Enhancing GridFTP performance using intelligent gateways

Onur Demir*, Michael R. Head, Kanad Ghose and Madhusudhan Govindaraju

Department of Computer Science,
State University of New York,
Binghamton, 13902-6000 NY, USA
E-mail: onur@cs.binghamton.edu
E-mail: mike@cs.binghamton.edu
E-mail: ghose@cs.binghamton.edu
E-mail: mgovinda@cs.binghamton.edu
*Corresponding author

Abstract: To improve throughput of grid data servers under heavy loads or under denial of service attacks, it is important to service requests differentially, giving preference to ongoing or imminent client requests. We show how such features can be efficiently implemented on a gateway that controls access to a pool of servers. We present performance results for a prototype system based on a dual-ported active NIC, and demonstrate that an efficient differentiated service policy can be implemented on such a gateway to minimise the grid service response time and to improve server throughputs under heavy loads and denial of service attacks.

Keywords: GridFTP; active NIC; intelligent gateway; enhanced throughput; admission control policies.

Reference to this paper should be made as follows: Demir, O., Head, M.R., Ghose, K. and Govindaraju, M. (2008) 'Enhancing GridFTP performance using intelligent gateways', *Int. J. High Performance Computing and Networking*, Vol. 5, No. 3, pp.135–143.

Biographical notes: Onur Demir is a PhD student in the Department of Computer Science at Binghamton University. His research interests are network security, parallel programming and file system design. He has an MSc Degree from Yeditepe University and a BS Degree from Marmara University.

Michael R. Head is a PhD student in the Department of Computer Science at Binghamton University. He research includes Component Frameworks for the Computational Grid as well as High Performance Web Services. He was awarded an MA Degree in Computer Science in 2004 from Brandeis University and BS Degrees in Mathematics and Computer Science from Binghamton University in 1999. He spent two years working on custom software in the Banking and Financial Industry for IBM.

Kanad Ghose is a Professor and Chair in the Department of Computer Science at SUNY Binghamton. His research interests are in all performance-oriented aspects of networking and operating systems, power-aware processor design and high-performance computing. He received his MS and PhD in Computer Science from Iowa State University in 1986 and 1988 respectively. He received BTech and MTech Degrees in Electronics from Calcutta University in 1978 and 1982. He has published extensively in his areas of research. His research sponsors include DARPA, NSF, AFOSR, NYSTAR and the Industry.

Madhusudhan Govindaraju is an Assistant Professor in the Department of Computer Science at SUNY Binghamton. His research interests are in the areas of grid computing, distributed object systems, high performance data transfer, high performance RMI, web services, component based technologies and problem solving environments. He is actively involved in the design and implementation of the XCAT toolkit. He is leading the effort to design a comprehensive benchmark suite for communication in Grid Web services. He completed his MS in Computer Science from Indiana University in 1998. In 2002, he received his PhD in Computer Science from Indiana University.

1 Introduction

GridFTP has emerged as a de-facto standard for secure, reliable and high-performance data transfer across resources on the Grid (Allcock, 2003). In the course of its evolution through several versions, GridFTP has built on the core FTP-based data transfers with additional features such as parallel transfers, striped transfers, authentication

and facilities to invoke the user-transparent resumption of transfers affected by network problems (Allcock et al., 2002).

Widely accessible and utilised GridFTP servers can be exposed to different types of client conducts and abuses. A group of legitimate clients – a flash crowd – can all start accessing the site within a short time window, creating heavy traffic and heavy loading on the server. Flash crowds are non-coordinated, intense and overwhelming client requests to a server, often in response to some specific event. Compromised client machines or deliberately configured grid hosts can also launch a Denial-of-Service (DoS) attack on the GridFTP server, with similar consequences. DoS attacks are essentially attacks that saturate the use of some specific computing or networking resource(s) of a system, such as a server or a sub-network, and deny legitimate users the use of these resources. Over the recent years, such Distributed DoS (DDoS) attacks to popular websites have become quite commonplace; the GridFTP servers are not exempted from such attacks. The wide variations in the GridFTP server's performance caused by these events are undesirable for applications that have ongoing transfers. GridFTP servers (hereafter, also referred to as servers), are in fact, particularly vulnerable to such attacks and events that cause dramatic momentary or sustained increases in load levels on the server, as an initiated transfer takes a relatively long time to complete, particularly if un-cached or large files are involved in the transfer. Moreover, an existing transfer by itself can demand additional processing (and memory) resources at the server if concurrent channels are used and when retries are automatically invoked on some of these channels because of networking errors or congestion. Thus, an implicit requirement of a GridFTP server is that it must preferentially serve ongoing data transfers over new transfer requests or over other types of requests (when the server provides services in addition to GridFTP). Unless such a preference is given, the ongoing transfers may time out and get resubmitted soon thereafter, adding to the server's load, making a bad situation worse. Furthermore, the server's utilisation is also reduced as some or all of partial progress made on the transfer is aborted.

SYN attacks (or SYN flood attacks) are a common form of DoS or DDoS attacks on servers. In a SYN attack, a SYN packet is sent by the attacking host to seemingly initiate the first phase of the 3-way handshake to set up a TCP connection, usually using a spoofed IP address. The attacker never completes the remainder of the connection setup process and ties up resources on the server till the specified timeout for connection establishment (70 s to a few minutes, depending on the kernel) expire. Till then, enough resources are tied up to deny connection requests from legitimate clients. Another common form of DoS or DDoS attack is to tie up one or more types of resources on the server by repeatedly requesting downloads of large files. One notable feature of attacks in this category is that the source addresses are not spoofed, as a genuine connection must be set up to actually transfer the file. Real-time performance guarantees are difficult to provide for servers due to several reasons. Because of the open nature of servers, the number of clients at a specific time is not completely predictable, although good approximations are possible. Load balancing techniques may be helpful for compensating sudden peaks in demands.

In this paper, we present a technique for selective admission control, implemented on an active network card based gateway (a.k.a. intelligent gateway) to a pool of GridFTP servers, which allow these servers to selectively process requests related to an ongoing transfer under heavy, unanticipated load conditions. The intelligent gateway relieves the actual servers from:

- distinguishing packets for ongoing transfers from packets that belong to other requests
- the overhead of processing packets for requests that will have to be eventually dropped in the process of prioritising the existing transfers when the servers are heavily loaded
- the bookkeeping overhead needed to resume an ongoing GridFTP transfer that was disrupted due to network errors/conditions.

Consequently, the utilisation of the GridFTP server improves dramatically and the response time to transfer requests remain relatively stable under a DoS attack or on unexpected heavy load. We evaluate our technique using a prototype implementation and present the experimental results. Our tests involve running several concurrent downloads in striping mode under different conditions. The requesting scripts record the connection time (time from initiation until actual transfer begins) as well as the total transfer time and number of bytes received. The conditions under which we test include: DDoS attack (large number of spoofed SYN packets), high server CPU load, and high server I/O load. Our results show that in each case, we can provide a similar level of service to ongoing clients as during normal, 'base case' conditions.

A fundamental tenet of grids is that resources within an organisation should be governed by local rules and policies. The gateway to the resources of a local organisation is often responsible for handling load-balancing, minimising response-time, maximising throughput, and for verifying the security credentials of each incoming request. The solution in this paper is consistent with this tenet.

We have made significant enhancements to our earlier work in this area (Demir et al., 2005). We have replaced the python simulator script with a real standalone GridFTP server. As a result, our reported performance results are more realistic. We have implemented an admission control mechanism for enhancing performance of trusted clients that use a non-replicated GridFTP server. We have also included performance results to test the scalability of the active NICs.

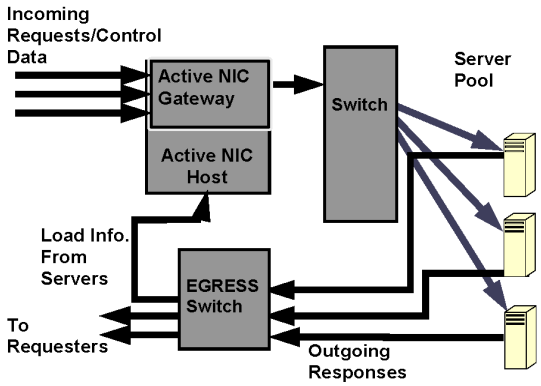
This paper is organised as follows. Section 2 briefly describes the concept of smart admission control. In Section 3, we first discuss the prototype system for the GridFTP server pool. Then, in Section 4, we describe admission control mechanisms. The scalability problem

is discussed in Section 5. The experimental results are presented in Section 6 and finally Section 7 presents our conclusions and future work.

2 Smart admission control

We consider a locally distributed server configuration, such as the one shown in Figure 1, where a pool of server machines implements the GridFTP server. The GridFTP server's performance can be severely limited by sudden increases in the requests for its services. Such increases will result in long response times or even in request time-outs. In general, as the request rates increase, the resultant increase on the server load causes the server response time to go up commensurately. Additionally, existing transfers are also delayed. To provide stable transfer times under abrupt increases in the load due to hostile events (such as a DoS attack) or due to rare but natural events (such as transfer resumption requests on network problems), an effective solution is to admit request packets selectively to the server. We now argue that such limiting is best performed by an intelligent gateway as it relieves the already-loaded server from the chores associated with such admission control.

Figure 1 Active NIC enabled GridFTP server architecture (see online version for colours)



To implement preferential admission control, the server has to track all ongoing transactions, the number of active service requests for each type of service (GridFTP and possibly others), and accept or deny incoming requests based on some criteria. However, this solution has some drawbacks. An individual server in a locally distributed server pool does not have information on the load and status of other servers. Consequently, server local decisions are not adequate in implementing load balancing or in inferring malicious events directed to the pool. Furthermore, under heavy load, the bookkeeping needed to monitor requests and to implement admission control policies can itself impose additional work on an already loaded machine. Finally, any malicious activity is hard to detect on individual servers.

Another solution may be to naively limit the incoming requests at the gateway leading to the server pool. This has some disadvantages. The ongoing GridFTP transfers are unknown to the gateway and associated packets may be dropped. It is also possible to deny the resumption request for an interrupted ongoing transaction request. A complete solution thus needs to take into account the context of a request. The load information of the servers is important as well; it is not possible to estimate a server machine's load by just examining the incoming packets. Load balancing can only be performed with accurate global knowledge of the load on each server machine.

It is precisely for the reasons listed above that we propose a solution of load and context conscious admission control to a GridFTP server pool using an intelligent gateway.

3 Prototype system details

Figure 1 shows the overall configuration for our prototype. One port of a dual-ported active Network Interface Card (NIC) based gateway acts as an interface to the GridFTP server. All admitted client traffic goes through the active NIC portal towards the server pool via the second interface on the active NIC. Responses from the server use a different path as shown, bypassing the gateway.

The active NIC is responsible for selecting and distributing incoming packets to the servers after subjecting them to a filtering rule. In particular, the intelligent gateway maintains information to prioritise ongoing transfers and information to perform load balancing. The server cluster provides a single IP (virtual IP, VIP) address to the internet, which is assigned to the incoming port of active NIC. The incoming packet headers are modified by the gateway, which changes the VIP with the IP address of the selected server machine. When the server machine responds to the request it uses VIP as the source IP.

The host, where active NIC is mounted (called the active NIC host), runs a daemon called the control agent. The control agent periodically collects information from server agents that run on the servers. The control agent uses this information to determine the dynamic packet filtering rules that have to be deployed on the gateway and updates the existing filtering rule set on the active NIC. Keeping the control agent on the active NICs host significantly eases the processing load on the active NIC.

In our prototype implementation, we have used a Ramix PMC 694 active NIC with dual 100 Mbits/sec Ethernet interfaces, two autonomous DMA controllers, a 233 MHz. Power PC CPU and 32 Mbytes of RAM and 8 Mbytes of Flash memory (Ramix Inc., 2002). The Ramix PMC 694 is a PCI card. The primary packet filter used on the active NIC gateway is based on the well-known BPF+ filter (Begel et al., 1999). The Ramix card runs RTEMS, a real-time operating system based on BSD UNIX and implements a TCP/IP stack to allow for TCP offloading. A proprietary library is used for communicating from the host PC to the PMC 694; this interface is not critical to the performance of our scheme. The server agents gather the information used to classify incoming packets as admissible

or non-admissible on a regular basis and pass this information to the control agent on the active NIC's host. The final decision for admission control and the dynamic alteration of the packet-filtering rule at the gateway is left to the control agent.

The data structure used to keep track of the IP addresses of hosts requesting a GridFTP transfer is PATRICIA tries, which is extremely efficient for inserting and searching such information (Morrison, 1968). The control agent, the server agent and the active NIC all use this data structure. The IP addresses of the clients constitute the keys in this data structure. Each entry has a time stamp for last access time. Entries are aged according to this time stamp, and eventually removed from the data structure when the last access time becomes older than one hour.

4 Admission control policies

To implement admission control policies for the GridFTP server, the intelligent gateway classifies requesting hosts by their source IP addresses into the following categories:

- *Green*. These are hosts that are currently in the middle of a GridFTP transfer. Our aim is to keep servicing these addresses regardless of the DoS attacks and loading caused by (non-GridFTP) services. This class has a dynamic nature and has to be updated regularly.
- *Unknown*. These are the hosts that have not used the GridFTP server within the finite history of server logs.
- *Preferred*. This optional class of hosts is specific to the server. The server can choose the set of preferred hosts that request file transfers based on the GridFTP authentication information, the host's domain, or any other criteria. Preferred hosts can also be specified through a static list.

After classifying the requesting hosts into groups, the control agent transfers the corresponding filter rule updates to the active NIC gateway. The load on the server and the number of half-open connections are the main criterion to decide what packets are allowed to enter the server. We considered two types of loading information for each server machine in the pool: CPU load and I/O load. CPU load can be measured by monitoring CPU utilisation and the I/O load is measured by monitoring number of I/O interrupts per second, and number of block operations done per second.

4.1 Admission control policy for coping with SYN attacks and SYN overloads

We now discuss the admission control policy for coping with SYN DoS attacks of traffic from requesters in the unknown category. Note that the naïve dropping of SYN packets or solutions like IP Traceback (Savage et al., 2000) for coping with SYN attacks will not be effective in dealing with this problem, as these solutions have no way of easily identifying packets that form part of or are associated with an ongoing GridFTP transfer (including automatically triggered retries for the transfer).

The server agent on each server machine monitors its pending connection queue (….. the SYN queue, used to queue up all incoming SYN packets – i.e., packets that have the SYN flag set in the IP header). Two thresholds, T1 and T2, where T1 > T2 are used to regulate incoming SYN packets in our admission control policy and these are used as follows:

- when the overall SYN queue size is below T1, SYN packets from all types of hosts (green, preferred and unknown) are allowed to enter the server
- when the SYN queue size is greater than T1 but less than T2, the admission control policy switches to soft limiting that permits packets from the green and preferred hosts to enter the server but limits the SYN packet rate from unknown hosts to only S1 SYNs per second
- when the SYN queue size exceeds T2, the admission control policy switches to hard limiting and further reduces the SYN arrival rate from unknown hosts to S2 SYN packets per second, where S2 < S1. S2, in fact can be zero.

For the experiments described later, we chose T1 and T2 as 550 and 700 respectively and as our default SYN queue limit was 769. We chose S1 and S2 as 7 and 2 respectively. The thresholds and SYN rates were determined to provide good overall performance over a wide range of GridFTP requests.

4.2 Admission control policy for coping with server overloads in a server pool

When a GridFTP server also provides other services, we need to have policies that allow the GridFTP services to remain stable despite loading on the server caused by the other services. We have considered two different types of loading on the server:

- 'compute' loading caused by the execution of scripts (such as cgi) that mostly consume CPU resources
- I/O loading caused by the file I/O accesses made by standard services (such as http).

The admission control policy implemented in this case requires the server agents to monitor the load level on their respective server machine. When the loading crosses a threshold level of L, the machine is considered to be heavily loaded and the server agent notifies the intelligent gateway to perform dynamic load balancing of the non-GridFTP requests at the gateway. As new non-GridFTP requests come in, they are preferentially directed to the machines that are not heavily loaded.

For the results reported later, we used L as 0.8 (that is 80%) of the peak compute or I/O load.

4.3 Admission control policy for coping with single server overloads

In Section 4.2, we discussed an admission control policy based on load balancing for a server pool where each GridFTP server is replicated. For a GridFTP server that has no replicas on the local network, local load balancing is not an option for enhancing the performance of trusted clients. If the server is running at a high CPU or I/O load level, as described in Section 4.2, the only way to enhance the throughput of the trusted clients is to limit the entry of requests from the other clients. To do this, the server agent again monitors the load levels on the GridFTP server. When the CPU or I/O load crosses a threshold level of *L*, the machine is considered to be heavily loaded and the server agent notifies the intelligent gateway to enable admission control on the requests coming from unknown clients at the gateway while giving trusted clients a higher priority for accessing the server. The gateway limits the requests from unknown clients in the following way. Each second, the first *M* requests coming from unknown clients are forwarded immediately to the server. Thereafter, within that one second period, *P* percent of the requests received from unknown clients are dropped at the gateway. A useful value of *M* can be found experimentally. This value is dependent on the characteristics of the server, the requests themselves, and the maximum number of requests that can be served. In essence, this admission policy provides a guaranteed minimum performance for unknown clients and preferential service for trusted clients under heavy server loading. Making *M* too large can result in too many requests from unknown clients to be served at the cost of serving fewer requests for trusted clients. Making *M* too small results in grossly limiting the service for unknown clients. A good choice of *M* is thus critical and can be chosen by the administrator to strike a good balance between the services rendered to both known and unknown clients under heavy loading. Similar policies can be applied to the choice of *P*. For the results reported here, we have chosen it to be 1/2 of the maximum number of requests the GridFTP server can sustain every second. Our experiments show the result of setting *P* to 30, 50 and 80%.

5 Scalability

The proposed solution was evaluated using a single active NIC gateway controlling access to a small server pool. A single gateway device may not be able to cope with the processing requirements for traffic directed at large server pools. The potential bottlenecks are the storage needed for the green or preferred class of IP addresses, the processing overhead for packet filtering and statistics collection at the gateway, and the performance of the gateway's network interfaces.

The proposed solution can be scaled up to meet the processing needs for protecting large-scale server pools as follows:

- Several active NIC gateways operating in parallel can be used. Multiple active NICs can be hosted on the common PCI bus of a single host. The PCI driver for the active NICs need to be modified to support the 'broadcast' of status information to all cards on a common PCI bus. This can be easily done by passing on the status information via a common memory-mapped buffer in the RAM of the host of the gateways. Additionally, a front-end load balancing switch can be used to direct the incoming server traffic to a specific gateway.
- The memory requirements for the IP address classes on each gateway can be prohibitive as the number of attacking clients increase. A solution here will be to use a dynamic data structure like MULTOPS (Gil and Poletto, 2001) that limit the storage usage and switch dynamically between maintaining information on a per IP address basis or on a subnet address basis depending on the amount of traffic data and offered traffic volume. We are currently implementing this alternative on our prototype.
- The processing capabilities on the active NIC gateways continue to increase steadily, and this offers some relief for the solutions targeting larger scale systems and traffic volumes. The emerging generation of cards from Ramix have such capabilities (dual or quad 1 GBits/secs interfaces, faster CPU, additional RAM etc.)

An alternative to using active NIC gateways will be to use network processors. We have an ongoing effort using the Intel IXP2400 NPU.

6 Experimental results

The servers used for the evaluation system are Pentium IV PCs running a modified version of Linux kernel 2.4.18. We used two switches and constructed two subnets with 100 Mbits/s ethernet. The server pool constitutes one subnet and the client GridFTP machines are from another subnet (representing the outside world). The active NIC is positioned as a gateway with its two ports connected to the two subnets. For some of the experiments, multiple addresses are assigned to network interfaces of client and load machines to extend the IP range. For each request, clients are able to select an IP assigned to the interface.

Requestors connect directly to the gateway, which in turn forwards packets to the server, rewriting packet headers, or potentially dropping packets when necessary. Our tests consist of running requestor scripts (calling globus-url-copy) repeatedly under the different experimental conditions. A client script connects, authenticates, sets up the connection parameters, and requests the transfer of a file. The server transfers the file as requested after which the client quits and reports connection latency (time from initiation of the connection until transfer begins), total time (time from initiation of the connection until transfer completes). To gather performance results, the

client script is run several times on each of the client machines.

The experiments investigating the stability of GridFTP under SYN attacks or SYN overloading involved the use of two client hosts that used IP spoofing to generate the SYN traffic. The scripts generating such traffic produced a maximum SYN traffic rate of 7500 SYN packets per seconds, sufficient to overwhelm the server when no admission control policy was implemented at the gateway. A separate set of hosts was used to generate the GridFTP requests. The experimental evaluations related to server overloading were carried out for two different types of loading: CPU-bound load and I/O bound load. We applied a simple rate limiting policy to provide clients with ongoing transactions to have an acceptable response time. For the experiments that used CPU-bound loading, the servers are flooded with client requests for number crunching applications. These applications mimic the action of CPU demanding services like encryption and data compression. We used http downloading requests to simulate I/O loading.

Adjusting the rate limits and deciding when to start and stop packet limiting are critical to our scheme. As explained in Section 4, the individual server hosts can generate an overloading notification for the gateway. A similar notification is generated when the SYN queue sizes at each server host crosses the thresholds described in Section 4.1. Depending on the type of notification, the gateway implements the admission control policies described in Sections 4.1 and 4.2. The thresholds (T1, T2, L) and rate limits (S1, S2) used for admission control impact the server response time and can be adjusted to maintain the desired performance characteristics.

Figures 2 and 3 show how client completion rate and server throughput are affected under a SYN attack. We ran our test under three different conditions as described in Section 4.1: tolerable SYN traffic rate, soft limiting and hard limiting conditions. The server's kernel was configured to buffer up to 769 SYN packets before dropping them. Because the connection timeout is set to 2 min, this means that if more than seven packets per second are forwarded to the server, the buffer will begin to overflow over time, and clients will not be able to connect. The server updates the gateway whenever a client successfully authenticates, and the gateway stores the IP address of that client. When the soft or hard limiting is enabled, the gateway never blocks SYN packets from the green and preferred hosts, so GridFTP clients that have ongoing transfers or clients that are preferred maintain a normal level of service even under a SYN attack. As can be seen in Figure 2, when there is no protection, and a SYN attack is under way, a very small number of client requests will actually be fulfilled. When the limit protections are in place, we find that the preferred client (prowl) is able to access the server as under normal conditions, without any time outs. Figure 3 shows the server side for the same experiment. When no protection is in place, the server is unable to send out any data. However, when the protections are enabled, overall server throughput is close to that of normal conditions for preferred client requests. Applying hard limits allows useful server throughput to be dedicated to preferred clients.

Figure 4 shows how reconnection latency is affected when new connections are balanced between two servers as in explained in Section 4.2. In this experiment, we have two servers handling all requests. Clients still connect directly to the active NIC gateway, but the gateway divides the requests between the two servers. The gateway consistently forwards packets from a given IP and port to the same server to properly maintain the connections. In our base case, there is no additional external CPU load applied to either of the servers beyond what is provided by the GridFTP requests. In the other two cases, other service requests (script execution requests and http downloads) provide the additional CPU and I/O loading. Without load balancing, new connection requests are distributed evenly between the two servers, when load balancing is enabled, the gateway polls the servers for their respective CPU utilisation at 30 second intervals. When a given server's CPU utilisation goes above threshold (L) of 80%, the active NIC begins sending a smaller percentage of new connection requests to it, directing the overwhelming majority of the 'load' requests to the second server host.

Figure 2 Client completion rate under SYN attack (see online version for colours)

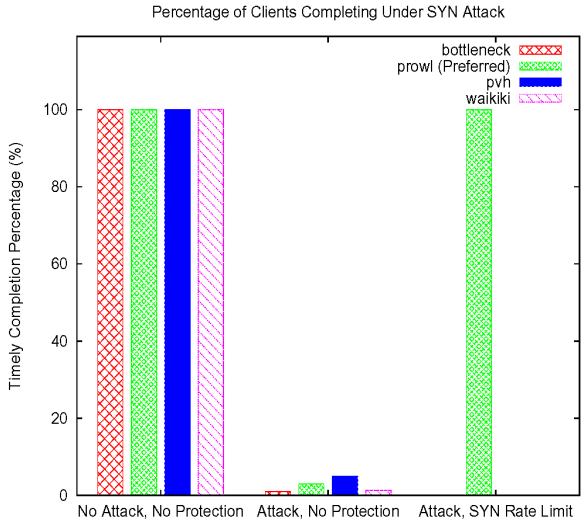
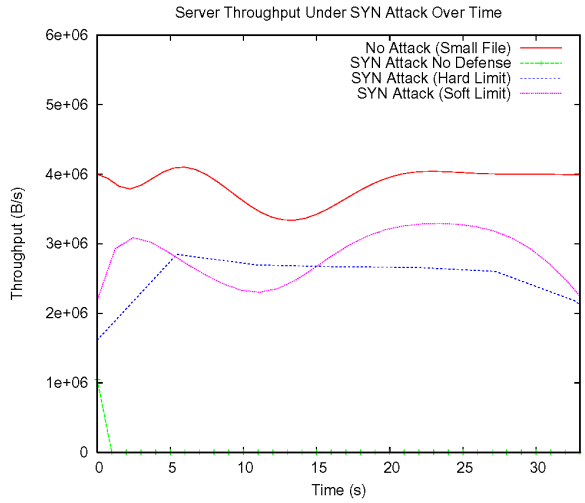
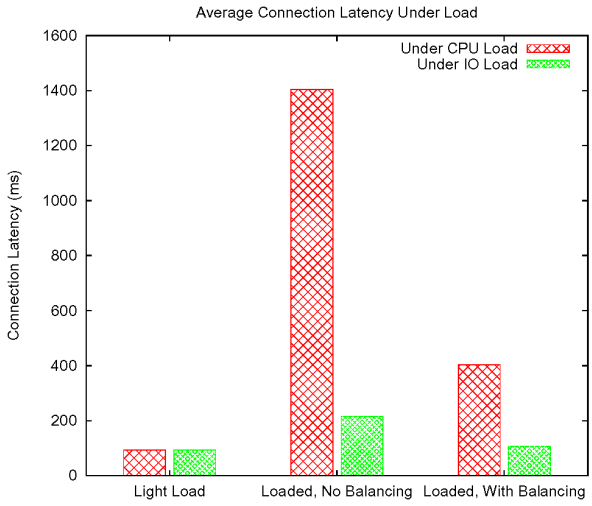


Figure 3 Server throughput over time during SYN attack (see online version for colours)



The results shown in Figures 2–4 clearly show that the performance of ongoing and preferred clients is maintained at levels very close to that of the base case even under SYN attacks and under very heavy server loading.

Figure 4 Connection latency under CPU and I/O load (see online version for colours)



We designed an experiment to demonstrate an improvement for soft real-time requirements for preferred clients. Assuming that the clients expect a bound on the service time for a given file size, a transfer should finish within that time-out period. In order to determine a time-out period for each file class, we measured the average transfer times for a wide range of file sizes within each class during lightly-loaded conditions and multiplied this duration by a factor of K to obtain the expected service time under heavy load conditions. In order to minimise the number of time-outs during periods of heavy server load, we used the Active NIC gateway to allow/deny requests based on the class of the requested file as explained in Section 4.3. As shown in Figures 5 and 6, the trusted client's performance can be enhanced during heavy CPU or IO load. Under heavy load conditions, our algorithms give preference to the trusted clients by limiting the new requests from non-trusted clients.

Figure 5 shows the server throughput for the different rate limits. The server throughput is negatively affected by the more aggressive admission control policies. However the response time and completion rate of the preferred clients gets better with admission control as seen in Figure 6. In Figure 6, we observe that the percentage of preferred requests that were completed on-time under high CPU load is approximately 35%. When we apply admission control, the request completion rate reaches as high as 65% by applying admission control to other client's requests. If we apply less aggressive admission control policies (the 50% and 80% admission rates), the performance profile becomes similar to the case without admission control.

Figure 7 shows the results of the experiment to test the scalability of the active NIC under heavy loading conditions. The filtering rules on the active NIC are strictly related to the number of trusted clients. Here we have used four differently sized sets of trusted clients, from 16 up to 128, to show the effect of the number of rules on performance of the active NIC based gateway to the GridFTP server. Figure 7 shows that the number of rules applied to the incoming traffic, by the gateway, has negligible effect on the throughput of the server.

Figure 5 Server throughput over time under IO load (see online version for colours)

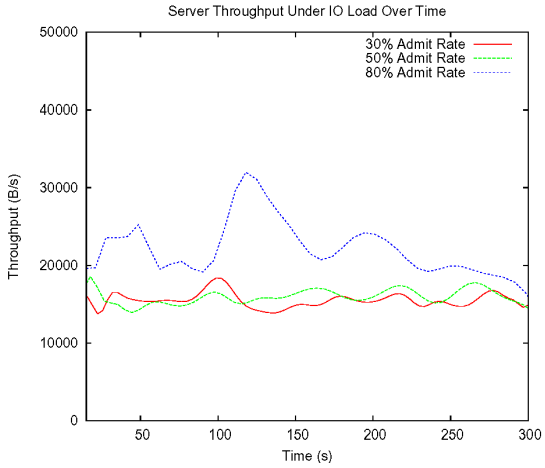


Figure 6 Percent of completed client requests under CPU load (see online version for colours)

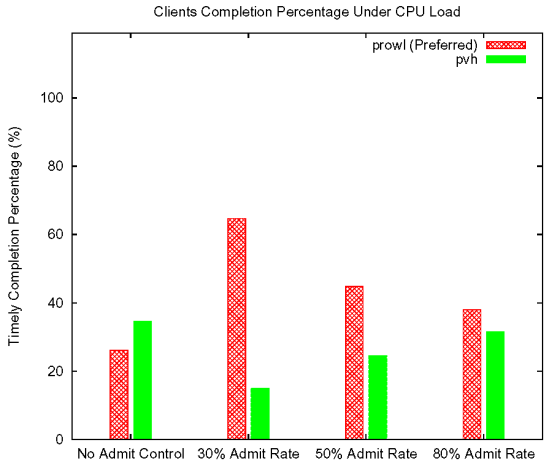
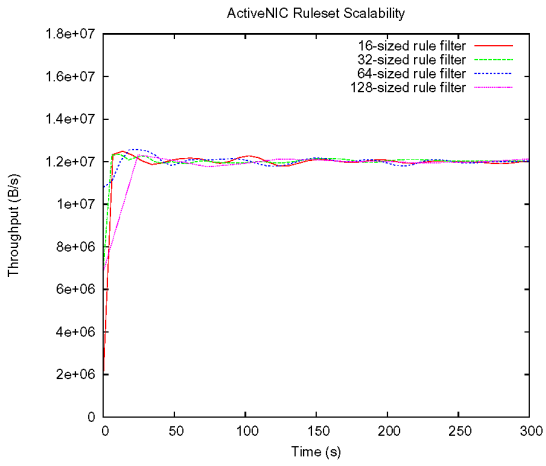


Figure 7 Scalability test for number of rules in the active NIC filter (see online version for colours)



7 Related work and conclusions

Our work involves providing differentiated service to GridFTP. A substantial amount of related work has been developed in support of these techniques for web servers, though little, as yet, as been developed for the GridFTP service.

There is a plethora of work in supporting differentiated services on web servers. Some examples follow. Operating System facilities for supporting differentiated services are explored in Aron (2000). The work of Chandra et al. (2000) uses transcoding technique to vary content resolution/quality to meet QoS needs on a per-client basis. The work of Zhang et al. (2002) proposes a technique for dynamically partitioning a server pool into classes and assigning servers to a specific class. In Chen and Mohapatra (2003) session-based relationships among http requests are used to device traffic conformation functions that are used for resource allocation to limit server overloading. In Ranjan et al. (2002), an adaptive technique for determining the number of servers needed to service requests with specific targets is introduced and evaluated through simulations against optimal configurations. All of these techniques allow packets to enter the server and then are differentiated within the server. This implies that the servers take a performance hit to examine an incoming request and then either rejecting it or delaying its service. The performance hit can be substantial under flash crowd traffic or when a DoS attack is in progress. We filter low priority requests at the gateway, freeing up the server resources to perform the services for the high priority classes.

A complete solution for dealing with DDoS attacks, by necessity has to be distributed and requires the coordination of several entities on the network. Since many types of DDoS attacks use spoofed IP source addresses, a rather naive prevention mechanism is to use simple egress filtering – filters in switches take the traffic out of a subnet to ensure that the source addresses of packets going out corresponds to valid host IP addresses within the subnet. Although it sounds simple, this solution is not practical – the large majority of subnets do not have egress routers with this capability; neither will this scheme be of any use unless the filters are configured correctly. IP-traceback – tracing packets back to the source – and similar techniques can be used to trace a large and unusual influx of packets from a specific port (or set of ports). With the use of traceback, controlling or limiting packet flow is a more sophisticated and distributed mechanism for coping with DDoS (Bellovin, 2000). Mazu networks offers a commercial product for defending against DDoS attacks, that relies on traffic flow monitoring (Mazu Networks, 2004); some other vendors offer similar products as well. Other distributed solutions for coping with DDoS are possible, including the use of trusted network components. Until these distributed solutions are standardised and widely adopted, servers have to deploy local solutions to protect themselves. Traceback and similar solutions (based solely on the monitoring of packet flow towards the servers) are generally incapable of dealing with load attacks, which do not always manifest themselves as a sudden burst of unusually heavy traffic. Furthermore, to detect such attacks, the en-route routers need to have the capability of examining the payload in the requests. Load attacks can be better dealt with by using the actual loading information at the servers. Distributing such loading information to en-route routers can be time consuming and complex – and, perhaps, practically infeasible. Solutions implemented on gateways closer to the server that incorporate the servers' loading information to perform dynamic packet filtering, as proposed in this paper, appear to be more attractive in coping with DDoS attacks.

We presented an intelligent gateway based solution for supporting differentiated service for a GridFTP server that preferentially services known clients under DDoS attack, and actively manages server load distribution based on the servers' systems' statistics. The capabilities of the active NIC-based gateway permit a dynamic mechanism to react intelligently to a denial of service attack, as well as external load on the servers to be efficiently implemented. The packet filtering rules at the gateway are dynamically altered based on the incoming packet rate and dynamic loading information periodically collected from each of the servers in the server pool.

We demonstrated how a flexible admission control policy can be implemented at the gateway to provide differentiated service to various client classes. The clients are classified based on whether or not they are known to the server. We also showed that the desired degree of real-time performance (bounded response time) can be guaranteed even under heavy server loading and denial of service attack by choosing rate limits appropriately. The proposed system is scalable, flexible and provides continuous service of the servers by performing dynamic request rate limiting at the active NIC-based gateway.

Acknowledgement

This work supported in part by the NSF under award No. EIA 9911099 and CNS 0454298.

References

Allcock, W. (Ed.) (2003) *GridFTP: Protocol Extensions to FTP for the Grid*, Global Grid Forum Draft, April, http://www.ggf.org/documents/GWD-R/GFD-R.020.pdf

Allcock, W., Bester, J., Bresnahan, J., Chervenak, A.L., Foster, I., Kesselman, C., Meder, S., Nefedova, V. and Quesnel, D., Tuecke, S. (2002) 'Data management and transfer in high performance computational grid environments', *Parallel Computing Journal*, Vol. 28, No. 5, pp.749–771.

Aron, M. (2000) *Differentiated and Predictable Quality of Service in Web Server Systems*, PhD Dissertation, Rice University, http://cs-tr.cs.rice.edu/Dienst/UI/2.0/Describe/ncstrl.rice_cs/TR00-369

Begel, A, McCanne, S. and Graham, S.L. (1999) 'BPF+: exploiting global data flow optimization in a generalized packet filter architecture', *Proc. SIGCOMM 99*, Cambridge, Massachusetts, USA, pp.123–134.

Bellovin, S.M. (Ed.) (2000) *ICMP Traceback Messages*, Internet Draft, http://lasr.cs.ucla.edu/save/rfc/draft-bellovin-itrace-00.txt

Chandra, S., Ellis, C.S. and Vahdat, A. (2000) 'Differentiated multimedia web services using quality aware transcoding', *Proc. 19th Annual Joint Conference of the IEEE Computer and Communications Societies. Proceedings (INFOCOM '00)*, Tel Aviv, Israel, March, pp.961–969.

Chen, H. and Mohapatra, P. (2003) 'Overload control in QoS-aware web servers', *Computer Networks: The International Journal of Computer and Telecommunications Networking*, Vol. 43, No. 1, May, pp.119–133.

Demir, O., Head, M.R., Ghose, K. and Govindaraju, M. (2005) 'Protecting grid data transfer services with active network interfaces', *Proceedings of Grid 2005 – 6th IEEE/ACM International Workshop on Grid Computing*, Seattle WA, pp.9–16.

Gil, T.M. and Poletto, M. (2001) 'MULTOPS: a data-structure for bandwidth attack detection', *Proceedings of USENIX Security Symposium'2001*, Washington DC, pp.23–38.

Mazu Networks (2004) *Enforcer*, Product Information and White Papers at: http://www.mazunetworks.com

Morrison, D.R. (1968) 'Patricia-practical algorithm to retrieve information coded in alphanumeric', *Journal of ACM*, Vol. 15, No. 4, pp.514–534.

Ramix Inc. (2002) *Intelligent Ethernet Adapter Product Guide*, available at: http://www.ramix.com

Ranjan S., Rolia J., Fu H. and Knightly E. (2002) 'QoS-driven server migration for internet data centers', *Proceedings of the Tenth International Workshop on Quality of Service (IWQoS 2002)*, Miami, USA, pp.3–12.

Savage, S., Wetherall, D., Karlink, A.R. and Anderson, T. (2000) 'Practical network support for IP traceback', *Proceedings of 2000 ACM SIGCOMM Conference, ACM SIGCOMM*, Stockholm, SE, August, pp.295–306.

Zhang, J., Hämäläinen, H. and Joutsensalo, J. (2002) 'A new mechanism for supporting differentiated services in cluster-based network servers', *Proceedings of the 10th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems (MASCOTS'02)*, Washington DC, p.427.