based on these quartiles is possible to conclude that, with the exception of some very high latencies for some message, UDP protocol keeps all the time in the same range, with practically no difference between the minimum and maximum times.

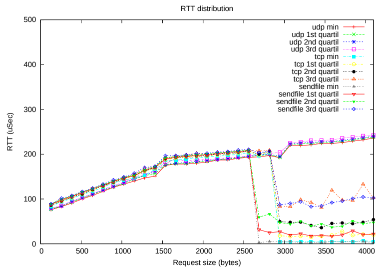
The RTT times distribution for TCP protocol (Fig. 1 (f)) allows to conclude that the variance of the values is very high. However, in most cases it is possible to obtain better results than UDP. As can be seen on Fig. 1 (f), only the results of the third quartil are superior to 2000 ms.

The Sendfile protocol is an optimized implementation of the TCP protocol, using in-kernel zero-copy sockets thus requiring less memory copy operations. The performance of this protocol, however, is very similar to the traditional TCP protocol, as shown on Fig. 2 (a), only getting better results for larger messages.

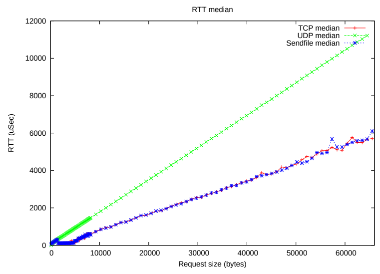
Finally, Fig. 2 (b) shows the behavior of all evaluated protocols for small messages on LAN networks. As can be seen, the behavior of all evaluated protocols is similar until



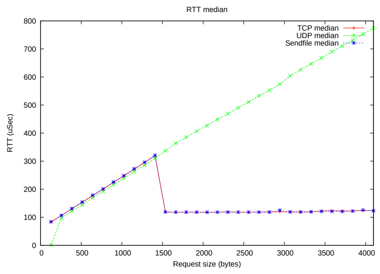
(a) Sendfile for LAN networks



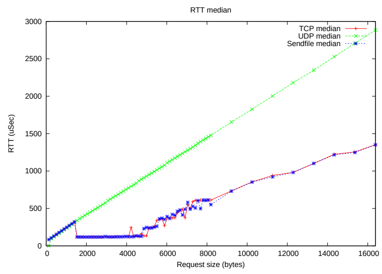
(b) Overall LAN evaluation



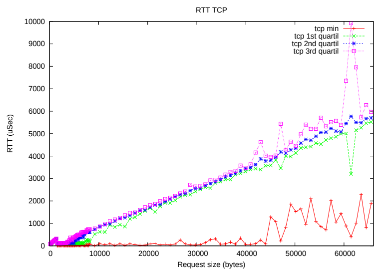
(c) RTT for Fast-Ethernet LAN



(d) RTT for Fast-Ethernet LAN, 4KB zoom



(e) RTT for Fast-Ethernet, 16KB zoom



(f) TCP for Fast-Ethernet networks

Fig. 2. Evaluation results, part II

the message size exceeds the MTU of the network. After that, each protocol presents a different behavior.

Local area network - fast Ethernet cross-over connection The next experiment was performed in order to evaluate a direct cross-over connection over a Fast-Ethernet network. No switches were used in this benchmark in order to measure the direct connection environment over a Fast-Ethernet network. The overall performance of the protocols is shown on Fig. 2 (c). By observing the figure it is possible to conclude that the TCP-based protocols have a significant performance boost when the message size exceeds 1400 bytes. The UDP protocol keeps the same stable behavior for all message sizes.

Taking a closer look at the round-trip times, as shown on the Fig. 2 (d), it is clear that the performance gain for TCP-based protocols occurs on messages that have more than 1400 bytes in size. Also, as can be seen on the Fig. 2 (e), all evaluated protocols are able to sustain a continuous behavior for small and medium messages, what does not happens on Gigabit Ethernet switched connection network, possible due to the network switch usage.

The Fig. 2 (f) presents the minimum, first, second and third quartiles of RTT times for TCP protocol for messages on a larger scale. Analyzing this figure, it is possible to note that the TCP protocol presents a sustained behavior up to messages with 40Kb of data. Exceeding that value, the results present a large variation and instability. The Sendfile protocol offers a similar behavior as shown on Fig. 3 (a).

The UDP protocol, on its turn, shows exactly the same behavior for all messages, as shown on Fig. 3 (b).

Finally, it was observed that the behavior of this experiment was very close to the expected results, with the exception of TCP-based protocols being actually much faster for larger messages than UDP.

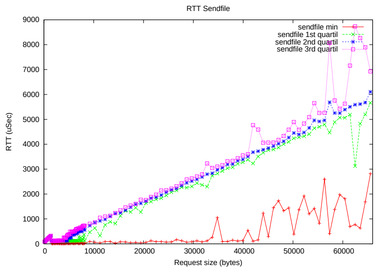
Metropolitan-area network This experiment evaluated the network RTT times for a wider network. The connection was established from USP to UFSCar, along a three-kilometer 100 Mbit fiber link.

The overall results for this experiment are shown on Fig. 3 (c). As can be observed on the figure, the overall behavior of the protocols remains the same as for the LAN network benchmark. However, it was not possible to benchmark the UDP protocol for all message sizes, as when the data size increases to more than 13Kb, the protocol passes to lose too much messages, making impossible the benchmarking without retransmission of the lost messages (which was not the goal of the experiment).

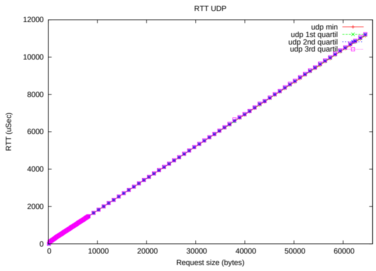
Analyzing the behavior of the protocols for small message sizes up to 4Kb, as shown on Fig. 3 (d), it is possible to ensure that the behavior of the protocols on a MAN network is similar to its LAN equivalent (with the difference that RTT times are much higher than their LAN counterparts). Another interesting detail is that the TCP-based protocols obtain their performance boost on a different message size – while on the LAN benchmarks this speed up occurred at $2*MTU$ message size, on the MAN benchmark the TCP protocols got faster for messages of around 1800 bytes. As on the LAN benchmark, the UDP protocol (on Fig. 3 (e)) keeps constant behavior for all request sizes.

While the TCP protocol presented better average RTT times, it also offered much higher times for the messages larger than 20Kb, showing an unpredictable behavior for large messages, as shown on Fig. 3 (f).

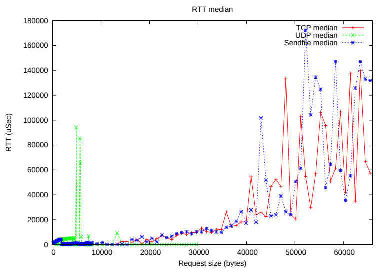
Taking a closer look at the TCP protocol for the MAN networks, it was observed that the times distribution is constant for messages under 4Kb of data, improving drastically the performance for messages of around 1800 bytes of data, as shown on Fig. 4 (a). For messages larger than 4Kb of data, the behavior of the protocols starts to vary too much



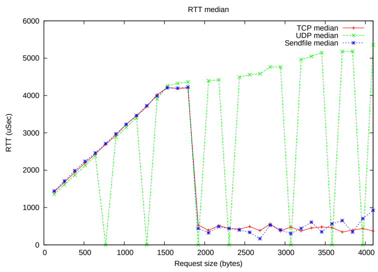
(a) Sendfile for Fast-Ethernet networks



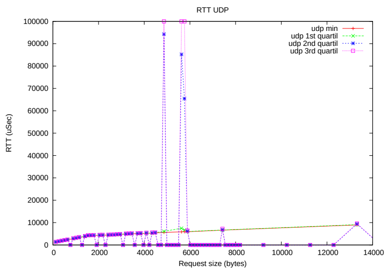
(b) UDP for Fast-Ethernet networks



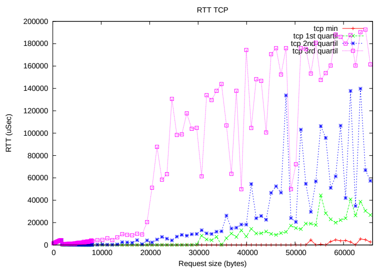
(c) RTT for MAN networks



(d) RTT for MAN networks, 4KB zoom



(e) UDP for MAN networks



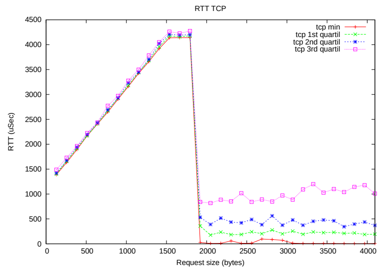
(f) TCP for MAN networks

Fig. 3. Evaluation results, part III

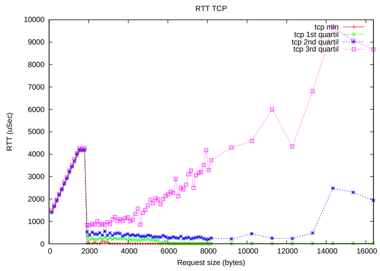
and gets unpredictable, as shown on Fig. 4 (b). Finally, the Sendfile protocol is still very similar to the TCP for the MAN networks, as shown on Fig. 4 (c).

Wide Area Networks This benchmark was performed in order to evaluate an wide area network between Brazil and Canada.

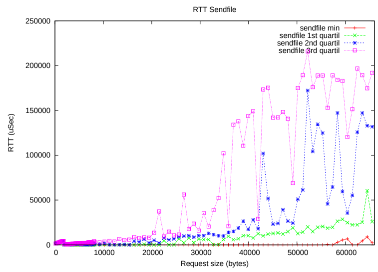
The overall results for the RTT are depicted on Fig. 4 (d). As can be observed, the UDP protocol was unable to complete the data transfer without packet losses for any message size, thus its results were discarded on this experiment. One interesting detail



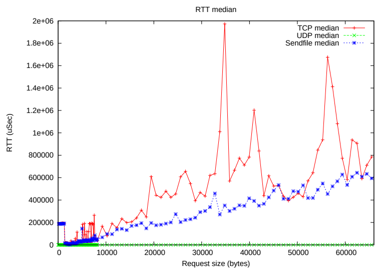
(a) TCP for MAN networks, 4KB zoom



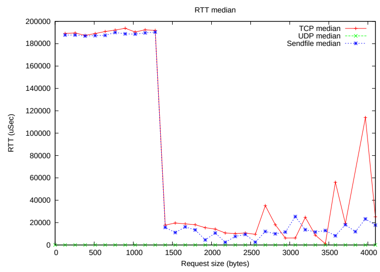
(b) TCP for MAN networks, 16KB zoom



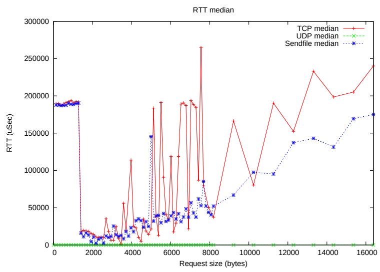
(c) Sendfile for MAN Networks



(d) RTT evaluation for WAN networks



(e) RTT for WAN networks, 4KB zoom



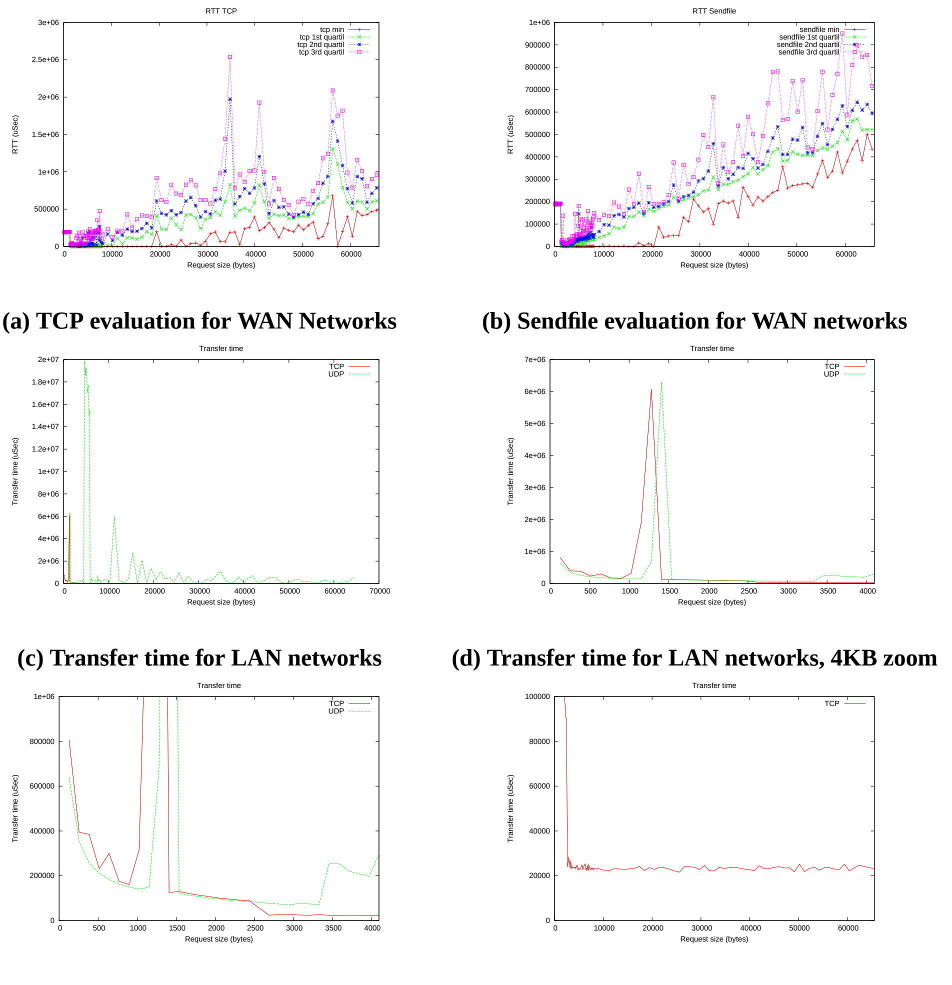
(f) RTT for WAN networks, 16KB zoom

Fig. 4. Evaluation results, part IV

is the unexpected performance of the Sendfile operation, which managed to get much better than the plain TCP protocol due to the high latency of the network.

Taking a closer look on Fig. 4 (e), it is possible to see that TCP-based protocols maintain the same behavior, once again showing a performance gain for the message size of approximately 1400 bytes.

Evaluating the protocols for messages smaller than 16Kb, as shown on Fig. 4 (f), it is possible to observe that, while the TCP protocol offers an unpredictable behavior, the Sendfile manages to keep a close-to-linear growth for increasing message size, thus offering a much better overall performance than the original TCP protocol.



(a) TCP evaluation for WAN Networks **(b) Sendfile evaluation for WAN networks**

(c) Transfer time for LAN networks **(d) Transfer time for LAN networks, 4KB zoom**

(e) Transfer time, 4KB zoom without peaks **(f) Transfer time for LAN networks, TCP protocol**

Fig. 5. Evaluation results, part V

The unpredictable behavior of the TCP protocol is illustrated on the Fig. 5 (a). Comparing the behavior of the TCP protocol to its Sendfile counterpart, as shown on Fig. 5 (b), it is possible to see that the Sendfile obtains better and more predictable results for wide area networks.

4.2 Transfer time

Having estimated the network configuration using the RTT experiments, the next step was to determine how these values influence on the network utilization. Thus, another series of experiments was performed. The next experiment consisted in a transfer of a

1Mb file in both direction, varying the message buffer size and measuring the overall transfer time, as illustrated by algorithm on Table 1.

Table 1. Transfer rate evaluation algorithm

$size := 1024 \times 1000;$	**// The total transfer size.**
$msg\_size :=< REQUESTSIZE >;$	**// Divide into smaller messages.**
$sent\_size := 0;$	**// Data already sent.**
$first\_time :=$ get_first_time();	**// Get the initial time.**
while($sent\_size < size$) {	**// Do the message transfer.**
send(buffer[msg_size]);	
receive(buffer[msg_size]);	
$sent\_size := sent\_size + msg\_size;$ }	
$last\_time :=$ get_last_time();	**// Get last time.**
$total\_time := last\_time - first\_time;$	**// Generate the elapsed time.**

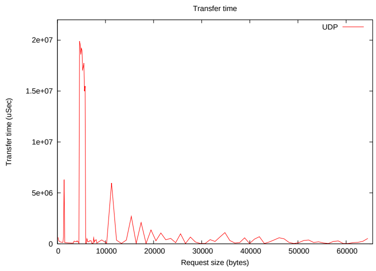
The total time elapsed for a fixed size data transfer can be used for effective transfer rate calculation. As we are using 1MB-size file, the resulting time represents the time necessary to transfer 1MB of data; the transfer rate can be calculated as $rate = 1Mb/time$.

LAN networks As was observed in the RTT experiments, for LAN networks the network is used in the same manner for any message size with UDP protocol, and obtains the most effective behavior with messages larger than 2800 bytes for TCP protocol.

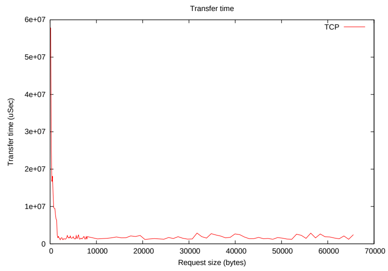
Fig. 5 (c) shows the global picture for the experiment. As can be observed, the UDP-based data transfer takes a very large amount of time for the data transfer, making the time elapsed by TCP transfer look negligible.

A closer look at the picture, as shown on Fig. 5 (d), shows that a very high elapsed time values takes place for both TCP and UDP protocols for messages with approximately 1Kb in size. A picture without the peak values is shown on Fig. 5 (e). As can be observed, the UDP protocol is actually faster than the TCP for small message size, but when the message size equals to 1500 bytes both protocol show equal speed, and after 2500 bytes of data the TCP protocol gets faster than UDP. Further analyzing the TCP protocol, as shown on Fig. 5 (f), it is possible to see that the elapsed time for data transfer is similar for any message size greater than 2500 bytes. The UDP protocol, as shown on Fig. 6 (a), shows inconstant behavior for all message sizes, with extremely high transfer times in some cases.

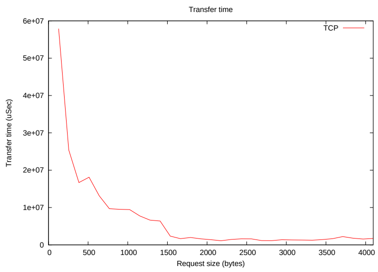
MAN networks In cases of MAN and WAN networks and the UDP protocol it was not possible to complete the 1Mb data transfer without any packet loss, so just the results of the TCP-based protocols are showed. The expected behavior of the TCP protocol is to show a performance boost for messages larger than 1800 bytes in size. The behavior of the TCP protocol is shown on Fig. 6 (b). As expected, the TCP protocol obtains the best performance for messages larger than 1500 bytes. This behavior is better shown on Fig. 6 (c).



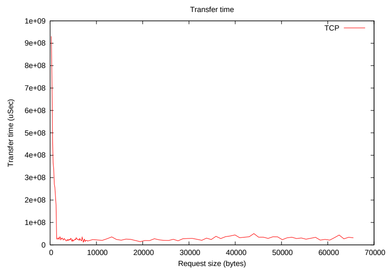
(a) Transfer time for LAN Networks, UDP protocol



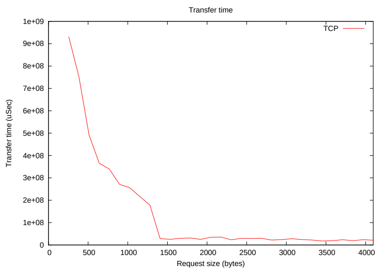
(b) Transfer time for MAN networks



(c) Transfer time for MAN networks, 4KB zoom



(d) Transfer time for WAN networks



(e) Transfer time for WAN networks, 4KB zoom

Message size	**LAN**	**MAN**	**WAN**
0Kb – MTU	UDP	UDP	SF
MTU – 4Kb	SF	SF	SF
4Kb – 16Kb	TCP	TCP	SF
16Kb – 32Kb	TCP	SF	SF
32Kb – 64Kb	TCP	TCP/SF	SF

(f) Summary: Best RTT results

Fig. 6. Evaluation results, part VI

WAN networks Finally, for the WAN networks the expected behavior was to experience a TCP speed-up for messages larger than 1500 bytes. The real behavior for the WAN networks is shown on Fig. 6 (d). As can be observed, the TCP protocol gets the best performance for messages larger than 1400 bytes in size, as expected. The behavior of the TCP protocol is better showed on Fig. 6 (e).

5 Conclusions

This paper performed an evaluation of network protocols for different network configurations, including LAN, MAN and WAN networks. The results were obtained for the Linux OS on its default configuration, and describe both 2.4 and 2.6 Linux kernel families. This paper, however, focused mainly on the currently stable Linux 2.6 family of kernels.

The Fig. 6 (f) resumes the results for the evaluated protocols and network environments.

Concluding the network evaluation of both Round-Time Trip times and data transfer times, it is possible to ensure that for LAN networks, the UDP protocol shows a better performance for messages smaller than 2800 bytes. When exceeding this message size, the UDP protocol does not offers acceptable performance.

TCP and Sendfile protocols show best performance when the message size exceeds $2 * networkMTU$ bytes for Round-Trip time evaluations. These protocols speedup happen at 2800 bytes for LAN networks, 1800 bytes for MAN networks and 1400 bytes for WAN networks.

UDP protocol is not recommended to be used on MAN or WAN networks due to a slower behavior and excessive packet loss. TCP protocol, on its turn, shows best results when used on LAN networks, being surpassed by the Sendfile for MAN and WAN networks. In all cases TCP-based protocols manage to obtain better transfer rate than the UDP protocol – for example, it is much faster to transfer 4Kb of data using TCP protocol than 1Kb of data using UDP.

The Sendfile protocol offers a better performance due to zero-copy sockets usage and in-kernel operations. This is observed mainly on WAN networks, due to high network latency values. This is very important for Grid environments which are based on long distance connections. Also, Sendfile shows the best performance for messages between approximately 2.8Kb and 4Kb in size in all cases.

References

1. I. F. Akyildiz. Mean value analysis for blocking queueing networks. *IEEE Transactions on Software Engineering*, 14(4):418–428, 1988.
2. A. Alexandrov, M. F. Ionescu, K. E. Schauser, and C. Scheiman. LogGP: Incorporating long messages into the LogP model for parallel computation. *Journal of Parallel and Distributed Computing*, 44(1):71–79, 1997.
3. E. Altman, K. Avrachenkov, and C. Barakat. TCP network calculus: the case of large delay-bandwidth product, 2002.
4. E. Altman, K. Avrachenkov, C. Barakat, and K. Chandrayana. Complementarity formulation for TCP/IP networks: uniqueness of solution and relation with utility optimization.
5. Y. Amir. An opportunity cost approach for job assignment in a scalable computing cluster. *IEEE Transactions on Parallel and Distributed Systems*, 11(7):760–768, July 2000.
6. Apple Computer, Inc. X-grid project. http://www.apple.com/acg/xgrid/.
7. S. Chodnekar, V. Srinivasan, A. S. Vaidya, A. Sivasubramaniam, and C. R. Das. Towards a communication characterization methodology for parallel applications. In *Proceedings of the 3rd IEEE Symposium on High-Performance Computer Architecture (HPCA-97)*, page 310, 1997.

8. H.-W. Chu, D. H. K. Tsang, and T. Yang. Bandwidth allocation for VBR video traffic in ATM networks. In *ICCCN*, page 612, 1995.
9. P. Costa, S. D. Zorzo, and H. C. Guardia. Progrid: A proxy-based architecture for grid operation and management. In *Proceedings of International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD03)*, 2003.
10. D. E. Culler, R. M. Karp, D. A. Patterson, A. Sahay, K. E. Schauser, E. Santos, R. Subramonian, and T. von Eicken. LogP: Towards a realistic model of parallel computation. In *Principles Practice of Parallel Programming*, pages 1–12, 1993.
11. R. F. de Mello and L. J. Senger. A new migration model based on the evaluation of processes load and lifetime on heterogeneous computing environments. In *International Symposium on Computer Architecture and High Performance Computing (SBAC-PAD04)*, page 6, 2004.
12. R. F. de Mello and L. J. Senger. A new slowdown model for heterogeneous multicomputing environments. *submited*, pages 1–23, 2005.
13. R. F. de Mello and L. J. Senger. A routing load balancing policy for grid computing environments. *submited*, pages 1–16, 2005.
14. R. F. de Mello et. al. Analysis on the significant information to update the tables on occupation of resources by using a peer-to-peer protocol. In *16th Annual International Symposium on High Performance Computing Systems and Applications (HPCS-02)*, Moncton, New-Brunswick, Canada, June 2002.
15. T. Eltetö, P. Vaderna, and S. Molnar. Performance analysis of tcp networks loaded by web traffic.
16. E. A. Ferreira and R. F. de Mello. Avaliação de desempenho de protocolos e bibliotecas de comunicação entre processos. In *3rd International Information and Telecommunication Technologies Symposium*, page 8, 2004.
17. I. Foster. Interneting computing and the emerging grid. `www.nature.com/nature/webmatters/grid/grid.html`, 2000.
18. M. Harchol-Balter and A. B. Downey. Exploiting Process Lifetimes Distributions for Dynamic Load Balancing. *ACM Transactions on Computer Systems*, 15(3):253–285, August 1997.
19. R. W. Hockney. The science of computer benchmarking. In *SIAM*, 1996.
20. A. Kherani and A. Kumar. Stochastic models for throughput analysis of randomly arriving elastic ows in the internet, 2002.
21. J. Padhye, V. Firoiu, D. Towsley, and J. Krusoe. Modeling TCP throughput: A simple model and its empirical validation. *Proceedings of the ACM SIGCOMM '98 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 303–314, 1998.
22. L. J. Senger, R. F. de Mello, M. J. Santana, and R. H. C. Santana. Improving scheduling decisions by using knowledge about parallelapplications resource usage. *submited*, pages 1–12, 2005.
23. A. Sivasubramaniam. Execution-driven simulators for parallel systems design. In *Winter Simulation Conference*, pages 1021–1028, 1997.
24. The Globus Project. http://www.globus.org.
25. J. S. Vetter and F. Mueller. Communication characteristics of large-scale scientific applications for contemporary cluster architectures. *Journal of Parallel and Distributed Computing*, 63(9):853–865, 2003.
26. W.C.Shefler. *Statistics: Concepts and Applications*. The Benjamin/Cummings, 1988.