Time for TOE

The Benefits of 10 Gbps TCP Offload

A Chelsio Communications White Paper

Abstract

TCP offload has been a controversial technology, partly because past implementations showed little or no performance benefit, and therefore no cost advantages. With the recent advent of 10 Gbps Ethernet, and the resulting CPU / network performance gap, TCP offload is worth revisiting. This paper shows that a properly designed TCP offload engine can significantly outperform non-offloaded NICs, in both latency and throughput, bringing 10 Gbps Ethernet up to specialized interconnect performance levels. In addition to performance improvements, the benefits of TCP offload extend to decreasing cost of operation by significantly reducing CPU usage, as well as addressing the memory subsystem bottleneck by allowing direct memory access on both send and receive. Furthermore, with the reliable transport layer implemented in hardware, it becomes possible to offload expensive byte touching operations and protocol functionality at higher layers, further increasing the value of the technology.

Introduction

The Internet, in all its facets, has become a central part of our daily life. Two infrastructure technologies define the Internet: the TCP/IP standards, which specify the protocols used end-to-end by Internet hosts, and Ethernet, which provides local connectivity. Throughout the 30 years of its existence, Ethernet has with its successive 10x speed increases, progressively displaced every other local networking technology in the speed class it moved into. The low cost of Gigabit Ethernet has now enabled its widespread deployment, and the cost of 10 Gbps Ethernet is falling dramatically, repeating the price drop typical of earlier Ethernet versions.

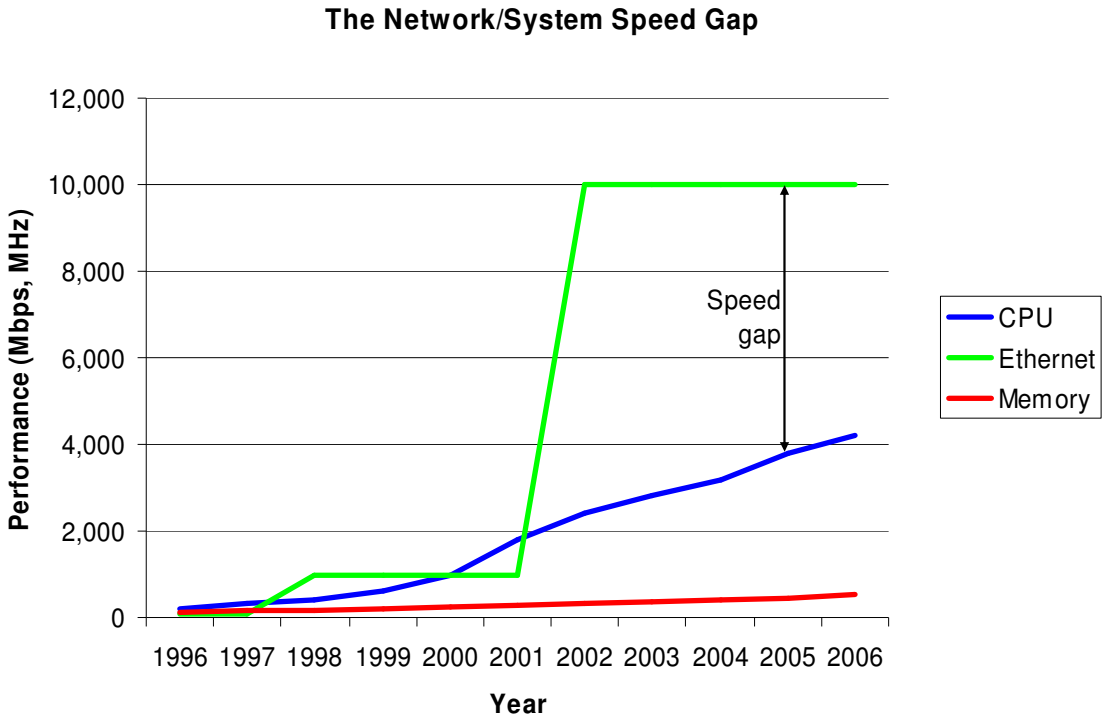

Figure 1: The network/system speed gap is increasing and persisting longer

Gigabit Ethernet desktop and laptop connectivity is now widespread, and this leads to the need for faster links at the server side. However, it is well recognized that the network processing capabilities of today's high end processors fall short at 10 Gbps network speed. In fact, the well known rule of thumb for processing need of a GHz per Gbps shows that the speed difference is now at an unprecedented level of three fold. This observation is confirmed by

experimental measurements which show that a top of the line CPU is fully pegged doing TCP/IP processing, at 3 to 4 Gbps of network bandwidth, leaving no cycles for useful applications. The performance gap is not expected to be bridged in the near future, particularly so as the bottleneck is expected to shift from the CPU to *other components*, mainly the memory and I/O subsystems, which lag behind the CPU speed curve, as illustrated in Figure 1. Memory speed has been increasing at a much slower pace than CPUs, and it is expected to become the main performance limitation in the future. In addition, future Ethernet speed increases are in the works, with 100 Gbps estimated to be available before the end of the decade, further widening the gap. There is therefore a strong motivation to move network processing to "smart" network adapters, or protocol offload engines (or TCP Offload Engines, in the context of Internet protocols). TOEs have the potential to address the memory system bottleneck issue, and their specialized nature allows better performance scaling than commodity processors.

Protocol offload is not a new idea. Indeed, every one of the Ethernet speed increases has been accompanied by a surge of interest in protocol offload: the move to 100 Mbps from 10 Mbps in 1993, to Gigabit in 1998 and to 10 Gbps in 2002. However, significant performance barriers in server performance have now appeared for the first time, as confirmed by the top CPU vendors [INTEL1,INTEL2]. Alternative ways are therefore needed for enabling the high performance brought by the latest Ethernet technology, which promises to displace the last few esoteric technologies still in use in special purpose storage and computing networks. The incentive is the resulting network convergence which is expected to bring in large gains in management and usage simplicity, as well as the connectivity benefits of a unified infrastructure, and last but not least the Ethernet economy of scale that will commoditize 10 Gbps Ethernet pricing.

In the following section, we first discuss the alternative approaches suggested for dealing with the performance problem at hand, through limited assist from "dumb" network interface cards (NICs). These include the support for large non-standard "jumbo" frames, TCP segmentation offload and large receive offload. We then move on to re-examine the various arguments given against the TOE approach in light of Chelsio Communications, Inc.'s Terminator engine characteristics, and show how these arguments fail to hold ground. In the process, we describe how full TCP offload as implemented in Chelsio's Terminator improves system performance beyond the expected drop in network processing loads. We finally provide tangible evidence of the promises fulfilled by Chelsio's TOE, as independently measured by some of the World's top laboratories and institutions.

"Dumb" NIC Approaches

Throughout the past few years, and particularly since the standardization of Gigabit Ethernet, a number of hardware assist functions have been proposed and implemented in "dumb" network adapters. Based on the findings of early studies of TCP/IP overhead such as [CLARK89], the first candidate for such support was the checksum operation. This feature is now typically available in common network adapters. In addition, more controversial support mechanisms have appeared in the same period, which are discussed next.

Jumbo Frames

The first of these proposals is the use of large "jumbo" frames in the network. It should be noted that the Ethernet standard limits the useful transfer unit (frame payload) to 1,500 bytes, which at 10 Gbps results in a packet rate of about 1 million per second. By increasing the size limit, it is possible to decrease the packet per second processing load on the servers, and therefore increase the wire performance. What may seem to be a good idea at first sight turns out to be impractical for a large number of reasons, and therefore not a viable solution to the performance problem. Indeed, jumbo frames are not standardized, not recognized by previously deployed equipment, and therefore not supported on most of the links in the Internet. In fact, today's switch-on-a-chip solutions, which provide very low latency on par

with specialized inter-connects are known to perform worse with jumbo frames than standard ones due to limited on-chip buffering. In addition, the benefits attributed to large frames are confined to one application, which is bulk data transfer. The performance of latency sensitive applications, on the other hand, will not benefit from large frames, and may actually degrade due to weaker pipelining and increased transmission delays. This problem is particularly relevant when considering a converged network shared by all application types. For these reasons, it is commonly considered as a fact that jumbo frames are more of a benchmarking tool than a real option for actual applications. Considering that they were introduced almost a decade ago, this explains their lack of deployment to this day.

TCP Segmentation Offload

An alternative to using jumbo frames is TCP segmentation offload (TSO), also known as large send offload (LSO). This dumb NIC feature allows software to pass large TCP packets to the network interface card, where they get segmented into standard sized frames, relieving the host CPU of that burden. In addition to the fact that it only helps in large transfers, this idea is marred by two issues, as discussed below.

The first issue is that the TCP protocol has been designed for and tested over two decades using standard Ethernet frames. In contrast, TSO exposes an artificially large segment size to the TCP layer, resulting in increased burstiness and resulting packet loss in switches, particularly in the desirable low-latency switch-on-a-chip architectures. Packet loss results in severe performance degradation for software TCP at high speeds. In addition, there are poorly understood interactions between TSO and the various TCP transmission schemes such as Nagle, Silly Window Syndrome Avoidance, delayed ACK and congestion control and which may potentially result in poor performance outside of the dedicated link environment where this scheme is typically tested. For these reasons, stacks which have enabled support for TSO have required modifications to address the various performance and correctness problems which have been encountered. Recent attempts at limiting the interference of TSO with TCP's mechanisms in Linux have actually resulted in decreased performance. It appears that unless significant research is done on the unknown side effects of TSO, it is not advisable to deploy it widely.

The second issue resides in the fact that the unnaturally large packet size results in poor multiplexing between different connections and decreased performance. Therefore, this may limit the applicability of TSO to single or low connection applications.

Large Receive Offload

A similar change has been proposed at the receiving end of TCP, whereby the network adapter merges incoming packets belonging to the same connection (among other conditions) into a larger one. When the various necessary conditions apply (i.e. the number of active connections is small and the application is bulk transfer), this scheme results in fewer, but larger, packets being handled by the TCP layer. Similar concerns can be raised here as to the impact of such abnormally large packets on the receiving mechanisms of TCP, and delayed acknowledgments (ACKs) in particular. Indeed the receiver is required to generate a sufficient number of ACKs to keep the "ACK clock" running. This clock is an essential aspect of TCP's congestion control concept, as it regulates the rate of a sender to the bottleneck speed. A broken ACK clock due to aggregation of packets results in burstiness and packet loss, similarly to the TSO approach. The problem is aggravated by the fact that ACKs are critical for fast recovery from packet loss. Again, the potential impact of reduced ACK rates on TCP's performance limits the applicability of LRO to controlled environments. Furthermore, the LRO scheme is limited to low connection counts, and provides no benefits otherwise. Finally, the per-byte costs due to copying data are not reduced, and remain as a main source of performance bottleneck.

In summary, the TSO/LSO and LRO mechanisms attempt to offload some of the work from the host CPU. They are limited in both the application of interest and the environment in which they can be used. Furthermore, they both interfere with and may ***break the vital TCP mechanisms*** which have kept the network from suffering congestion collapse. With the availability of Chelsio's proven TOE, there is no need to resort to unnatural modifications to TCP's critical mechanisms and be limited to the poor man's offload which LSO and LRO are. Indeed, given the prevalence of TCP (it is known to carry 95% of all traffic in the Internet), it must be stressed that widespread deployment of such changes may have unforeseen and potentially harmful impact on the stability of the network.

TOE: For and Against

Having considered the various proposed or available dumb NIC schemes, discussed their benefits and exposed their limitations, it is time to do the same for offload engines. We go through the large number of arguments for and against them and discuss how Chelsio's Terminator engine compares to the typical pre-conceived image of an offload engine. In this discussion, we'll follow the order of the arguments given in the well known paper [MOGUL04], which provides a compilation of all the known arguments against TOE, but also makes the case for TOE.

Basic Concerns

We start by the basic concerns expressed about TCP offload, and which if true, question the basic usefulness of the technology.

TCP Processing is Cheap

The first argument brought up against TOE is that TCP processing is not expensive, as found in a 1989 study [CLARK89]. In other words, a TOE which only removes TCP processing overhead is not very useful. There are many reasons this argument is actually weaker than it appears. First, the conclusion above is taken out of context. The referenced study does not conclude that network processing is cheap, but instead that most of the overhead is not in the TCP fast path but rather in the associated functions such as buffer management, timer management, checksums and copies. A TCP offload engine which only speeds up the fast path clearly cannot help much, as evidenced by some commercial TOE products. In summary, this argument does place the onus on the TOE software and hardware designers to address the real problems, but does not preclude significant performance benefits to a well designed solution.

Second, the study focused on the main TCP path and did not consider the costs of the full protocol, such as connection establishment, handling retransmissions, out-of-order data and other exceptions. There is a common misconception that such occurrences are rare in practice, particularly in a controlled LAN environment. In reality, a high speed 10 Gbps network sees an increased likelihood of packet drop due to link mismatches and equipment limitations compared to slower networks. It is important to note here that, at 10 Gbps speed, what would appear as "short" retransmission delay (e.g., a few milliseconds) can cause severe throughput loss. Such delay is readily incurred when packets are sent over PCI links and processed in software, even with TCP's Fast Retransmit mechanism. Furthermore, packet drop also results in re-ordering of packets at the receiving end. Therefore, a high performance TOE needs to support packet re-assembly to avoid further delay as software re-assembles the out-of-order data. With a full hardware TCP implementation on both ends of a connection, retransmission and recovery happen orders of magnitude faster than possible with a software stack, resulting in line rate performance even in lossy networks. These aspects allow the Chelsio Terminator to provide high throughput and low latency through switch-on-a-chip architectures, hiding the limitations in the buffering resources available in the switch, and turning Ethernet into an infrastructure which competes with exotic inter-connects, such as Infiniband and Myrinet (see [OSU05] for experimental results).

Finally, system bottlenecks have changed since the study was performed (more than 15 years ago). In particular, at 10 Gbps rates, the limits of interrupt handling and process scheduling capabilities are tested to the extreme.

The Chelsio Approach
Chelsio's Terminator engine implements a fully IETF RFC standards compliant TCP/IP stack, thereby bypassing all software processing between the network interface and the application layer, including connection setup and teardown, timer management, retransmissions (at microsecond level resolution) and other exception handling. In addition, it provides a large packet interface to the application in both the TX and RX directions, without breaking the dynamics of TCP on the wire or by resorting to non-standard jumbo frames. The engine is shown to provide 4 to 7 times the performance per CPU of a software stack, even without eliminating data copies. With the latter implemented, the CPU utilization drops by another factor of 5.

Commodity CPUs Scale Faster than TOE

The second argument given against TOE is that host CPU speeds benefit from Moore's law, which allows them to improve beyond the capability of offload adapters, quickly rendering the latter obsolete. This argument is relevant to some approaches for protocol offload, namely using embedded CPU(s) on the adapter board to perform the protocol processing (putting an identical CPU to the host's does not make economic or practical sense nowadays). Clearly, performing the work of one of today's top end system processors requires multiple embedded CPUs, which typically run an order of magnitude slower. Such a configuration therefore suffers from scalability limitations which affect both low connection count and high connection count performance, due to pipelining and caching problems.

The Chelsio Approach
In contrast, the Terminator architecture is a specialized data flow pipelined VLIW processor, and has been demonstrated to give line rate performance for one up to more than 14 thousand connections. Terminator's architecture provides 10 Gbps line rate at 125 MHz, which is well within today's ASIC technology capabilities, and is therefore readily scalable to faster speeds, such as 2x10 Gbps operation. Commodity CPUs are not expected to be capable of such speeds in the foreseeable future.

Other Concerns

Further technical arguments against TOE in the list of [MOGUL04] are somewhat less fundamental than the first two, but the following paragraphs address each of these items in turn.

Complex interfaces to TOEs, a problem found in some offload designs, where the TOE API is complicated enough to negate the benefits of protocol offload. This is clearly an implementation specific concern. The Chelsio Protocol Language, CPL, provide a lightweight interface to the TOE which does not suffer from this problem.

Suboptimal buffer management, due to lack of support for upper layer protocol (ULP) parsing and separation of application level headers from payload. Chelsio's Terminator is capable of parsing and splitting the headers for popular protocols such as iSCSI and iWarp, and computing the header and payload CRC separately. It also implements generic header splitting to be used for other ULPs.

Connection management overheard, due to sharing and coordinating connection state between the adapter and the host OS is not a concern for Chelsio's full offload approach. Connection state management overhead is particularly problematic for partial offload solutions, which are the source of such concerns. In contrast, whenever Terminator is handling a connection, there is no need for connection state coordination with the host stack.

Resource management concerns, in the sharing of TCP buffers between the stack and the adapter. Again these concerns are typical of partial offload approaches, which require such sharing of state and payload, but are not applicable to Terminator's full offload approach.

Event management limitations of transaction (e.g., Web) servers, which are not perceived to be addressed by offload. This concern is not supported by evidence, and is in fact contradicted by measurements made with Chelsio's TOE, which show up to a 10 fold increase in achieved transaction rates for Web servers (see performance results below).

Scalability limitations, due to the real estate constraints in an off-board protocol implementation. Chelsio's offload solution addresses these concerns by providing protocol offload as a stand alone component, which falls back to software processing when hardware limits are reached. However, experimental evidence in tests with large numbers of connections shows that host software limits are significantly lower than hardware limits on a Chelsio 10 Gbps adapter. The figure below shows Chelsio's software architecture, which illustrates how the TOE stack sits fully astride the software stack, and does not interfere nor modify it and provides a very thin layer between applications and the offload adapter.

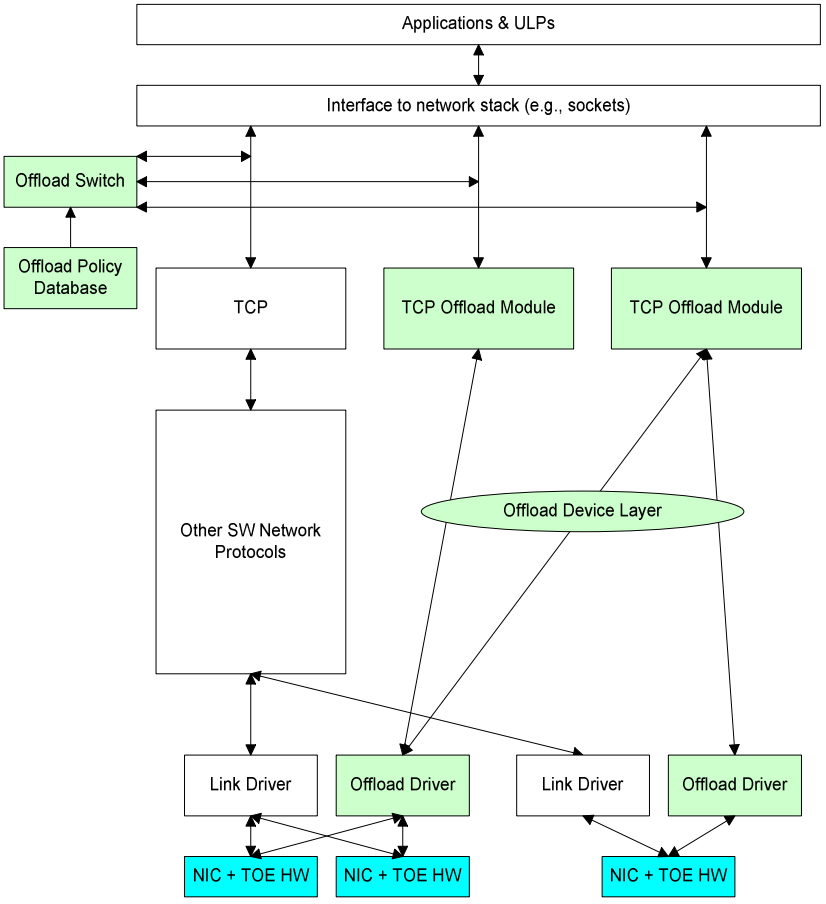

Figure 2 Chelsio Software Architecture

TOE vs. TOE

It is important to note here that the answers to the arguments against TOE are associated with a particular TOE architecture. Not all TOEs provide the same benefits and have the same characteristics, and therefore the discussion above may not always result in the same

assessment regarding the benefits of TCP offload. The question becomes then, what makes a TOE a good TOE? The answer lies in two aspects.

First, the architecture needs to provide the required performance scalability. A good TOE provides both line rate throughput and low latency, along all axis of interest: small and large packets, one to thousands of connections and microsecond to second round trip times. Chelsio's Terminator architecture was designed from the grounds up with these objectives in mind. For more details on the Terminator architecture and how it compares to other approaches, refer to the Chelsio whitepaper available at http://www.chelsio.com.

The second aspect is the building of value add functionality on top of the basic TCP processing offload. As discussed earlier, offloading the processing of the fast path does not provide tangible benefits. It is the handling of the full TCP protocol in hardware that allows Ethernet to rise to specialized interconnect performance levels, and the support of higher layer functionality that truly exposes the potentials of the technology. The following two sections expand on these ideas.

TOE's Value Proposition

This section highlights the many benefits of using a TOE rather than a regular network interface card. These benefits can be categorized into two classes. The first includes the performance advantages of TOE, and its capacity as a **technology and application enabler**. The second focuses on the decreased total cost of ownership of an offload equipped network system compared to a non-offloaded one. A simple comparison shows that TOE provides a considerable return on investment.

The combination of the two classes is analogous to the dynamics which turned video adapters into essential components in computer systems, freeing the CPU to from a specialized function to handle the application load, i.e. the function it is best suited for.

Performance Gains

Having considered the various typical arguments against TOE, and argued why Chelsio's approach avoids the pitfalls at the root of these arguments, it is important to discuss the actual performance impact of offloading the TCP/IP stack. Chelsio's Terminator engine is the first and currently unique 10 Gbps offload engine. Therefore, the experimental results presented in this paper shed a completely new light on the dynamics of offload at high speed, and the benefits which offload provides in such environments.

Before we present sample measurement results in the following section, it is worth repeating that effective offload does not limit itself to processing TCP's fast path. Rather, it is the ability of a TOE to perform the full transport layer functionality which is essential to obtaining tangible benefits. The important aspect of the transport layer is it being the **process-to-process** layer, i.e. the data passed to the TOE comes straight from the application process, and the data delivered by the TOE is ready to be passed to the application process. In contrast, lower layers only provide unreliable delivery functionality, which limits the usability of the data at these levels. This opens the way for very powerful extensions to pure protocol offload, including:

1. **Direct Data Placement** (DDP), which addresses the memory subsystem bottleneck problem on receive,
2. **Direct Data Sourcing** (DDS), which addresses the memory subsystem bottleneck problem on send,
3. **Application layer data integrity check** (CRC) offload, typically used in data critical applications, which are not satisfied with the relatively weak Internet checksum protection (e.g., iSCSI),

4. Further **application layer offload**, such as application layer payload recovery and end-to-end security protocol offload,
5. Per connection TCP level **traffic management** and quality of service

The list above is not exhaustive, nor can it be, since the possibilities are virtually limitless. Indeed, the ability of a TOE to touch the in-order, **process level byte stream** provides potentially as many different uses as there are TCP applications. When **byte touching** functionality is considered, the presence of a TOE improves system performance by several folds. For instance, a TOE can offload the payload CRC computation and provides zero copy on receive for iSCSI, a combination which has been measured to provide a 16 fold improvement in performance per CPU cycle compared to a software stack.

In summary, the core of these arguments is that TCP offload makes possible **Application Layer Offload**, which significantly increases its benefit to overall system performance, and boosts the value of the technology.

Total Cost of Ownership

We now move on to discussing the economic benefits of TOE, and its impact on the cost of deploying network applications. When working out the cost of deploying an application, it is essential to consider all the components of the complete system, including hardware, software, management, and various operational costs.

Equipment Costs

By significantly reducing the load on the host CPUs, as illustrated in the following section, TOE makes it possible to provide the same application level capacity using fewer CPUs and systems compared to non-offloaded adapters. Experimental results consistently show that TOE is capable of providing higher performance than non-offloaded adapters, at a fraction of the CPU usage. A typical example is that a system with a TOE can operate at line rate while using half a CPU. In contrast, a non-offloaded system burns two CPUs to achieve the same performance, and therefore requires additional CPUs to run the application itself. The following figure illustrates how the savings achieved by using a TOE instead of a regular NIC provide an impressive return on investment (ROI), given the economies of multi-processor systems (in this case, SunFire systems). The ROI is particularly interesting given that TOE comes at virtually no premium compared to a regular NIC.

Management and Development Costs

As demonstrated in the performance results of the following section, a TCP offload engine enables an Ethernet network to provide specialized interconnect performance in a familiar environment. TOE preserves the popular sockets layer for network programming, as well as the usual configuration tools used in the ubiquitous TCP/IP over Ethernet networks. This eliminates the need for dedicated personnel trained in exotic technologies to run computing clusters or other specialized networks, such as storage area networks (SANs). In fact, 10 Gbps Ethernet with TOE can be deployed as a single networking infrastructure within an institution, which integrates all the services needed. This unification results in further significant savings in equipment and management costs.

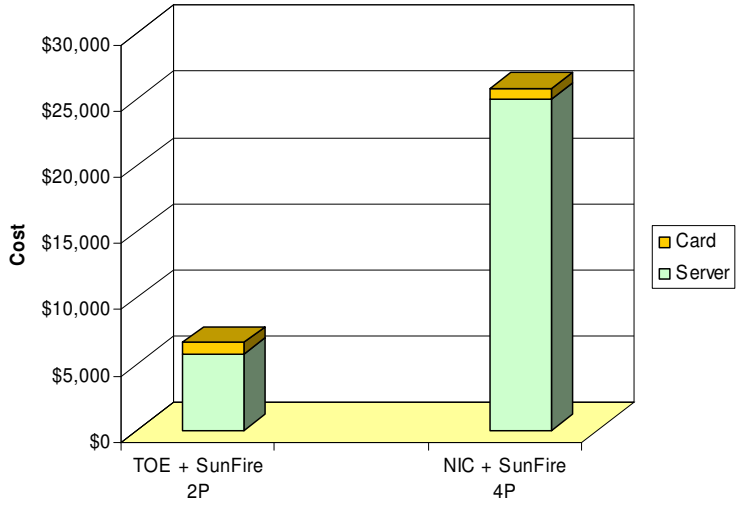

Figure 3: TOE obviates the need for an expensive 4-P system

Operational Costs

Not only does the reduction in required hardware resources directly translate into savings in equipment and real estate cost, but it also results in significant operational savings. Indeed, today's CPUs and the accompanying memory draw an order of magnitude more power than a network adapter, with corresponding power and cooling expenses. More precisely, Chelsio's TOE draws 5 Watts more than a regular NIC, while a high end CPU/RAM combination easily draws power in excess of 200 Watts. These may exceed the cost of the TOE over a one year period.

Software License Fees

Additionally, for some applications, it appears that hidden software costs may actually overshadow these considerations. For instance, database software is typically licensed on a per-CPU basis, including per core for multi-core dies. With license fees running as high as $8,000 per CPU, the benefits of releasing CPU cycles and decreasing the number of CPUs become obvious.

In conclusion, the performance benefits of a properly designed TOE translate into significant savings when considering the cost of a network application. These advantages to deploying Chelsio's offload enabled network adapters are further reinforced by the fact that Chelsio's TOE is offered at no premium compared to other, non-offloaded HBA's.

The following section provides experimental evidence of the benefits of TCP offload, as measured in independent published studies.

Chelsio's Terminator Performance

In this section, we present experimental evidence of the performance benefits of TCP offload, as implemented in the Terminator engine. The results presented here were first published in an independent study by the RADIANT group at the Los Alamos National Laboratory (LANL) [LANL05] and a joint study between LANL and the Network-Based Computing Laboratory at Ohio State University [OSU05].

The first graph below shows single connection uni-directional throughput obtained over 10 Gbps Ethernet link with and without TOE. Notice how, given a reasonable message size (e.g., 1KB),

the performance with TOE stabilizes at 7.5Gbps (limited by the PCI-X bus bandwidth), whereas without TOE it only peaks at less than 5Gbps. On average, TOE provides double the throughput of a conventional NIC at half the CPU utilization, a **factor of 4 reduction in total CPU needs**.

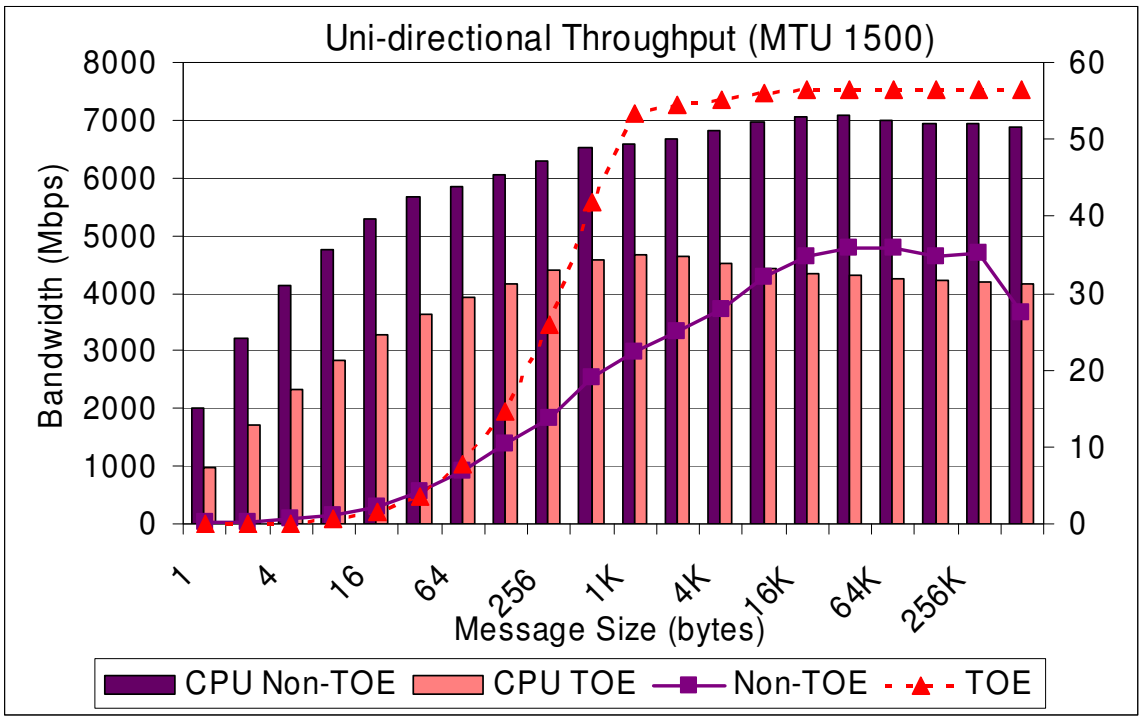

Figure 4 Uni-directional Throughput

Figure 3 shows a similar, but more pronounced, performance benefit in MPI throughput tests, confirming the observation made above.

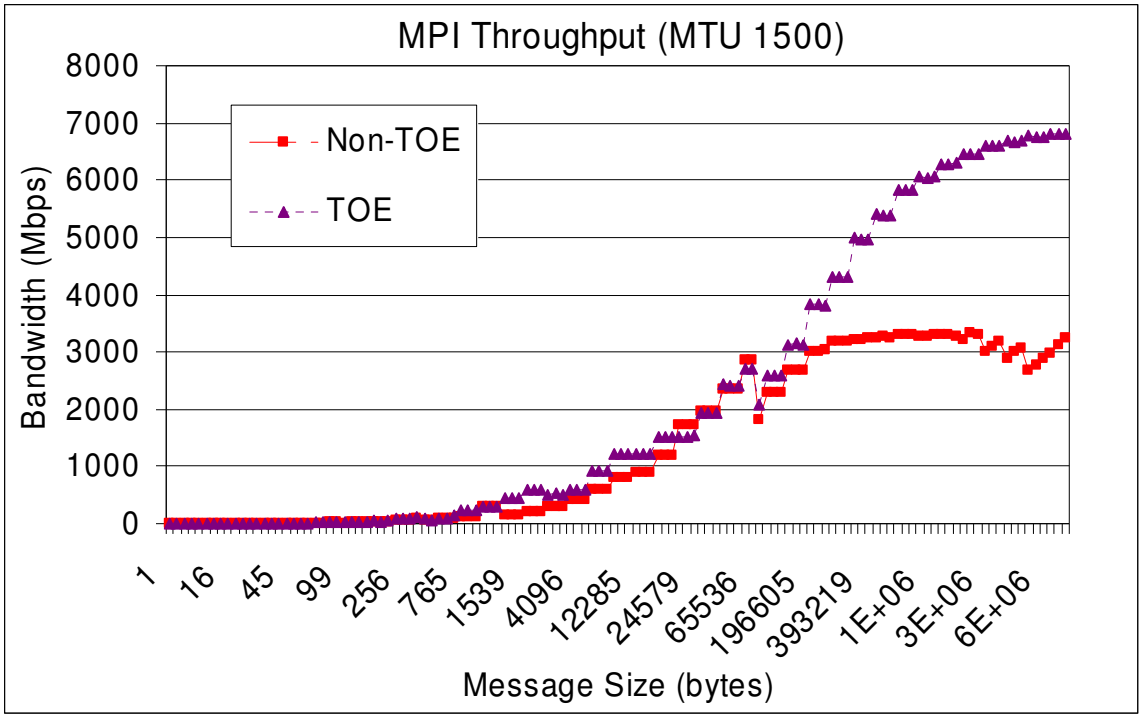

Figure 5 MPI Throughput

The following figure shows the performance of the popular Apache Web server in Transactions per Second (TPS). The alpha values correspond to different Zipf content popularity distributions. The tests demonstrate 60% to 1000% higher capacity with TOE compared to non-offload across the range.

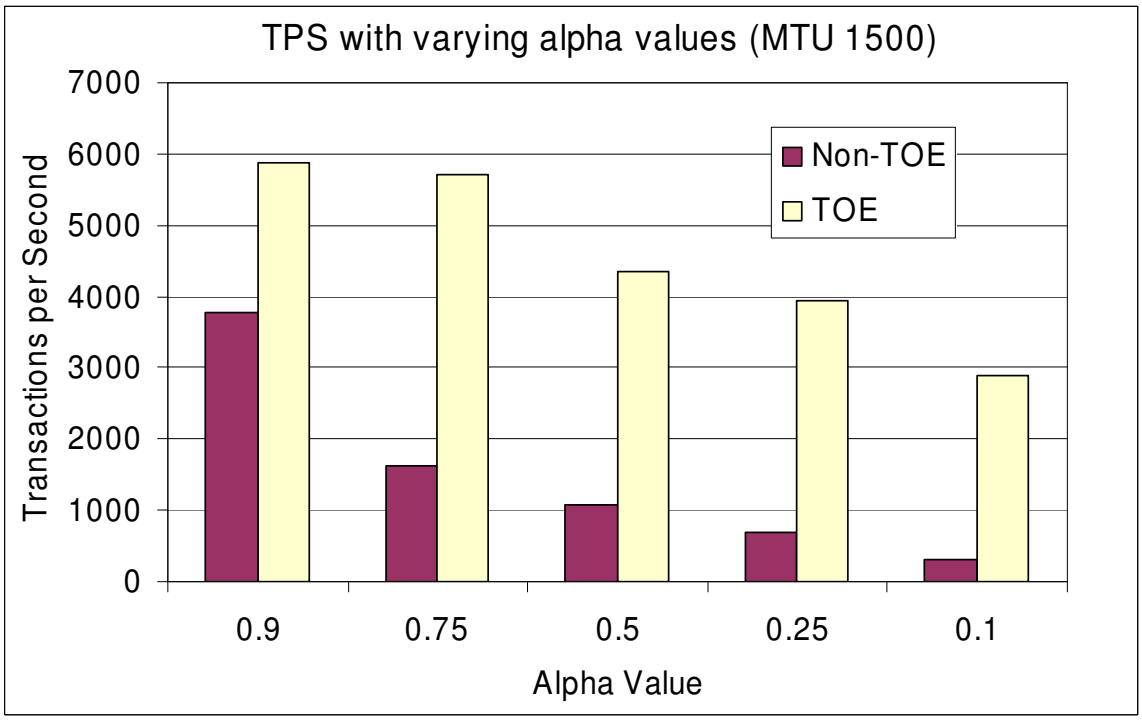

Figure 6 Apache Transactions per Second

The previous results having established the concrete performance benefits of offload for 10Gbps Ethernet, we now move to tests comparing 10 Gbps Ethernet with TOE to specialized interconnect technologies, namely InfiniBand and Myrinet.

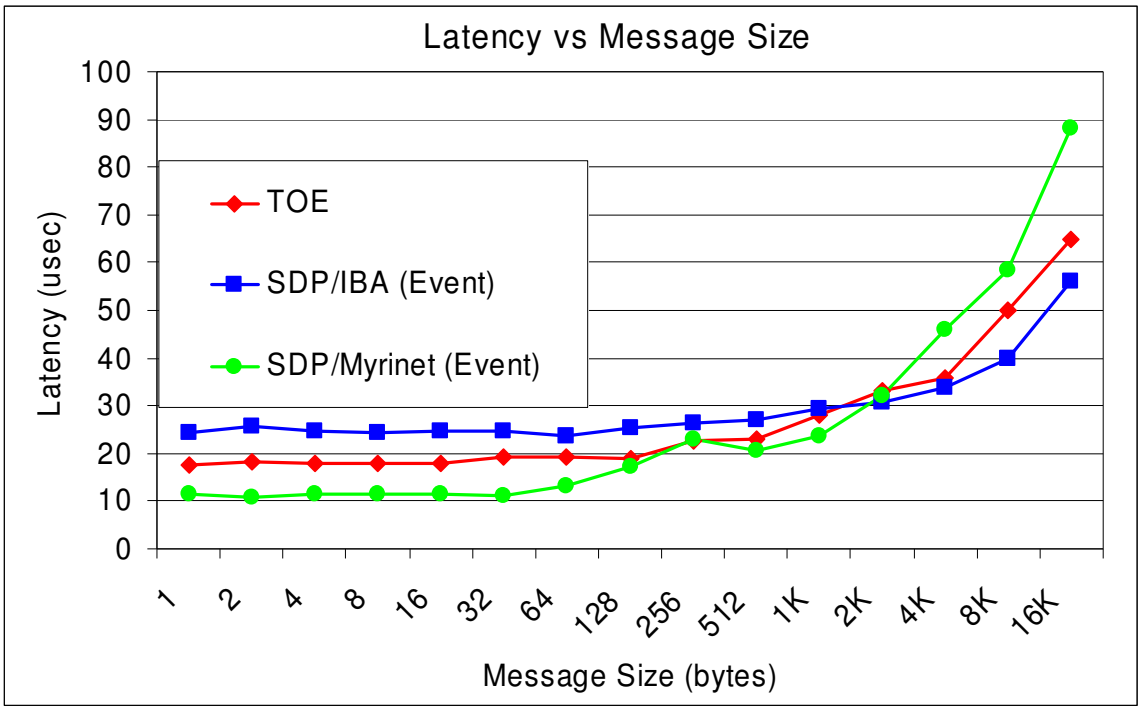

Figure 7 Latency vs. Message Size for the 3 Interconnects

Figure 5 above shows that, in terms of latency, 10 Gbps Ethernet with TOE is well within the range achieved by Infiniband and Myrinet in realistic usage scenarios. The following set of graphs show that Ethernet with TOE provides higher throughput in a distributed file system application than the other two interconnects, in both the Read and Write directions.

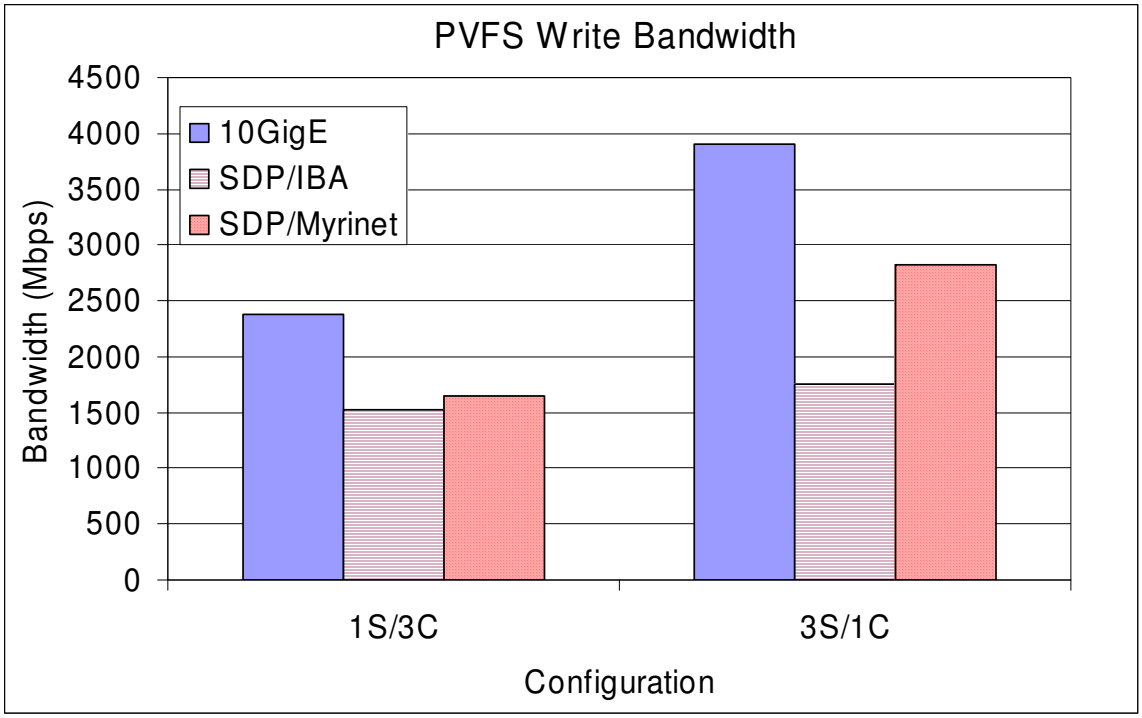

Figure 8 PVFS Write Bandwidth

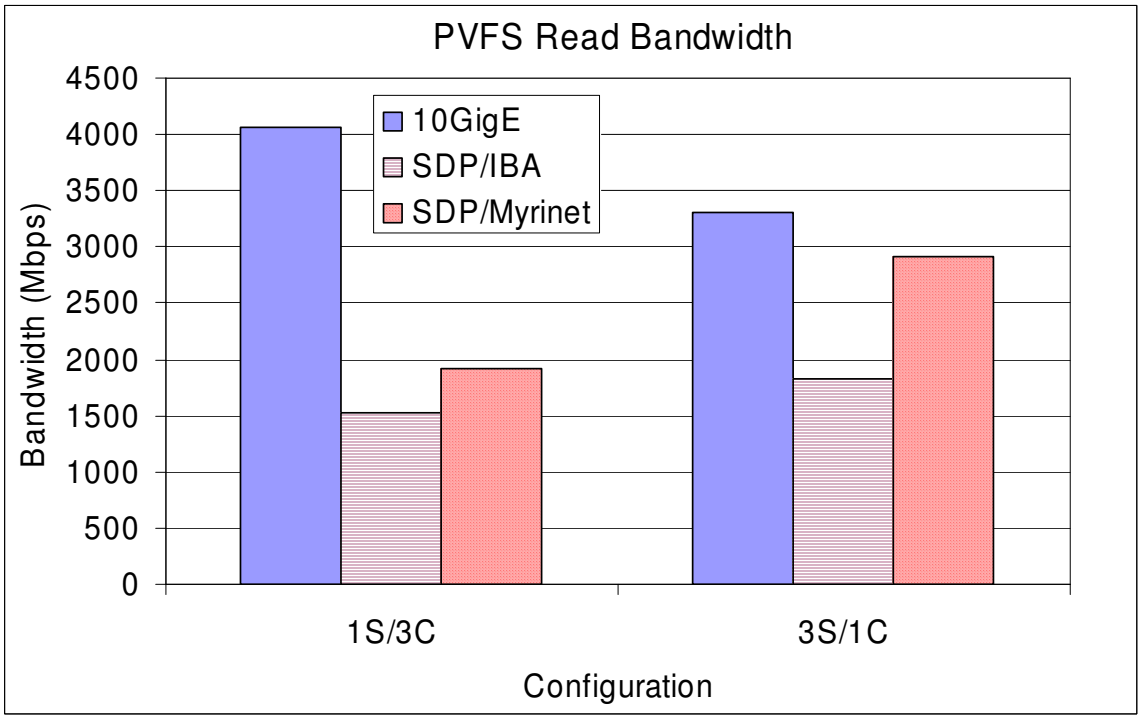

Figure 9 PVFS Read Bandwidth

The results presented here are but a sample of the data collected in the studies. The interested reader is referred to the corresponding papers, which present further evidence of the benefits of offload compared to conventional NICs, and demonstrate that TOE allows 10 Gbps Ethernet to become the interconnect of choice even for cluster computing applications,

which so far have relied on exotic technologies to provide the high performance levels required.

Conclusion: TOE's Time Has Come

This paper makes the case for TCP offload as the solution to today's computer system vs. network speed gap. The various non-offload NIC approaches proposed today were discussed and contrasted to the full offload approach taken in designing the architecture of Chelsio's Terminator engine. This architecture was shown to avoid the limitations attributed to TCP offload engines, which are largely due to the past experience with poorly designed implementations.

There are many reasons TOE is better positioned to succeed today than when the first attempts were made 10 years ago. ASIC technology now enables much higher design integration and complexity, TCP is mature, no major changes have happened to the reliable data delivery mechanisms since RFC1122 in 1989, and no significant changes in its congestion control mechanisms since NewReno and SACK in 1996. A programmable implementation is capable of implementing the latest high speed modifications suggested to these mechanisms. Finally, the lessons learnt from the mistakes of the past have allowed new design approaches, such as the Terminator architecture which avoids the pitfalls that have so far marred most attempts at TCP offload. Performance results of independent studies demonstrate the benefits of TOE compared to non-offloaded Ethernet NICs, and show that TOE allows Ethernet to compete head-to-head with specialized interconnects such as InfiniBand and Myrinet.

In the future, as system processing capabilities improve and network speeds increase, TOE will remain the technology which will bridge the persistent speed gap between the two.

For more information about Chelsio Communications and the Terminator architecture, visit the Chelsio web site at www.chelsio.com or send an e-mail to info@chelsio.com.

References

[INTEL03] J. Xu, N. Borkar, V. Erraguntla, Y. Hoskote, T. Karnik, S. Vangal, J. Rattner, "A 10 Gbps Ethernet TCP/IP Processor", In Proceedings of Hot Chips 15, August 2003.

[INTEL04] G. Regnier, S. Makineni, R. Illikkal, R. Iyer, D. Minturn, R. Higgahallu, D. Newell, L. Cline, A. Foong, "TCP Onloading for Data Center Servers", IEEE Computer, November 2004.

[CLARK89] D. D. Clark, V. Jacobson, J. Romkey, H. Salwen, "An Analysis of TCP Processing Overheads", IEEE Communication Magazine, Vol. 27, No. 2, June 1989.

[LANL05] W. Feng, P. Balaji, C. Baron, L. N. Bhuyan, D. K. Panda, "Performance Characterization of a 10-Gigabit Ethernet TOE", in Proceedings of HOT Interconnects, August 2005.

[OSU05] P. Balaji, W. Feng, Q. Gao, R. Noronha, W. Yu and D. K. Panda, "Head-to-TOE Evaluation of High-Performance Sockets over Protocol Offload Engines", in Proceedings of Cluster 2005, September 2005.