GFD-I.037
Category: Informational
Grid High Performance Networking Research Group

Volker Sander (Editor)
November 22, 2004

Networking Issues for Grid Infrastructure

Status of this Memo

This memo provides information to the Grid community. It does not define any standards or technical recommendations. Distribution is unlimited.

Comments

Comments should be sent to the GHPN mailing list (ghpn-wg@gridforum.org).

Abstract

Grids are built by user communities to offer an infrastructure helping the members to solve their specific problems. Hence, the geographical topology of the Grid depends on the distribution of the community members. Though there might be a strong relation between the entities building a virtual organization, a Grid still consists of resources owned by different, typically independent organizations. Heterogeneity of resources and policies is a fundamental result of this. Grid services and applications therefore sometimes experience a quite different resource behavior than expected. Similarly, a distributed infrastructure with ambitious service demands stresses the capabilities of the interconnecting network more than other environments. Grid applications therefore often identify existing bottlenecks, either caused by conceptual or implementation specific problems, or missing service capabilities. Some of these issues are listed in this document. The Grid High-Performance Networking (GHPN) Research Group focuses on the relationship between network research and Grid application and infrastructure development. This document summarizes networking issues identified by the Grid community.

Table of Contents

1. Introduction

The Grid High-Performance Networking (GHPN) Research Group focuses on the relationship between network research and Grid application and infrastructure development. This document summarizes networking issues identified by the Grid community.

2. Scope and Background

Grids are built by user communities to offer an infrastructure helping the members to solve their specific problems. Hence, the geographical topology of the Grid depends on the distribution of the community members. Though there might be a strong relation between the entities building a virtual organization, a Grid still consists of resources owned by different, typically independent organizations. Heterogeneity of resources and policies is a fundamental result of this. Grid services and applications therefore sometimes experience a quite different resource behavior than expected. Similarly, a distributed infrastructure with ambitious service demands stresses the capabilities of the interconnecting network more than other environments. Grid applications therefore often identify existing bottlenecks, either caused by conceptual or implementation specific problems, or missing service capabilities. Some of these issues are listed below.

3. Use Case Documents

Several reports did already summarize the particular demand of Grid applications. This section gives lists some major result.

ENACTS is a Co-operation Network in the 'Improving Human Potential Access to Research Infrastructures' Programme funded by the DGXII's IHP programme and key user groups. The network produced a set of studies of key enabling technologies including a report about Grid service requirements [ENACTS]. The report summarizes 85 responses of a questionnaire comprised of 48 points. Among other things it states that an increase of the raw performance of the network infrastructure will not be sufficient to implement Grid services relying on **network QoS**.

The e-Science Gap Analysis [GAPANA] is a comprehensive survey on the results of personal interviews from mid February to early April 2003 of 80 scientists concerning the current state of Grid and Cyberinfrastructure technology with respect to their use in e-Science. In the context of the Grid-Network interface, the report states the demand for treating the network as a Grid resource hat supports **network resource reservation and claiming**. The provided layer-3 **diffserv**-based services, extended layer-2 **VLANs** and **point-to-point switched layer-1 connections** should be accessible through the Grid middleware.

The EU DataGrid project [DATAGRID] estimates the scale of the aggregate bandwidth requirements for High Energy Physics, Earth Observation and Medical Images applications. For HEP Monte Carlo data access it is assumed that 0.15 TBytes of data must be shipped from store to processing site. DataBase access implies few GBytes transfers in minute's scales. For bulk data, the requirement is to transfer few 100 GBytes in day scale. Service qualities other than bulk best-efforts will be required by many applications. Remote interactive and control traffic will require low delay and packet loss for small traffic volumes. Access to **VPN facilities** comes to be another requirement.

The GRIDWELTEN project was funded by the he German Ministry of Research and Development (BMBF) under the control of DFN with the goal to evaluate the of high-performance computing resources through Grid software architectures. The final report of the project [GRIDWELTEN] summarizes the results of a user requirements survey in German HPC centers. The survey was accomplished by a questionnaire consisting of 47 questions completed by 63 users. A section of the questionnaire was dedicated to the evaluation of the current mode of Grid usage including the particular networking requirements, with a focus on MPI-based communication of parallel programs. The report concludes that the majority of users require **network latencies to be in range of 0 up to 100 microseconds**, while typical programs were strongly dependent on network bandwidth with a good portion that required **bandwidth of over 1 Gbps**. As a consequence, distributed supercomputing applications (i.e. metacomputing) typically have quite challenging demands in terms of the network.

Within the Global Grid Forum, the Open Grid Services Architecture Research Group discusses a draft document that describes use cases [OGSACASES]. It lists a set of OGSA services that need to be carefully coordinated. With respect to the networking issues that were summarized in this document, it states that resource brokers must assure the availability of compute, data storage and network bandwidth for on-time simulation and analysis. Hence, different types of brokers must be carefully coordinated. The Grid Resource Allocation Agreement Protocol Working Group discusses a usage scenario document [GRAAP] in which the demand for a coordinated resource allocation is also related to the network. As a concrete example, it lists the network demand of the UK-Reality Grid project. The most pressing requirement for **advance reservation** in RealityGrid arises out of the need to co-allocate (or co-schedule) processors to run a parallel simulation code and multiple graphics pipes and processors on the visualization system. Based on current projections, the largest computationally-steered simulations that RealityGrid is likely to undertake will require bandwidth between the simulation and visualization systems of order 1 Gbps in order to achieve satisfactory interactivity. The bandwidth requirements between visualization systems are less demanding – 100 Mbps will be adequate for most purposes – but reasonably **good latency and jitter characteristics** are desirable. Thus the ability to make advance reservations of network bandwidth with certain quality of service characteristics and using the same protocols as for the reservation of processors are seen as desirable by RealityGrid.

Summarize:

High throughput	**bandwidth of over 1 Gbps.** Tera Bytes transfers
High performance and QOS control	Grid services relying on **network QoS.** **Reasonably good latency and jitter characteristics. For a**

	broad range of parallel programs a spreading among distributed Grid resources would require network latencies below 100 microseconds
Resource reservation	**network resource reservation and claiming**
Advanced Network services accessibility	**diffserv**-based services, extended layer-2 **VLANs** and **point-to-point switched layer-1 connections** should be accessible through the Grid middleware.
Security	**Access to VPN facilities**

The network, seen as a resource of the Grid environment should provide:

- High performance transport for bulk data transfer (over 1Gb/s per flow)

- Performance controllability to provide ad hoc quality of service and traffic isolation.

- Dynamic Network resource allocation and reservation

- Security controllability to provide a trusty and efficient communication environment when required

- High availability when expensive computing or visualization resources have been reserved

- Multicast to efficiently distribute data to group of resources.

Other issues have been pointed out by the networking community:

- What is the grid traffic impact on infrastructures and other traffics?

- How to integrate wireless network and sensor networks in Grid environment?

Six main functional requirements expressed by grid applications are selected and examined in the rest of this document: high throughput, performance controllability, network resource reservation capability, security controllability, high availability, and multicast. Finally, the remaining two issues are discussed. For each functional requirement, the text identifies current issues and lists existing solutions and proposed alternatives. We attempt to provide a typology of issues to identify long terms research areas, medium term research areas and short terms engineering works.

4. High Throughput

This section discusses issues related to High Performance transport and especially limits encountered with the TCP protocol.

Requirements:	1) High average throughput 2) Advanced protocol capabilities available and usable at the end systems 3) Lack of use of QoS parameters
Current issues	1) Low average throughput 2) Semantic gap between socket buffer interface and the protocol capabilities of TCP.
Analyzed reasons	1a) End system bottleneck, 1b) Protocol misconfigured, 1c) Inefficient Protocol 1d) Mixing of congestion control and error recovery 2a) TCP connection Set up: Blocking operations vs asynchronous 2b) Window scale option not accessible through the API
Available solutions	1a) Multiple TCP sessions 1b) Larger MTU 1c) ECN
Proposed alternatives:	1) Alternatives to TCP (see DT-RG survey document) 2) OS by-pass and protocol off-loading 3) Overlays 4) End to end optical paths

4.1 Issues related to API

This section describes problems encountered with the limited functionality offered by programming interfaces to transport protocols, with a focus on TCP.

The evolution of the Transmission Control Protocol (TCP) is a good example on how communication protocols evolve over the time. New features were introduced to address experienced shortcomings of the existing protocol version. However, new optional features also introduce more complexity. In the context of a service oriented Grid application, the focus is not on the various protocol features, but on the interfaces to transport services and the end-to-end performance obtained. Hence, the question arises whether the advanced protocol capabilities are actually available at the diverse end-systems and, if they are, which usage constraints they imply

A widely deployed interface to implementations of the TCP protocol stack is provided by the **Berkeley socket interface** which was developed at the University of California at Berkeley as part of their BSD 4.1c UNIX version. The fundamental abstraction of this API is that communication end-points are represented as a generic data structure called socket [RFC147]. The interface specification lists a set of operations on sockets in a way that communication can be implemented using standard input/output library calls. It is important to note that the abstraction provided by sockets is a multi-protocol abstraction of communication end-points. The same data structure is used with Unix services as files, pipes and FIFOs as well as with UDP or TCP end-points.

Though the concept of sockets is close to that of file descriptors, there are, however, essential differences between a file descriptor and a socket reference. While a file descriptor is bound to a file during the open() system call, a socket can exist without being bound to a remote endpoint. For the set up of a TCP connection sender and receiver have to process a sequence of function-calls which implement the three-way handshake of TCP. While the sender issues the connect()-call, the receiver has to issue two calls: listen() and accept().

An important aspect is the relation between the above listed call-sequence and the protocol processing of the TCP handshake. While the listen()-call is an asynchronous operation which is related to the receipt of TCP-SYN-messages, connect() and accept() are typically blocking operations. A connect()-call initiates the three-way handshake, an accept call processes the final message.

There is, however, a semantic gap between the programming interface introduced by the socket abstraction and the protocol capabilities of TCP. While the protocol itself offers the explicit use of the window scale option during the three-way handshake, there is no way in commonly used operating systems to explicitly set this option by issuing a specific call to set particular properties, the setsockopt()-call.

In fact, the window scale option is typically derived from the socket buffer size used during the connect()- and listen()-call. Unfortunately, this selection is often done on a minimum base which means that the minimum required window-scale option is used. To explain this mechanism in more detail, suppose that the used socket buffer size would be 50KB,100KB, and 150KB.

In the first case, the window scale option would be not used at all. Because the TCP protocol does not allow updating the window scale option afterwards, the maximum socket buffer size for this session would be 64KB, regardless whether socket-buffer tuning libraries would recognize a buffer shortage and would try to increase the existing buffer space.

In the second case, many operating systems would select a window scale option of 1. Hence, the maximum socket buffer size would be

128KB. In the final case, the window scale option used is 2 which results in a maximum buffer size of 256KB.

This argumentation leads to the conclusion that any buffer tuning algorithm is limited by the lack of influencing the window-scale option directly.

4.2 Configuration Issues

Similarly to the above described influence of the selected socket buffer size, the configuration of the operating system does have a strong impact on the achievable level of service. Widely deployed operating systems offer a broad variance of tuning parameters which immediately affect the higher-layer protocol implementations.

For UDP based applications, the influence is typically of less importance. Socket buffer related parameters such as the default or maximum UDP send or receive buffer might affect the portability of applications, i.e. by limiting the maximum size of datagrams UDP is able to transmit. More service relevant is the parameter which determines whether the UDP checksum is computed or not.

The potential impact on TCP based applications, however, is more significant. In addition to the limitation of the maximum available socket buffer size, a further limitation is frequently introduced by system wide limitations of the **congestion window** as well. Here, an operating system tuning parameter additionally limits the usable window size of a TCP flow and might therefore affect the achievable goodput even though the application explicitly sets the socket buffer size. Further on, parameters such as delayed acknowledgements, Nagle algorithm, SACK, and path MTU discovery do have an impact on the service.

4.3 OS and system-level optimizations

The evolution of end-to-end performances hinges on the specific evolution curves for CPU (also known as Moore law), memory access, I/O speed, and network bandwidth (be it in access, metro, core). A chief role of an operating system is to strike an effective balancing act (or, better yet, a set of them) given a particular period in time along the aforementioned evolution curves. The operating system is the place where the tension among curves proceeding at different pace is first observed. If not addressed properly, this tension percolates up to the application, resulting in performance issues, fairness issues, platform-specific counter-measures, and ultimately non-portable code.

To witness, the upward trend in network bandwidth (e.g., 100Mb/s, 1Gb/s, 10 Gb/s Ethernet) put significant strain on the path that data follow within a host, starting from the Network Interface Card (NIC) and finishing in an application's buffer (and vice-versa). Researchers and entrepreneurs have attacked the issue from different angles.

In the early '90's, [FBUFS] have shown the merit of establishing shared-memory channels between the application and the operating system, using immutable buffers to shepherd network data across the user/kernel boundary. The [FBUFS] gains were greater when supported by a NIC such as [WITLESS], wherein buffers such as [FBUFS] could be homed in the NIC-resident pool of memory. Initiatives such as [UNET] went a step further and bypassed the operating system, with application's code directly involved in implementing the protocol stack layers required to send/receive PDU to/from a virtualized network device. The lack of system calls and data copy overhead, combined with the protocol processing becoming tightly coupled to the application, resulted in lower latency and higher throughput. The Virtual Interface Architecture (VIA) consortium [VIAARCH] has had a fair success in bringing the [UNET] style of communication to the marketplace, with a companion set of VI-capable NICs adequate to signal an application and hand-off the data.

This operating system bypass approach comes with practical challenges in virtualizing the network device, while multiple, mutually-suspicious application stacks must coexist and use it within a single host. Additionally, a fair amount of complexity is pushed onto the application, and the total amount of CPU cycles spent in executing network protocols is not going to be any less.

Another approach to bringing I/O relief and CPU relief is to package a "super NIC", wherein a sizeable portion of the protocol stack is executed. Enter TCP Offload Engines (TOEs). Leveraging a set of tightly-coupled NPUs, FPGAs, ASICs, a TOE is capable to execute the performance-sensitive portion of the TCP FSM (in so-called partial offload mode) or the whole TCP protocol (in full offload mode) to yield CPU and memory efficiencies. With a TOE, the receipt of an individual PDU no longer requires interrupting the main CPU(s), and using I/O cycles. TOEs currently available in the marketplace exhibit remarkable speedups. Especially with TOEs in partial-offload mode, the designer must carefully characterize the overhead of falling off the hot-path (e.g., due to a packet drop), and having the CPU taking control after re-synchronizing on the PCB. There are no standard APIs to TOEs.

A third approach is to augment the protocol stack with new layers that annotate application's data with tags and/or memory offset information. Without these fixtures, a single out-of-order packet may require a huge amount of memory to be staged in anonymous memory (lots of memory at 10Gb/s rates!) while the correct sequence is being recovered. With these new meta-data in place, a receiver would aggressively steer data to its final destination (an application's buffer) without incurring copies and staging the data. This approach led to the notions of Remote Direct Data Placement (RDDP) and Remote Direct Memory Access (RDMA) (the latter exposing a read/write memory abstraction with tag and offset, possibly using the former as an enabler). The IETF has on-going activities in this space [RDDP]. The applicability of these techniques to a byte-stream protocol like TCP, and the ensuing impact on semantics and layering violations are still controversial.

Lastly, researchers are actively exploring new system architectures (not necessarily von Neumann ones) wherein CPU, memory, and networks engage in novel ways, given a defined set of operating requirements. In the case of high-capacity optical networks, for instance, the Wavelength Disk Drive [WDD] and the OptIPuter [OPTIP] are two noteworthy examples.

4.4 TCP Protocol Considerations

This section lists TCP related considerations.

4.4.1 Slow Start

Particularly when communication is done over a long distance the question arises whether the slow start mechanism of TCP is adequate for the high-throughput demand of some Grid applications. While slow start is not always necessary, some ISPs mandate it. Before thinking about using less than recent history rather than recent measurements, one should look at the Congestion Manager and TCP PCB state shearing work first!

4.4.2 Congestion Control

Congestion control is mandatory in best-effort networks. ISPs might even interrupt the service when congestion control is not performed at all.

AIMD is not the only solution to a fair, convergent control rule for congestion avoidance and control. Other solution are around - Rate based, using loss, or ECN feedback, can work to be TCP fair, but not generate the characteristic Saw Tooth.

A survey on congestion control algorithms for non-TCP based applications can be found at [TCPFRD].

4.4.3 Assumptions and errors

Most connections do not behave like the Padhye equation, but most bytes are shipped on a small number of connections, and do - c.f. Mice and Elephants.

The jury is still out on whether there are non greedy TCP flows (ones who do not have infinite sources of data at any moment).

4.4.4 Ack Clocking

Acknowledgements clock new data into the network - aside from rare (mainly only on wireless nets) ack compression, this provides a rough “conservation” law for data. It is not a viable approach for unidirectional (e.g. multicast) applications.

4.4.5 Connection setup and teardown

Any sort of remote procedure call (method invocation) requires the following basic messages to be transferred:

- Request: “please execute X (with parameters Y)”
- Response: “done, here is the result”

If such a communication utilizes TCP, at least nine messages have to go back and forth: a three-way handshake that opens the connection, followed by the actual request and response messages and four messages that are required to close the connection (assuming that messages are interspersed so that there are no extra acknowledgments required for the request and reply packets). This leads to a delay of at least two round-trip times at either end of the connection. In [TAN], this process is outlined in comparison with an experimental TCP extension entitled

"Transactional TCP (T/TCP)", where the connection setup, request, teardown and response messages are piggybacked in a way that reduces the total number of messages to three, with one round-trip time on either end of the connection. T/TCP is specified in [RFC1644] and is expected to remain experimental because of security drawbacks.

In the case of a Grid application using a remote Grid Service, the communication is typically carried out with SOAP over HTTP over TCP - the notion of a single TCP connection is lost, leading to connections being opened and closed for each consecutive call to a remote Grid Service, even if it is located on the same machine and the calls are carried out one after another. Furthermore, when utilizing a Grid Service for the first time, a Grid Service Instance must be spawned by the Grid Service factory. The aforementioned request is then preceded by the message "create a Grid Service Instance", causing two TCP connections two be set up and torn down in succession. Clearly, this is highly inefficient in a widely distributed Grid with long delays.

Two potential workarounds seem to be obvious:

- use T/TCP in a trusted environment
- reuse open TCP connections

In the context of Grid Services that were based on SOAP over HTTP, the underlying hosting environment should support persistent connections as defined in [RFC2616].

4.5 Multi-Stream File Transfers with TCP

From a performance point of view, transporting data across multiple TCP sessions is much more effective than tunneling through a single TCP session and the difference is proportional to the square of the number of TCP sessions. (see annex A)

For more details on TCP performance see for example [SIMMOD]. For ongoing work in the context of improving the TCP performance in high-speed wide area networks see for example [QUSTART, SCATCP, FASTTCP]. The key issues of [RFC2914] are related to fairness and flow granularity (and acceptable definitions thereof). For information on alternatives and variants to TCP, see [SURTRAN]. It is a survey prepared by the GGF Data Transport Research Group that has recently been merged with the GHPN RG.

4.6 Packet sizes

The performance improvements of LANs have always pushed the MTU up - since ATM LANs (remember the fore asx100) jumbo frames, i.e. 9280 byte packets, were used. A particular problem arises with global Grids, as the maximum segment size is limited by that of the weakest link. Most end-domains use 100BaseT for at least portions of their domain. Hence, it is quite unlikely to see more than only occasional the special case non 1500 byte path. However, with path MTU discovery, TCP automatically uses the appropriate MTU.

The use of Sub-IP packet sizes is an alternative consideration. Some systems (ATM) break packets into tiny little pieces, then apply various level 2 schemes to these pieces (e.g.

rate/congestion control). Most of these are anathema to good performance. More detailed information can be found at [NLANR] and [RFC1191].

5. Performance controllability

Each Grid application may have different Quality of Service requirements of the network. For example, a visualization Grid application may require high bandwidth, low latency connectivity for storage access while a computationally-intensive Grid application may require just a best-effort IP service for data movement. The Grid resource allocation algorithm may not be able to allocate the proper grid resources without having the knowledge of network services and SLS parameters available at each grid location (including the respective networking domains).

This section describes experienced issues related to the access to QoS control features.

Requirement:	1) Traffic protection 2) QoS-aware networking infrastructure
Current issue	1) API: Form of SLA with measurable parameters constituting a SLS
Available solutions	1) Overprovisionning 2) DiffServ 3) MPLS-TE with DiffServ Scheduling
Proposed alternatives:	Bandwidth on-Demand, lambda on-Demand… Overlays

5.1 Service Level Agreement (SLA)

Connectivity or data transport service between two geographically dispersed locations is usually provided by an independent third party, generically called a Service Provider (SP). The Service Level Agreement (SLA) is a contract agreed upon between the SP and the service consumer (in this case a grid subscriber) detailing the attributes of the service like connection uptime, scheduled downtime, unscheduled downtime, service provider liabilities among others. Since the SLA contains business-related parameters that are outside the scope of this document, the term Service Level Specification (SLS) [RFC3260] will be used to specify the technical qualities of the service.

5.1.1 QoS and SLS Parameters

A QoS-aware networking infrastructure would be adequate for many Grid environments. While there are many papers on QoS-mechanisms, only a few describe anything anyone has deployed. Even with a QoS-aware infrastructure deployed and working, there can be an inherent mistrust between a provider and a consumer necessitating a form of SLA with measurable parameters constituting a SLS.

QoS-parameters of particular interest typically include

- Throughput
- Delay
- Availability
- Security/integrity*
- Packet Loss*
- Diversity (another dimension to availability)* ...

* Are/May not be commonly specified

SLAs are around already despite non widespread QoS - however, SLAs are widely used only intra-ISPs (some Internet Exchanges offer SLAs but end-to-end SLAs are scarce).

5.1.2 Demanded Services

The demand of Grid applications can be addresses by the following fundamental services:

- Access to a premium service which offers low-latency communication between the two end-points. This service assures that the individual packets which were conformant to a given traffic profile (typically token/leaky bucket constrained) were transported to the destination within a given delay boundary. In addition to the classical real-time traffic, such as voice over IP or video conferencing, the Grid introduces more challenging communication demands, for example in the context of a distributed VR-environment in which the haptic is remotely driven.
- Access to a guaranteed rate/bandwidth on-demand service. This service follows the assurance of the Premium service with respect to the avoidance of packet drops, but does not have to state strong delay boundaries.

While a guaranteed rate service allows for the implementation of deadline data transfers, a less-than best-effort service, i.e. the scavenger service, is of particular interest to support high-throughput communication of single applications in order to allow for fairness among competing best-effort transfers.

Finally, these services should not only be applicable to support point-to-point communication, but also to support many-to-many overlay structures.

5.1.3 Additional Threats to QoS: Theft and Denial of Service

While QoS is in the focus for a while, traffic protection is really what people want - If I send x bps to site S, what y bps will be received, how much d later?

To guarantee y=x, while minimizing d, you need:

- **Admission Control:** assures that we are not sharing the infrastructure as we would if we adapted under congestion control

- **Scheduling:** uses prioritization of packets to avoid arbitrary queuing delay

- **Re-routing:** May also need to be controlled and pre-empted. Alternate routes (also known, unfortunately as protection paths) may be needed if we want QoS to include availability as well as throughput guarantees and delay bounds.

- **Flow Aggregation :** techniques to scale traffic management for QoS - by only managing classes of aggregates of flows, we get to reduce the state and signaling/management overhead for it. VPNs/tunnels of course are aggregation techniques, as are things that treat packet differently on subfields like DSCP, port among other methods.

Additional guidance from the Security Considerations sections will be needed in order to avoid QoS violations through attacks exploiting security loopholes in the network infrastructure.

5.2 Grids and SLS

Grids are built by user communities using resources that are typically geographically dispersed, even if they belong to the same administrative organization. Grid applications utilizing the distributed compute and storage resources depend on the underlying network connectivity provided by the transport service provider for successful and timely completion. There is a high likelihood that the remote members of a virtual organization have different transport providers for their service. It is also possible that each Grid location has different service and physical layer connectivity combinations at the network access i.e. IP over SONET leased line service, or a L2 Ethernet/Frame Relay/ATM service. All these factors lead to different SLS's at each location and can cause a Grid application to get inconsistent end-to-end Quality of Service especially in case of failures. For example, if a Grid application requiring transport level performance requires resources at a location with SLS for Layer 3 (IP) service, it has to derive through unspecified means the transport layer service equivalent to ensure compatible service levels.

It should be noted that even though a SP provides an SLS compliant service, the Grid application may not get the right level of service due to performance of network owned by an end-domain. Any end-domain within a virtual organization needs to provide similar SLS for its own internal networks in order for guaranteed end-to-end application QoS.

A common template to specify Grid SLS with measurable performance parameters related to grid applications will be needed for the grid application to work seamlessly across diverse geographical locations. The parameters of SLS can then become a great tool for grid users to measure the quality and reliability of the offered service over time.

5.2.1 SLS Assurance

Currently, the transport service provider provides the mechanisms to monitor the network and assures the user of compliance to the negotiated SLS requirements and parameters. The grid user does not have any means to independently measure and verify the SLS negotiated or determine if the network QoS needed by the application is being met at each location and thus, cannot guarantee Grid application performance. Providing mechanisms to the Grid applications to monitor network SLS parameters and have access to network alerts, errors and outages will result in better resource selection and also assure end-to-end service quality to the grid application. There are cases where the SP is not able to provide customers access to network information for SLS monitoring and assurance purposes. In that case, the SP should be able to measure and monitor end-to-end application performance and keep a real-time log accessible by customers to ensure SLS compliance.

5.2.2 On-demand SLS

One of the major values of the Grid is the ability to form virtual organizations dynamically to access the resources need for a particular application. The compute and storage resources are dynamically allocated from an available pool. For example, a compute intensive, high-energy physics application can use the majority of Grid compute resources for a few weeks and then a data intensive data-mining application, can leverage the same compute/data resources with different network requirements. Currently, the SLS's are negotiated at time of service, and do not change through the length of service contract. Providing dynamic network resources with associated dynamic SLSs will help deliver a Quality of Service based on application needs as well as provide efficient use of available network resources.

5.3 Overprovisioned networks

The challenging network requirements of Grid applications are often associated with the demand to access an overprovisioned network. In assuming network capabilities without limitations, the demand of Grid application would clearly be satisfied. However, the assumption of offering nearly unlimited bandwidth capabilities is not always true.

The costs of deploying optical networks are affected by a mixture of link and equipment costs. While link costs are typically sub-linear to the capacity, the equipment costs for beyond 2.5 Gbps interfaces are still super-linear. Further on, the amount of parallel wavelength multiplexed within a single fiber is also still limited, either caused by the limited capabilities of the existing fibre itself, or by the dimension of the optical cross connects. A network supporting hundreds of lambdas on a particular fiber is not emerging within a reasonable time scale.

On the other hand, end-systems can be expected to be attached by Gigabit Ethernet interfaces now, and 10GiGE in the near future. Also, the Grid is about to deploy applications which aim for the actual use of the available bandwidth capabilities. This leads to an environment in which the classical onion model, i.e. an increase of bandwidth capabilities when moving towards the core, is problematic. The concept of overprovisioning might therefore not scale with the deployment of Grid applications. Meshing, i.e. the use of multiple fibers, could be an economic solution to this. Here, however, one has to consider that Grid users are not really concerned about capacity, but about goodput. Mis-ordered packets must be avoided when meshing is implemented. On the other hand, mashing nicely fits to the concept of parallel file transfers introduced in section 3.3.

Finally, the challenging bandwidth demand of some Grid applications question the economic reasoning of overprovisioning, particularly when the application end-points are at varying end-points (e.g. access to a central data repository) or the usage profile varies over time (co-allocation with other resources).

5.4 QoS-Realization

Congestion control results in an approximately fair distribution of bottleneck bandwidth. While this is appropriately in a best effort network where no user paid more for getting an advanced service, a service oriented network must address this inter-class differentiation. Service differentiations at the access links are a potential solution to this. In this scenario, customers that paid less were bottlenecked at their access links in that case.

The differentiated services architecture [RFC2475] provides a framework for implementing scalable service differentiation in the existing Internet. It addresses the scalability problems of the former Integrated Services approach by an aggregation of flows to a small number of traffic classes. Packets are identified by simple markings that indicate the respective class. In the core of the network, routers do not need to determine to which flow a packet belongs, only which aggregate behavior has to be applied. Edge routers mark packets and indicate whether they are within profile or, if they are out of profile, in which case they might even be discarded at the edge router. A particular marking on a packet indicates a so-called Per Hop Behavior (PHB) that has to be applied for forwarding of the packet. The Expedited Forwarding (EF) PHB [RFC3246] is intended for building a service that offers low loss and low delay, namely a Premium Service.

While the concept of aggregate scheduling addresses the demand for scalability it causes potential problems by the varying constitution of aggregates in multi- and demultilplexing points. In using the knowledge about the composition of aggregates in each node and by actively applying traffic engineering functions such as provided by the Multi-Protocol Label Switching architecture [RFC3272] well defined service parameters can be guaranteed using deterministic queuing theory [DFNPAB].

6. Dynamic Network resource allocation and reservation

Requirements:	1) Advance reservation capabilities
Current issue	UNI 1.0 does not provide appropriate functionality
Analyzed reasons	
Available solutions	Prototype Implementation of the GFD-E.5

	Advance Reservation API
Proposed alternatives:	

6.1 Bandwidth on-demand Service Specification

Assuming that network services can be used by Grid applications to compose higher level services, the question arises whether there are particular provisioning capabilities which are of benefit. The coordinated allocation of multiple resources is a challenge. The start up of the individual service requests somehow has to be synchronized without wasting potentially scarce and thus expensive resources by an allocated service request which has to wait for the allocation of related tasks. One potential solution to this is given by the ability to reserve resources in advance. Within the Grid Resource Allocation Agreement Protocol (GRAAP) Working Group of the Global Grid Forum, the term advance reservation was defined as follows:

> An advance reservation is a possibly limited or restricted delegation of a particular resource capability over a defined time interval, obtained by the requester from the resource owner through a negotiation process.

The Optical Internetworking Forum (OIF) has published an implementation agreement for interfacing to services in optical networks. This optical User Network Interface (UNI) offers [OIFUNI] a GMPLS-compatible way to implement bandwidth on-demand services. It thus has a strong relation to the service oriented view of the Grid. However, the current UNI 1.0 version does not fully cover the functionality required by a Grid infrastructure.

6.2 Issues related to API

The GFD-E.5 Advance Reservation API document describes an experimental interface to advance reservation capabilities. The API can be considered a remote procedure call mechanism to communication with a reservation manager. A reservation manager controls reservations for a resource: it performs admission control and controls the resource to enforce the reservations. Note that the reservation manager is also responsible for performing the required clean-up action once the reservation has ended.

The GFD-E.5 document describes a C-binding of this API which allows for a uniform programming model which is capable of making and manipulating a reservation regardless of the type of the underlying resource. It thereby simplifies the programming when an application must work with multiple kinds of resources and multiple simultaneous reservations. The document defines a set of reservation related functions and their parameters. Resource specific service parameters are encoded in a particular resource description language.

A prototype implementation of this API is given by the General-purpose Architecture for Reservation and Allocation [GARA].

7. High availability

Network reliability is mandatory for Grid applications. Disruption can occur due to several reasons: congestion along the path, failed link, failed node, administrative change in the path (in MPLS cloud paths are the so called Label Switched Paths-LSP).

Requirements:	1) Network reliability 2) Efficient Routing
Current issue	IP-Restoration QoS-Routing
Analyzed reasons	
Proposed alternatives:	MPLS-TE Multipath OSPF Overlays and P2P Ipv6

Priorities for good routing system design are:

7.1 Fast Forwarding

Packet classification and switched routers have come a long way recently. While it is quite unlikely that software-based solutions were capable to beat the hardware in core routers, they can potentially compete nicely in access devices. Certainly, there is no reason why a small cluster couldn’t make a good 10Gbps router - but there is every reason why a PCI bus machine makes out at 1Gbps!

7.2 Faster Convergence

Routers and links fail. The job of routing protocols such as OSPF/IS-IS and BGP is to find the alternate paths quickly - in reality they take a whole to converge – Interior Gateway Protocols take a while (despite being mainly link state nowadays) because link failure detection is NOT obvious - sometimes the implementations have to count missed HELLO packets (since some links do not generate an explicit clock). BGP convergence is a joke, but there are smart people on the case.

7.3 Theory and practice

Most the problems with implementing routing protocols are similar to problems of classic distributed (p2p/autonomous) algorithms: dealing with bugs in other peoples implementations - it takes a good programmer about 3 months to do a full OSPF. It then takes around 3 years to put in all the defenses.

7.4 Better (multi-path, multi-metric) routing

Equal cost Multipath OSPF and QOSPF have been dreamt up - are they used a lot? In limited cases, Multipath appears to work quite well. Multimetric relies on good understanding of traffic engineering and economics, and to date, hasn't seen the light of day. Note that also, in terrestrial tier one networks, end-to-end delays are approaching transmission delays, so asking for a delay (or jitter) bound is getting fairly pointless - asking for a throughput guarantee is a good idea, but doesn't need clever routing!

7.5 MPLS

The Multi-Protocol Label Switching Architecture (MPLS) [RFC 2702] [RFC 3346] is based on a functional decomposition of forwarding and control plane.

While the experiences gained by the use of MPLS show that for level 2 protection and for provisioning of Differentiated Services based SLAs MPLS can help [DFNPAB], other experiences show, however, that some functions (e.g. Multicast) are not well supported on an MPLS-substrate.

Traffic Engineering (TE) [RFC 3272] refers to the mechanism of selecting the paths chosen by data traffic in order to facilitate efficient and reliable network operations while simultaneously optimizing network resource utilization and traffic performance. The goal of TE is to compute path from one given node to another such that the path does not violate any constraints (bandwidth/administrative requirements) and is optimal with respect to some scalar metric. Once the path is computed, TE is responsible for establishing and maintaining forwarding state along such a path.

The existing Interior Gateway Protocols (IGPs) are not adequate for TE. Routing decisions are mostly based on shortest path algorithms that generally use additive metric and do not take into account bandwidth availability or traffic characteristics.

The easiest way to provide such features would be to use an overlay model, which enables virtual topologies on top of the physical networks. The virtual topology is constructed from virtual links that appear as physical links to the routing protocol. Further, the overlay model should be able to provide constraint based routing, traffic shaping and policing functionality, survivability of the virtual links. These capabilities allow easy movement of traffic from an over subscribed link to an underused one. Multiprotocol Label Switching (MPLS) [RFC 2702] [RFC 3346] could be one of these overlay models.

One of the most appealing features of MPLS-TE (MPLS-Traffic Engineering) is the possibility to provide non disruptive traffic across the LSP. In case of outage, the upper level application will not notice service disruption (this is known as "make before break"). In such a way, it is possible to better manage the network bandwidth availability of the grid itself because of larger bandwidth availability enables grid users and developers to build more complex, high-power Grids for tasks such as image rendering. Otherwise there is the risk of either an inefficient grid, a grid that is incapable of handling its load, or of clients that start to chronically overuses their available network resources.

7.6 BGP

Policies are hard - BGP allows one to express unilateral policies (the same idea could be used for policy management of other resources like CPUs in the Grid). However, it results in difficulties in computing global choices (esp Multihoming) - there are fixes.

More information can be found at:

- http://www.potaroo.net/
- http://www.telstra.net/gih
- NANOG

See also Overlays (e.g. RON, and “underlay” routing in planetlab).

7.7 Support for Overlay Structures and P2P

Overlay structures provide a way of achieving high-performance using existing network infrastructure. Resilient overlay networks [RON] allows applications to detect and recover from path outages and other routing problems. Features like application-controlled routing, multi-path routing and QoS routing can have great impact on performance of Grid applications. Though this has promising implications, placing of overlay nodes can be a tricky problem.

Overlays and P2P (e.g. Pastry, CAN, Chord, Tapastry, etc) are becoming commonplace - the routing overlay du jour is probably RON from MIT - these (at best) are an auto-magic way of configuring a set of Tunnels (IPinIP, GRE etc). I.e. they build VPNs. In fact routing overlays may be a problem if there is more than one of them (see SIGCOMM 2003 paper on selfish routing). But there are moves afoot to provide one (e.g. see SIGCOMM paper on underlays).

P2P: are slightly different since they basically do content sharing and have index/search/replication strategies varying from mind-numbingly stupid (napster, gnutella) to very cute (CAN, Pastry). They have problems with Locality and Metrics, so are not the tool for the job for low latency file access. In trying to mitigate this, they (and overlay routing substrates) use ping and pathchar to try to find proximal nodes: c.f. limitations of Ping/Pathchar convergence when not native (errors/confidence).

More Infromation can be found at [ORAM].

7.8 IPv6

IPv6 was initially designed to solve the problem for the operational Internet, caused by the diminishing availability of address space in the 32-bit limited (and CIDR structured) IPv4 hierarchy. IPv6 has 128 bit addresses, which should be plenty, even with fairly generous allocation and structure. However it has several other benefits: firstly we can now allocate for multicast, and re-allocate for mobile, without a lot of the address collision detection machinery required for v4 (at least without having to invoke it so often) which makes dynamic addressing and group communication much more viable; a large address space obviates the need for NATs too which improves the chances of pure end-to-end connectivity without extra NAT-traversal stages; IPv6 also features a flow identifier field, which, by analogy with MPLS labels, could be used to speed-up the identification (and possible grouping) of packets into flows (and aggregates) for special purpose forwarding treatment; IPv6 has seen more enthusiasm in Europe and Japan since the burgeoning wide-area wireless provider and subscriber communities there have immediate urgent need for the address space and dynamicity.

The original motivation (lack of unicast address space) has proved to be somewhat less urgent since the advent of better aggregation, allocation policies, and DHCP widespread use, although the newer large wireless providers do not necessarily agree with this viewpoint. Estimates vary, but current best industry guesses (vis Geoff Huston at Telstra) put the end of the line at about 17 years off still.

Operationally, the zero-knowledge configurability of IPv6 is potentially very useful in large site management.

Interoperation between v6 and v4 is available in a number of ways (at least until IPv4 runs out of space, and even thereafter through 6to4 (effectively NATs).

Most host OS vendors are behind v6 (linux, bsd, Windows all have good to excellent support). The big missing piece is a stable router deployment in cores. This does not obstruct IPv6s usefulness in the edge (e.g. wireless access), but does undermine the use of flow id for forward performance where it matters (core) or multicast.

8. Security controllability

This section describes experienced issues related to security.

Requirements:	1) Site and Network security 2) On demand security
Current issue	1) that there isn't a "one size fits all" site and network security solution 2) computing overhead, packet header overhead, high-availability, and policy 3) Firewalls/NATs and Grids
Analyzed reasons	
Available solutions	Middleboxes with L4-7 impact but lead to ossification around a L4 protocol called TCP VPNs

Network security can be implemented at the link level (i.e., L2, as in WEP or Frame Relay security), at the network level (i.e., L3, as in IPsec), and at the application level (i.e., at layers above 4, like TLS). These approaches have well-known strengths and weaknesses, re-enforcing the concept that there isn't a "one size fits all" network security solution. Additionally, these approaches are not mutually exclusive. They can coexist quite nicely and can be applied incrementally, as the traffic flows from private enclaves to the public, insecure Internet. While "the more, the merrier" argument typically holds when dealing with security, there are important issues in computing overhead, packet header overhead, high-availability, and policy.

8.1 Firewalls

Firewalls pose interesting problems in Grid environments. Since Grid toolkits like Globus use non-standard ports for communication, job submission etc. configuration of both the toolkit and the firewall is required and cumbersome. Firewalls have to be configured to allow non-standard ports. To facilitate this process and avoid allowing un-wanted traffic, toolkits have to be configured to use these ports consistently. There are two parts to the firewall configuration: client-side and server-side. For example, Globus uses callbacks to call functions on the clients. This requires the firewall to be configured to allow incoming ports [GTFWALL].

On the other hand, Grid toolkits have to be developed with firewall awareness. This may involve developing trusted proxies or other methods of secure means of tunneling. Grid protocols can be made firewall aware too.

Firewalls impact network performance and pose problems for maintaining quality of service. This is due to the overhead involved in analyzing the network traffic. It places burden on the CPU and the machines can becomes a bottleneck. There is always a trade off between performance and security.

8.2 Network Address Translators

Network Address Translators pose similar problems to firewalls as described above. Callbacks to clients form servers used by Globus, for example, require specific configuration to get through NATs. The NAT needs to be configured to allow such traffic patterns as well. Maintaining servers behind a NAT is hard if not impossible. For instance, Globus security mechanisms [GTFWALL] do not allow servers to be placed behind a NAT as they need to know actual IP address.

8.3 Security Gateways

GGF's Grid Security Infrastructure (GSI) qualifies as application-level security. As any other application-level security schema, it targets true end-to-end security, thus removing the annoying problem (as well as vulnerability) of trusting network intermediaries. When properly configured, GSI is "good to go" over any network extent, regardless of its level of security.

In many a scenario, however, it is expected that local policies will dictate the use of a security gateway (e.g., an IPsec device) between private and public enclaves. Most security gateways do not discriminate between traffic that needs security versus traffic that is already secured in an end-to-end fashion. To an operator, the gateway's appeal is that it is a fixed point of transit between private and public enclaves, and its well-being can be easily audited. A GSI user can argue that the gateway needlessly adds meta-data overhead to the packet, and likely represents a bottleneck (e.g., heavy duty crypto processing) if the gateway insists in applying another layer of authentication and confidentiality. The IPsec tunnel-mode protocol (commonly used by security gateways) inserts a new IP header and a AH/ESP header, and there may be a chance that the new packet comes to exceed the link MTU (e.g., the Ethernet maximum frame size). The problem is further exacerbated by the fact that IP fragmentation is a deprecated feature (i.e., all firewalls reject IP fragments nowadays), and Path MTU discovery may fail to detect the actual MTU available.

Given that local policies are neither necessarily reasonable nor flexible, a Grid Security Infrastructure user can relax the security stipulations at her end, and, for instance, skip encrypting traffic if the security gateway is known to do so already, and she can live without confidentiality across the limited network extent between the Grid application and the security gateway. With state of the art technology, this type of reasoning cannot be automated in any way, and the GSI user is left with ad-hoc interpretation of her local policies, intervening security gateways, topologies, and the likes.

Some network gateways may attempt to compress traffic prior to its traversing a limited-bandwidth network extent. The composition of encryption and compression raises an issue of temporal dependence amongst the two. Compression is likely to yield gains when performed before encryption. Conversely, compression results in no gains and gratuitous overhead if performed after encryption. In fact, an encrypted set cannot be compressed, because the bit distribution operated by the encryption algorithm voids all known compression techniques, which thrive on regular patterns. Should data be encrypted at the GSI level, any attempt to compress data past that point will produce no benefit, and will rather add overhead; data must be compressed prior to the GSI layer. If the GSI user delegates encryption to a security gateway, then there will be solid opportunities to compress the data at the NIC level or inside the network.

8.4 Authentication and Authorization issues

Authentication (AuthN) and Authorization (AuthZ) are typically implemented as network services. That is, they reside in the network and are implemented within a consolidated locus for directories and access policies. This way, revocation of privileges, auditing, and any other management operation are particularly efficient. AAA (Authentication, Authorization, and Accounting) is a widely used denomination for this class of network services.

It is imperative that the AuthN and AuthZ services be available at all times, else the end-systems' security fixtures that depend on them will come to a screeching halt (while caching of earlier AuthN and AuthZ decisions at the end-systems level is not a good idea, in that it circumvents revocation actions that may have happened meanwhile). This availability requirement poses a burden on the server(s) implementing AAA functionality (typically a fault-tolerant cluster of servers), as well as the network paths connecting end-systems to AAA services. The latter may all of a sudden become unreachable due to slow router convergence after partial failures in the network, inadvertent SLA breaches, or outright malicious intrusion and DoS attacks underway.

The centrality of AAA services and their unexpected unavailability thus warrant the syndrome that Butler Lampson aptly described as: "A distributed system is one in which I can't get my work done because a computer I've never heard of has failed".

In GSI, the security mechanisms are accessed through an indirection layer called GSS API, which hides to the user the fact that, for instance, Kerberos is being used instead of PKI. While GSS is a sophisticated and useful programming model, there is a flip side to it in case of failures. Should the Kerberos server(s) become unreachable, the troubleshooting of the ensuing failures may turn out to be cumbersome (the Kerberos server playing the role of the computer never heard of in Lampson's citation). Whereas other systems requiring an explicit Kerberos login by a user (e.g., the Andrew distributed File System) are more amenable to track down the failure (though the failure will still be fatal until the Kerberos service comes back on line).

8.5 Policy issues

The sites forming a Virtual Organizations may very well live by different security standards. While one site has established a sophisticated certificate practice statement, at another site of the same VO the passwords are written on the back of keyboards, and private keys are unprotected. The wide variety of crypto parameters creates a host of potential pitfalls. In fact, the vast majority of security exploitations leverage the weakest policy definitions and especially their implementations. Exposure to these risks is inherent to the way Grids work, which motivates the ongoing effort in the Security Area as in the GGF GCP WG.

Security gateways enact a Layer 3 overlay (i.e., based on IP, IPsec) that suffers similar vulnerabilities. In this space, the IETF is actively working on IP-level security policies (IETF IPSP WG). It will take a while before the outcome of this work will be widely available in the security gateway marketplace.

Due to the different nature of application-level security and network-level security, the former and the latter can coexist while using entirely different mechanisms and policies. In many organizations, however, it becomes attractive for the two security approaches to share in on some of the AAA fixtures, and on the hefty costs incurred by organizations to make these fixtures work dependably (e.g., high availability, policy stipulations, certificate authorities, auditing, etc.). The implementation of the PKI infrastructure is a potential point of convergence. GSI can leverage PKI infrastructure through the GSS API, while the Internet Key Exchange (IKE) protocol can perform certificate-based peer authentication (i.e., via X.509v3) using digital signatures.

It has been noted earlier on (section 7.1) that a GSI user can delegate some of the security protection to a legacy security gateway, thus eliminating the overhead of security measures being applied twice to the same data. There is no way, however, for the GSI user to get a quantitative, objective measure of the relative strength in application-level and network-level security, when considering both security mechanisms and the policies involved. The finalization and market adoption of the outcomes of GGF GCP WG and IETF IPSP WG will go a long way towards providing a framework upon which automated evaluation tools can be built.

8.6 Middleboxes with L4-7 impact

The vision of a network agnostic to any L4-7 consideration has supported the explosive growth of IP networks over the last 15 years. The increasing relevance of security, mobility, gigabit-range throughput, streaming media, have de-facto implanted the appreciation for L4-7 issues at crucial points inside the network. The ensuing network nodes with L4-7 scope (in short, middleboxes) include: firewalls and intrusion detectors, SSL accelerators, traffic-shaping appliances, and load balancing intermediaries (often generalized as elements of a content delivery network). In more subtle ways, even the traditional L2-3 routers/switches now factor L4 considerations in the form of active queue management (e.g., RED) tailored to TCP, the dominant L4 protocol (90% of traffic over backbone extents is carried by TCP).

With middleboxes, the greater efficiencies and "hi-touch" services come with all important side-effects, which fall in two realms. Firstly, the network has built-in knowledge of some L4-7 protocols, and can show resistance to using some other L4-7 protocols, much the same way it shows resistance in upgrading from IPv4 to IPv6(as one would expect for a L3 protocol). Secondly, there is a need to discover and signal such middleboxes to select one of several pre-defined behaviors.

As a practical consequence of the first side-effect, for the foreseeable future Grid communities will have the freedom to use any L4 protocol as long as it is TCP! Let us consider the case of a Grid infrastructure interested in using the SCTP protocol [RFC2960] for its bulk data transfers. SCTP is a standard-track L4 protocol ratified by the IETF, with TCP-like built-in provisions for congestion control, and thus safe from a network perspective. This example is not fictional, in that SCTP does bring interesting elements of differentiation over TCP (e.g., datagram delineation, multi-homing, etc.), which become especially appealing at gigabit rates. Across the end-to-end path, the points of resistance to SCTP will likely show up in a) termination points (contrast with the state-of-the-art high-performance TCP's Off-load Engines, TOE, in silicon), b) intrusion detection points (where a protocol's FSM must be statefully analyzed), c) firewalls with application-proxy capability (another instance of protocol termination or splicing, see case a), and d)content delivery networks (wherein the protocol is terminated and security processing is rendered prior to steering the data, contrast with the state-of-the-art TOEs and SSL accelerators, also in silicon). All of this warrants SCTP the risk of falling off several hot-paths, not to mention clearing all the security checkpoints along the way.

But there is more to it than just ossification around a L4 protocol called TCP. The TCP operating requirements are practically limited to using fixed destination port numbers, because firewalls and intrusion detection devices have fundamental troubles coping with dynamic ports usage (the H.323 circles first learned this lesson, the hard way). In fact, many a community resorted to the extreme point of sanctioning that their destination port number be port 80, regardless of their higher-level protocols and applications, thus de-facto voiding the very value of firewalls and intrusion detection.

As said for the second class of side-effects, an application will likely need to discover and signal "middleboxes" in order to access the QoS and security behaviors of choice. Without signaling from the application, the middlebox may even dispose of the soft-state associated with the application, and reuse the resources for other applications (this is a typical syndrome with firewalls and "silent" long-lasting TCP connections). Unfortunately, this is an area still showing a wide variety of plays. The wire protocol can be in-band (e.g., SOCKS) or out-of-band (e.g., RSVP). Furthermore, the programming model can be structured around APIs, or require a point-and-click GUI session, or a command-line-interface (CLI) script.

The common case of long-lasting TCP connections traversing one or more middleboxes is worth a special mention. It has been observed that the intervals without traffic may result in a loss of the soft-state at the middleboxes (even though the TCP flow is alive and well). To avoid this, Grid developers are often tempted to use the KEEPALIVE option in the TCP protocol (accessible through a "setsocketopt()" system call in common OSs). It must be noted that KEEPALIVE is a frowned upon option in TCP. In fact, no RFC mandates its implementation. [RFC1122] discusses its implications (while acknowledging that popular implementations went off coding it as a "premium" feature a long time ago). Developers are encouraged to build their liveness handshakes (if any) into the protocol(s) above TCP, resulting in more accurate liveness reports on the actual endpoints.

SOCKS [RFC1928] is an attempt to standardize the exchange between application and firewall, though the market adoption and degree of confidence on the overall security solution are spotty at best. This fragmented and still immature solution space does not help Grid users who, among others, would certainly benefit from a comprehensive, unified style of interaction with middleboxes.

Standard signaling protocols are expected to come from the IETF MIDCOM working group [MIDCOM]and the NSIS working group [NSIS] at the IETF (though the actual APIs are out of their scope).

8.7 VPNs

With a Virtual Private Network, a user has the experience of using dedicated, secure links of various reach (LAN, MAN, or WAN), even though in reality the actual network is built out of Metro/WAN network extents over public, insecure networks (such as the Internet). VPNs are known to scale quite well, from the consumer market (e.g., telecommuters using VPNs across WiFi and the Internet) to the large enterprise market (e.g., for branch-office to headquarters communication). A popular VPN choice is to use the L2TP protocol in conjunction with the IPsec protocol [RFC2401] and the Internet Key Exchange (IKE) protocol [RFC2409]. In a VPN, either the ingress point, or the egress point, or both can have portable, pure-software implementations, or come in appliance-style embedded setups.

Once a VPN is established, the VPN is meant to be entirely transparent to the user. As such, Grid applications will typically continue to use security fixtures of their own, in an end-to-end fashion, and the existence of an underlying VPN covering a portion of the end-to-end extent goes totally un-noticed. There are, however, two important exceptions.

VPN protocols have provisions for periodically renegotiating new keying material, so as to maintain the integrity of the VPN for a very long time (possibly indefinitely). In practice, however, local security protocols must require users to periodically re-instate their credentials into the VPN console, to take into account changes in personnel's authorization. This added burden can be irksome to many a Grid user, especially when there are long-running tasks at stake, and the VPN is provisioned via an appliance that can only be operated via point-and-click sessions or command-line-interface scripts (as the vast majority of appliances are, today). Furthermore, a Grid application can employ N VPNs at once, each subject to different administration policy. These situations are clearly vulnerable to operator errors, given that application and VPN console(s) are totally disjoint.

When a VPN has a fatal error, the application will discover it the hard way, with traffic coming to a screeching halt, and retransmission attempts going off periodically. Whenever the application and VPN console are disjoint, there is no way for the application to restart the VPN, or signal a 3rd party to do so.

It would be nice if the Grid application could access the VPN console, re-affirm credentials, and register for notifications through an API like the GSS or the Advance Reservation API listed above.

9. Multicast

The ever growing needs for computation power and accesses to critical resources have launched in a very short time a large number of grid projects. The very basic nature of a grid is to allow a large community of people to share information and resources across a network infrastructure. Most of the grid usages nowadays consist in (i) database accesses, sharing and replication (DataGrid, Encyclopedia of Life Science), (ii) distributed data mining (seti@home for instance) and, (iii) data and code transfers for massively parallel job submissions. For the moment, most of these applications imply a rather small number of participants and it is not clear whether there is a real need for very large groups of users. However, even with a small number of participants, the amount of data to be exchanged can be so huge that the time to complete the transfers can rapidly become unmanageable! More complex, fine-grained applications could have complex message exchange patterns such as collective operations and synchronization barriers.

Multicast [DEE88] is the process of sending every single packet from the source to multiple destinations in the same logical multicast group. Since most of communications occurring on a grid imply many participants that can be geographically spread over the entire planet, these data transfers could be gracefully and efficiently handled by multicast protocols provided that these protocols are well-designed to suit the grid requirements. Motivations behind multicast are to handle one-to-many communications in a wide-area network with the lowest network and end-system overheads while achieving scalability.

Requirements:	1) Reliability 2) Low recovery latency 3) Efficient congestion control

Current issue	Limited deployment No real standard no ‘one solution fits all’
Analyzed reasons	
Available solutions	
Proposed alternatives:	End-system/end-host multicast Overlays, P2P

In contrast to best-effort multicast, that typically tolerates some data losses and is more suited for real-time audio or video for instance, reliable multicast [SRELMUL] requires that all packets are safely delivered to the destinations. Desirable features of reliable multicast include, in addition to reliability, low end-to-end delays, high throughput and scalability. These characteristics fit perfectly the need of the grid computing and distributed computing communities. Embedding a reliable multicast support in a grid infrastructure would not only optimize the network resources in term of bandwidth saving, but also increase both performances for applications, and interactivity for end-users, thus bringing the usage of grids to a higher level than it is at the moment (mainly batch job submission).

Here is some necessary background on main multicasting protocols and mechanisms in IP networks. Internet Group Management Protocol (IGMP) is used by hosts to join or leave a multicast group. RFC 3376 describes IGMPv3. As regards multicast forwarding algorithms, there are two main families of algorithms: reverse path forwarding (RPF) and center-based tree (CBT). The former yields two advantages because of fastest delivery of multicast data and different tree creation for different source node resulting in more efficient utilization of network resources. The latter utilizes another method to calculate optimum paths and its main disadvantage consists in creating suboptimal path for some sources and receivers.

Based on these two main algorithms, there were developed several multicast routing protocols as Distance Vector Multicast Routing Protocol (DVMRP), Multicast OSPF (MOSPF) and Protocol Independent Multicast (PIM). DVMRP was initially defined in RFC 1075 and it uses RPF algorithm. Multicast Extensions to OSPF (MOSPF) is defined in RFC 1584. It is not a separate multicast routing protocol as DVMRP. This protocol forwards datagrams using RPF algorithm and it does not support any tunneling mechanism. Unlike MOSPF, PIM is independent of any underlying unicast routing protocol and has two different ways of operation: dense mode (PIM-DM) and sparse mode (PIM-SM) defined in RFC 2362. The former implements the RPF algorithm. The latter uses a variant of CBT algorithm. PIM-DM should be used in contexts where the major part of hosts inside a domain needs multicast data but also in contexts where senders and receivers are relatively close, there are few senders and many receivers, multicast traffic is heavy and/or constant. PIM-DM does not support tunnels as well. One of the main benefits of PIM-SM is the capability to reduce the amount of traffic injected into the network because of multicast data are filtered from network segments unless a downstream node requires them. Furthermore pruning information is maintained only in equipments connected to the multicast delivery tree. PIM-SM is well suited for those situations in which there are a large number of multicast data streams flowing towards a small number of the LAN segments and also in those environments in which there are few receivers in a multicast group or when senders and receivers are connected through WAN links or the stream is intermittent.

With regard to inter-domain routing, there are two approaches to multicast domains interconnection: Multicast Source Discovery Protocol (MSDP) and Border Gateway Multicast Protocol (BGMP). They both are not currently IETF standards.

Nowadays, MBONE is still operational but multicast connectivity is natively included in many Internet routers. This trend is growing and will eliminate the need for multicast tunnels. Current MBONE environment is only a temporary solution and will be obsolete when multicasting is fully supported in every Internet router.

Recently, development of multicast systems has accelerated thanks to new and improved applications such as many grid applications: teleimmersion, data distribution, gaming and simulation, real-time data multicast. Many of these applications use UDP instead of usual TCP because of reliability and flow control mechanisms have not been optimized for real-time broadcasting of multimedia data. In some contexts, it is preferred to loose few packets instead of having additional TCP delays. In addition to UDP, many applications use Real-Time Transport Protocol (RTP).

Another open issue is concerned with multicast security, that is, securing group communications over the Internet. Initial efforts are focused on scalable solutions with respect to environments in which there are a single sender and many recipients. Initially, about multicast data delivery, IP-layer multicast routing protocols are principally considered (with or without reliability) such as those exposed before. Typically, each group has its own trusted entity (Group Controller) that manages security policy and handles group membership. Some minimal requirements are group members' admission and source/contents authentication; DoS attacks protection is desirable as well. Considering that many applications fall in one to many multicast category, each one with its own requirements, it is not a feasible way to think of a "one size fits all" solution. So it is going to define a general framework characterized by three functional building blocks: data security transforms, group key management and group security association, group policy management. With regard to large multicast groups, see for instance [MSEC]. Actually, there are no standards. Some working groups inside IETF and IRTF are actively working on this very crucial topic.

Besides the routing layer discussed previously, multicast at the transport layer mainly provides the reliable features needed by a number of applications. Many proposals have been made during these past 15 years and early protocols made usage of complex exchanges of feedback messages (ACK or NACK) [XTP95][FLO97][PAU97][YAV95]. These protocols usually take the end-to-end solution to perform loss recoveries and usually do not scale well to a large number of receivers due to the ACK/NACK implosion problem at the source. With local recovery mechanism, the retransmission of a lost packet can be performed by any receiver in the neighborhood (SRM) or by a designated receiver in a hierarchical structure (RMTP, TMTP, LMS [PAP98], PGM [GEM03]). All of the above schemes do not provide exact solutions to all the loss recovery problems. This is mainly due to the lack of topology information at the end hosts and scalability and fairness with TCP still remain open issues.

Given the nature of the information exchanged on a grid, reliable multicast is the best candidate for providing an efficient multipoint communication support for grid applications. The objectives are ambitious: extending the current grid capabilities for supporting fully distributed or interactive applications (MPI, DIS, HLA, remote visualization...). With the appropriate reliable multicast facilities, grid infrastructures would be more efficient to handle a larger range of applications.

There are however a number of factors that seriously limit the availability of multicast on large scale networks such as the Internet or a grid infrastructure. Some are technical, others are more politic.

If we consider a dedicated grid infrastructure with all participants and ISPs willing to move forward (unfortunately this is not the case), then issues related to inter-domain routing, security or firewalls could be fixed quite easily with the current tools and protocols (MBGP and MSDP for inter-domain routing and for controlling sources for instance, PIM-SSM for security), especially when the size of the group is not very large. What's left is the core problem of reliable multicast: how to achieve scalability of recovery schemes and performances? As stated previously in the brief background, there is no unique solution for providing multicast facilities on an internetwork: end-to-end, with local recoveries, with router assistance... To this long list, should be added the alternative solutions to IP multicast based on overlays and host-based multicast that scale quite well up to some hundreds of receivers [STO00][HUI00][SAY03]. In this context, it seems very reasonable to consider all possibilities and to have specific solutions for specific problems. One example could be to have an overlay-based multicast for small groups of computing sites and a fully IP multicast scheme for larger groups. The main difficulties are then to provide a multicast support for high throughput (job and data transfers) and low latency (for distributed/interactive applications).

Regarding how the multicast support should be presented to the user or the application, there are several design choices that we believe can coexist (and are fully complementary): a separate program 'a la' ftp or a separate library to be linked with the application or a fully integrated solution with high interaction with the grid middleware, this last solution being the more transparent one for the end-user, but also the most difficult to achieve.

While Tier 1 multicast routing works (most ISPs run core native multicast), Inter-domain routing is rather problematic (it is getting better...MSDP Problems, App Relay Solutions). Similarly, there are some candidate protocols for reliable multicast, but nothing exists that is as solid as TCP in 1988.

Address Allocation and Directories are not great yet, hence beacons and so on.

- Access Networks are in bad shape...e.g.

- DSLAMs don’t do IGMP snooping

- Cable don’t do IGMP snooping
- Dialup can’t hack it at all

10. Sensor Networks and wireless technologies

Sensors are a vital part of the Grid because they allow integration of real-time data into grid applications.

Requirements:	Integration of wireless sensors networks
Current issue	power limited very dynamic topology unreliable communications IP overhead TCP limits over wireless networks
Available solutions	
Proposed alternatives	

Some sensors will be complex, and wired directly into the Grid, using standard Grid protocols [GAYNOR]. An XML-based language for In-situ and remote sensors can be found at [Bo03]. However, other sensors will be embedded devices without access to wired power, and wired Internet service. In this draft, we focus on the small, embedded version. These wireless sensors have great potential to provide vast data to grid users, but there are many challenges before this can happen. These small sensors are power limited because they are battery powered, sometimes they are unconnected from the Internet, their topology is very dynamic and the communications between sensors and their base stations is often un-reliable.

There are many wireless technologies that these sensors can use: WiFi, Cellular, Bluetooth, passive (i.e. Rfid tags) free space lasers, and many other ideas. The emerging low level 802.15 [802.15] standard and zigbee [zigbee] which sits above is useful for low-powered sensors and is being adopted by some vendors. There are also many classes of processors connected to the sensor, or the sensed object. Rfid tags have no processing power, and don't need external power. These passive tags can transmit simple data a short distance. Mote technology [XBOW] such as smart dust [PKB99] has simple processors with efficient non-IP communication stacks, other sensors (such as those developed by Deborah Estrin's group at UCLA [CENS]) use iPaq PDA's with 802.11 wireless that run Linux along with a full TCP/IP stack. These many options cause a complex trade-off between CPU cycles, and number of bits transmitted. For example, with smart dust technology this trade-off between CPU cycles, and bits transmitted is discussed in [HILL].

One attribute of sensor networks will be the dynamic nature of their connectivity to the Internet. Remote sensors might rely on a satellite passing overhead before they can transmit their data. This on/off connection is not well suited to many Internet protocols. Current research at Intel Labs [FK03] is addressing how applications can access data from sensor networks over long-distance unreliable links.

Another problem with sensor networks can be robust communications between sensors, and between sensors and their base stations. Traditional MAC layer protocols such as Ethernet are too expensive for sensors such as Smartdust. The thin MAC protocol used by Smartdust does not have reliability. Robust and efficient communication is another area of current research.

Power management to extend battery life is a prime consideration in many sensor applications. Power management defines many characteristics of a sensor node: how often does it listen for incoming messages, how often does it wake up to send a message, how are CPU cycles traded for number of bits transmitted, addressing scheme, and protocols at the MAC and network layers.

For many applications IP at the network layer is too inefficient both in terms of power and bandwidth. The high overhead of IP makes no sense for many applications of technology such as "Smartdust"[PKB99] where longevity of batter life is important. Motes, the open version of smart dust runs a barebones operating system (TinyOS [TOS]) and communications stack optimized to save the number of bytes transmitted. The 20 bytes of the IP header requires too much power and to much bandwidth for many applications that Smartdust is targeted at.

Having a non-IP networking layer in the wireless sensor network is a major complication, but current research [CHMRW03] has proposed solutions. As long as you have dual stack host and translation software these non-IP sensors can become part of a virtual wireless grid because applications can have loosely coupled, yet fine grained access to sensor data (or access to the gateway node that has access to the sensor data).

While within a particular sensor realm each sensor will have a unique address, in the global sense this is not likely. For end-to-end applications a mapping is needed to allow applications to access (and identify) sensor data. It is not clear what end-2-end connectivity means in the sensor world. Access to individual l sensors may not scale, but access to data from each sensor via architecture such as Hourglass (a proposed system sensor infrastructure from Harvard) and others [TDB][COUGAR][NDKG02] scale better, and addresses many of the unique problems of providing applications with fine-grained access to sensor data.

Routing within sensor nets is hard because of limited power, the dynamic nature of sensor networks, and their non-unique addressing across sensor realms. Sensor networks may only have unique addressing within a region (similar to the NAT architecture). The low bandwidth and power makes traditional Internet routing protocols (such as RIP, and OSPF) to expensive. Sensor nets also have the potential to be much more dynamic than the current Internet. We expect sensors to be on people, cars, and other moving objects. One open research questions is how to route in such a constrained and dynamic environment. Routing strategies such as GPRS address non-unique address between in different sensor regions. Other routing algorithms such as data-centric [KEW02] present a framework based on data aggregation rather than routing based on shortest path. Geographic [KA01] routing might work well when sensors have built in GPS systems. The point is: routing within a region of a sensor network is specialized, and highly dependent on the application.

10.1 TCP and Wireless Networks

TCP was designed for reliable delivery of data across various network paths. However, TCP algorithms developed during the last two decades are mostly empirical and based on assumptions that hold in wired networks [Jac88], but not necessarily in wireless environment. One of the assumptions is that transmission channels only incur low bit error rate (BER). As a result, the TCP flow control mechanisms were developed assuming that packet loss occurs primarily due to congestion somewhere in the network. In wireless links the assumption does not hold, some TCP algorithms may not work as intended and the performance of TCP is nowhere near as efficient on wired networks [KY02].

Wireless links are inherently unreliable, BER is significantly higher (3 to 4 order of magnitude) than in wired links [Pen00]. Random packet losses occur frequently. They can be caused by a number of environmental features, such as fading (fluctuation of signal strength) due to obstructions, atmospheric conditions and interferences. When a group of such losses occur in quick succession, TCP attributes such losses to congestion control, rather than corruption, and so incorrectly invokes the congestion control mechanisms. As a result, the bandwidth may be under utilized for no good reason.

User mobility is a unique feature of wireless networks. As a user moves around, s/he needs to maintain their connections to the nearest base station. When the user moves out of range of the base station, its connection has to be handed over to another base station. During a handover, packets can often be delayed or lost. Again, TCP cannot distinguish between losses due to such handovers and due to congestion and random losses [CI95]. TCP conservatively exercises slow start congestion control mechanism, causing further waste in transmission bandwidth.

Some other factors that affect TCP performance over wireless networks include: limited bandwidth, unfairness due long round trip times, and interactions with other protocols.

Present wired technologies support much higher data rates than wireless ones. While the low bandwidth itself isn't really a problem, many of the error correction and loss detection mechanisms employed by wireless technologies can use up a substantial portion of this bandwidth, leaving precious little for user data.

Wireless links exhibit longer round-trip time than wired links. It is well-known fact that [XPMS01] TCP is biased against long round-trip time connections; hence these connections will end up with a smaller share of the link bandwidth (it takes longer time to update its sending window to its optimum rate)

Link layer protocols implementing their own error recovery may interact adversely with TCP. If the link protocol attempts to retransmit packet invisibly to TCP, it may takes too long, and subsequently there may end up being multiple simultaneous transmissions of the same packet, which is a huge waste of bandwidth if it happens regularly.

Various attempts in solving the problems by researchers in the area [XPMS01, BPSK77] are presented below.

Modification of TCP: TCP SACK [RFC2018] is a technique to recover from multiple packet losses in the sender window by using Selective Acknowledgement option. The SACK option allows receivers to additionally report non-sequential data they have received, and the sender subsequently retransmits all known lost packets. Currently TCP SACK is implemented in such operating systems such as window 2000m but it has not been conclusively proven whether the increased memory needs and power consumption make it worthwhile implementing in a wireless situation. Solution such as Indirect TCP (I-TCP) [BB95] suggests that a base station actually maintains two separate types of connections, between the wireless connection on one side, and the wired connection on the other, and two separate protocols on these connections (TCP on the wired, something else on the other). However, this approach does not ensure end-to-end delivery of packets and does not work well if the connection is split several times over the course of the connection.

Another approach is to modify TCP to provide an Explicit Loss Notification, to alert the sender when an error is detected in the wireless network. However, there have been no efficient solutions to date [PMS00].

Replacement of transport protocol: There are several possible transport protocols that could replace TCP in development, such as Wireless Transmission Control Protocol (WTCP) and Wave and Wait Protocols (WWP) [Pen00]. These may perform better than TCP over wireless links, however, as suggested earlier, it is not entirely practical to try and bring these in on a global scale.

New wireless link protocol: Link layer protocol that runs on top of the Physical Layer that has immediate knowledge of dropped frames and thus can respond faster than higher-level protocols. It can take over the task of ensuring the reliable delivery of packets that are lost due to errors, effectively hiding these losses from TCP and avoiding congestion control measures [LRK099].

Similarly, packets lost during handover should be quickly retransmitted as soon as a connection is made with the next base station. Unfortunately, it is not easy to design such a protocol that can operate under different environment [PG99][CLM96]. These protocols can use a considerable proportion of the bandwidth under high error circumstances, as well as requiring extra processing time and power usage, all of which tend to be precious commodities in wireless devices. Snoop [BSK95] and TULIP [PG99] are examples of such an approach.

Network modifications: General Packet Radio Service (GPRS – a 2.5 Generation wireless networks) extends the data capabilities of the Global System for Mobile phone (GSM) networks from Packet Data on Signalling-channel Service (PDSS) to higher data rates and longer messages by introducing packet switching and IP to mobile networks. It manages data packets switching in a more efficient manner than existing GSM networks, allowing for a much higher data rate than at present- up to 171.2 kbps (theoretically) vs 9.6 kbps. However, further experiments are required to determine its benefits.

In general, researchers still have not been able to establish a way in which TCP can run over wireless links with the same efficiency as it currently runs over wired links. Wireless links still run at lower speeds and higher error rates, resulting in low data throughputs. This makes TCP a very poor wireless solution. If wireless speeds increase, TCP would be quite suitable as a wireless protocol. The behaviour of TCP over 2.5/3G (Generation) wireless links is largely unknown. Currently, there is work in progress in the Internet Engineering Task Force (IETF) community to specify TCP configuration for data transmission over 2.5G and 3G wireless networks. It seems that the most promising solution to creating an efficient path for TCP over wireless networks would be a combination of an efficient link layer protocol in addition to a clever variant on TCP. A cross-layer optimization design may be appropriate in this situation. An efficient link layer protocol capable of hiding all data losses due to errors, handovers, etc., from TCP would be of great use. A variant of TCP capable of avoiding the inefficiencies over wireless would be valuable. Part of this could be the development of an effective form of Explicit Loss Notification.

Transport Error Notification (ETEN) for Error-Prone Wireless and Satellite Networks: TCP was designed for channels with low bit error rate (BER) and as a result, the TCP flow control mechanisms assume that every packet loss is due to network congestion (router buffer overflow) rather than packet corruption. When TCP detects a packet loss, it always exercises its congestion control to reduce the flow’s transmission rate to protect network stability. This assumption does not hold in wireless and satellite networks where error rates are non-negligible. Consequently, the TCP performance is significantly reduced in these wireless environments when its congestion control reduces its transmission rate unnecessarily on packet corruption.

If a TCP sender can distinguish packet loss due to congestion from packet loss due to corruption, better performance may be achieved.

Several approaches have been proposed to distinguish congestion losses from errors. Implicit approaches to distinguish error from congestion have not been successful [BV98, DMKMV01]. Explicit congestion notification (ECN) approaches [RFC3168] can be used to infer losses due to corrupted packets, however, the inferences are mainly heuristic and hence it is difficult to develop effective methods to deal with corruption.

Explicit Transport Error Notification (ETEN) mechanisms [KAPS02] are proposed to improve the throughput of TCP over error-prone wireless and satellite networks.

These mechanisms assume that either a) sufficient information can be recovered from the corrupted packet or b) only cumulative error rates for each link can be calculated.

With (a), the corrupted packet can be related to its flow and hence appropriate and effective action can be taken. With (b), it is still difficult to ascertain congestion losses and corruption losses and hence these mechanisms are less effective.

Analogous to ATM backward explicit congestion notification (BECN) and to ATM forward explicit congestion notification (FECN), notification of corrupted packets can be sent directly back to the sender (as in Forward ETEN mechanism) or proceed to the destination to be returned to the sender (as in Backward ETEN mechanism).

Notifications can be sent out-of-band by generating a new signaling message and send it back to the sender or piggy-backed on the uncorrupted packet of the corresponding flow.

ETEN mechanisms can operate on a per-packet or on a per-aggregate basis. Per-packet ETEN offers substantial gains in TCP goodput in the absence of congestion, but is insignificant in the presence of congestion. Implementation-wise, it is impractical since it requires intermediate nodes to reliably extract information from the headers of corrupted packets. Corrupted packets are often discarded by the link layer before they even reach the IP layer.

Cumulative ETEN (CETEN) appears more attractive to implementation but its effectiveness depends on how a cumulative indication measure is chosen and evaluated and on how accurate a corruption loss can be distinguished from a congestion loss. Early version of CETEN did not realize the potential gains promised by per-packet schemes [KAPS02].

A later and promising version of CETEN was reported in [5]. In this version, CETEN provides the TCP sender a corruption rate statistics to drive the behaviour of the congestion control algorithms.

The CETEN requires TCP protocol modifications. It adds "corruption survival-probability" field to each packet. This value represents the probability that a packet avoids corruption as it traverses the network path. The survival probability field is set to 1.0 by the source of the packet and is updated by intermediate nodes along the path. The transport endpoint at the destination keeps a record of the survival probability of the forwarding path and echoes the probability back to the sender in the next ACK packet transmitted.

Two variations of CTEN were proposed: $CETEN_P$ and $CETEN_A$. In $CETEN_P$, a coin is flipped based on the estimated fraction of the losses due to corruption over the total losses to decide whether the lost is due to corruption or congestion. If the loss is attributed to corruption the lost segment is retransmitted without modification to the congestion state.

In $CETEN_A$, the TCP's multiplicative decrease factor (MDF) is modified to vary between ½ and 1. If all loss is due to congestion the standard MDF of ½ is used (cwnd is reduced by half). However, if all loss is due to packet corruption an MDF of 1 is used (i.e., no cwnd reduction).

Preliminary simulation results showed that the combination of TCP SACK and CETEN is a promising approach in mitigating the problems corruption-based losses pose to TCP performance in wireless and satellite networks. The authors of CETEN claimed an order of magnitude increase in TCP goodput over stock TCP SACK over a range of packet error rates, however, it was not clear how the results were related to the bit error rates. The results [KSEPA04] showed that ETEN mechanisms are useful within a certain BER range. When the BER rate is low, TCP SACK alone can handle corruption well without CETEN. When the BER is high, the retransmission time out (RTO) becomes prevalent and CETEN is no longer effective. However, within this useful BER range, many issues remain to be resolved before CETEN can be deployed.

Issues to be considered:

Feedback information. The issue is how to derive corruption-survival probabilities and whether these probabilities are appropriate measure for corruption control. Even if they are relevant, it is still difficult to determine their application scope in terms of flow types (long vs reactive) and time scale. In the simulation presented, these values were configured and tracked over time. It is difficult to judge whether this reflects well the real conditions.

Scalability. Additional processing that has to be done at each router along each TCP flow and this would render the scheme unscalable.

Interlayer interaction. The TCP corruption loss notification requires the cooperation of lower layer protocols where survival probabilities can best measured and calculated. This violates the traditional protocol layered architecture.

Protocol modification. The CETEN requires an additional field within the TCP header to carry the corruption probabilities between a TCP source and its destination. This may make it difficult for CETEN to be adopted widely.

Security. Some receivers may send bogus corruption report in an attempt to game congestion control. Security measures have to be developed to handle this issue.

In general, an effective congestion control takes appropriate action to mitigate the "congested condition" as well as to recover losses by retransmitting packet. Current CETEN corruption control scheme retransmits lost packet but does nothing to mitigate the "error condition" of the network, additional scheme such as an adaptive FEC may be useful. Adaptive FEC may apply different forward error correction schemes depending on the error rate of the channel.

In light of the above discussion, it may be useful to consider a cross-layer design between the link layer and the transport layer where cross-layer optimization can be performed. The link layer is appropriate and effective in handling most error conditions;

however, it can pass relevant information (both error and congestion) to the transport layer to handle end-to-end issues.

10.2 Mobile and Congestion Control

Mobile nodes experience temporary indications of loss and congestion during a hand-off. People have proposed mechanisms for indicating whether these are “true” or chimera.

11. Grid Traffic

Requirements:	Understand the nature of the Grid traffic and how it may impact the network and the protocols
Current issue	Internet Traffic is self similar, what’s about Grid traffic? Traffic Phase Effects Flash Crowds
Available solutions	Network monitoring
Proposed alternatives	

11.1 Observed Traffic

Observations (see many IMW papers) are that traffic is currently mainly made up of mice (small, slow) flows and elephants (large, fast, long) flows at the individual 5-tuple level, and at the POP aggregate level.

Traffic itself is self similar, i.e. arrivals are not i.i.d. However, this doesn’t actually matter much (there is a horizon effect)

Traffic Phase Effects: p2p (IP router, multiparty applications etc) have a tendency (like clocks on a wooden door, or fireflies in the Mekong delta) to synchronize. This can result into several problems.

11.2 Flash Crowds

Flash crowds might be caused by scenarios such as genome publication of new result followed by simultaneous dbase search with similar queries from lots of different places.

11.3 Asymmetry

Many things in the net are asymmetric - see ADSL lines, see client-server, master-slave, see most NAT boxes. See BGP paths. Note: assumptions about symmetry (e.g. deriving 1 way delay from RTT) are often wildly wrong. Asymmetry also breaks all kinds of middle box snooping behavior.

12. Acknowledgments

The authors would like to thank Matt Welsh and Dario Vlah from Harvard University for helpful comments.

13. Authors Information

Volker Sander (Editor)
Forschungszentrum Jülich GmbH
v.sander@fz-juelich.de

William Allcock
Argonne National Lab.
allcock@mcs.anl.gov

Pham CongDuc
Ecole Normale Superieure Lyon
Congduc.Pham@ens-lyon.fr

Inder Monga
Nortel Networks
imonga@nortelnetworks.com

Pradeep Padala
University of Florida
ppadala@cise.ufl.edu

Marco Tana
University of Lecce
marco.tana@unile.it

Franco Travostino
Nortel Networks
travos@nortelnetworks.com

Jon Crowcroft
University of Cambridge
Jon.Crowcroft@cl.cam.ac.uk

Mark Gaynor
Boston University
mgaynor@bu.edu

Doan B. Hoang
Department of Computer Systems
Faculty of Information Technology
University of Technology, Sydney,
dhoang@it.uts.edu.au

Pascale Vicat-Blanc Primet
INRIA
Pascale.Primet@inria.fr

Michael Welzl
University of Innsbruck
michael.welzl@uibk.ac.at

14. References

High availability (routing)

[RFC 3272] Overview and Principles of Internet Traffic Engineering. D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, X. Xiao. May 2002

[RFC 2702] Requirements for Traffic Engineering Over MPLS. D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManus. September 1999

[RFC 3346] Applicability Statement for Traffic Engineering with MPLS. J. Boyle, V. Gill, A. Hannan, D. Cooper, D. Awduche, B. Christian, W.S. Lai. August 2002

Security controllability

[RFC2401] Kent, S. and Atkinson, R., "Security Architecture for the Internet Protocol", RFC2401, November 1998

[RFC2409] Harkins, D. and Carrel, D., "The Internet Key Exchange (IKE)", RFC2409, November 1998

[RFC2960] Stewart, R., et.al., "Stream Control Transmission Protocol", RFC2960, October 2000

[GTFWALL] Globus Toolkit Firewall Requirements http://www.globus.org/Security/v2.0/Globus%20Firewall%20Requirements-0.3.pdf

[RFC1122] R. Braden, Requirements for Internet Hosts -- Communication Layers, RFC1122, October 1989

Multicast

[SRELMUL] Scalable Reliable Multicasting projects http://dsonline.computer.org/middleware/MWprojects.htm#scalable

[DEE88] S. E. Deering and D. R. Cheriton. Multicast Routing in Datagram Internetworks and Extended LANs. In ACM SIGCOMM'88 and ACM Trans. on Comp. Sys., Vol. 8, No. 2

[GEM03] Jim Gemmell and al. The PGM Reliable Multicast Protocol. In IEEET TNetworks. Multicasting: An Enabling Technology. (previously an internet draft: T.~Speakman et~al. Pgm reliable transport protocol specification. internet draft, 1998)

[STO00] Ion Stoica, T. S. Eugene Ng and Hui Zhang. REUNITE: A Recursive Unicast Approach to Multicast. In IEEE INFOCOM 2000

[HUI00] Yang-hua Chu, Sanjay G. Rao, and Hui Zhang. A Case forEnd-

System Multicast. In ACM SIGMETRICS 2000

[SAY03] A. El-Sayed, V. Roca and L. Mathy. A survey of Proposals for an Alternative Group Communication Service. In IEEE Network magazine, Special issue on ``Multicasting: An Enabling Technology',January/February 2003

[YAY95] R. Yavatkar, James Griffoen, and Madhu Sudan. A reliable dissemination protocol for interactive collaborative applications. In ACM Multimedia, 1995

[FLO97] Sally Floyd, Van Jacobson, Ching-Gung Liu, Steven McCanne, and Lixia Zhang. A reliable multicast framework for light-weight sessions and application level framing. In IEEE ACM Transactions on Networking, 5(6), 1997

[PAP98] Christos Papadopoulos, Guru M. Parulkar, and George Varghese. An error control scheme for large-scale multicast applications. In Proc. of the IEEE INFOCOM, March 1998

[PAU97] S. Paul and K. Sabnani. Reliable multicast transport protocol (RMTP). In IEEE JSAC, Spec. Issue on Network Support for Multipoint Communications, 15(3), April 1997

[XTP95] XTP Forum. Xpress Transport Protocol Specification, March 1995

[MSEC] "The Multicast Security Architecture",draft-ietf-msec-arch-01.txt

[BB95] A. Bakre and B. R. Badrinath, "I-TCP: Indirect TCP for mobile hosts, Proceedings of the 15th Int. Conf. Distributed Computing Syst., May 1995.

[Bo03] Botts, MIke, Sensor Model Language (SensorML), XML-Based Language for In-situ and Remote Sensors, NIST workshop on Data Exchange Standards at the Jobsite, May 29, 2003, http://www.bfrl.nist.gov/861/CMAG/CMAG workshop/botts.pdf

[BPSK77] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz, "A Comparison of Mechanisms for Improving TCP Performance over Wireless Links," IEEE/ACM Transactions on Networking, Vol. 5, No. 6, Dec. 1977, pp. 756-769

[BSK99] H. Balakrisnan, S. Seshan, and R. H. Katz, "Improving reliable transport and handoff performance in cellular wireless networks," ACM Wireless Networks, vol. 1, Dec. 1995

[BV98] S. Biaz, and N.H. Vaidya, "Distinguishing Congestion Losses from Wireless Transmission Losses: A Negative Result," *Proceedings of the Seventh International Conference on Computer Communications and Networks (IC3N)*, New Orleans, October 1998.

[CENS] Deborah Estrin, cens.ucla.edu

[CHMRW03] J. Crowcroft, S. Hand, R. Mortier, T. Roscoe, Timothy, and A. Warfield, "Plutarch: An Argument for Network Pluralism", Workshop on Future Directions in Network

Architecture, Sigcomm 2003.

[CI95] G. Xylomenos, G. Polyzos, P. Mahonen, and M. Saaranen, “TCP Performance Issues over Wireless Links,” IEEE Communication Magazine, April 2001, pp. 52-58.

[CLM96] H. Chaskar, T. V. Lakshman, and U. Madhow, “TCP Over Wireless with Link Level Error Control: Analysis and Design Methodology,” IEEE/ACM Transactions on Networking, 1996.

[COUGAR]http://cougar.cs.cornell.edu/

[DAMAT] B. Allcock, J. Bester, J. Bresnahan, A. L. Chervenak, I. Foster, C. Kesselman, S. Meder, V. Nefedova, D. Quesnal, S. Tuecke. “Data Management and Transfer in High Performance Computational Grid Environments”, Parallel Computing Journal, Vol. 28 (5), May 2002, pp. 749-771.

[DFNPAB]http://www.pab.rwth-aachen.de

[DMKMV01] S. Dawkins, G. Montenegro, M. Kojo, V. Magret, and N. Vaidya, “End-to-end Performance implications of Links with Errors,” *Request for Comments: 3155*, August 2001.

[ENACTS]http://www.epcc.ed.ac.uk/enacts/gridservice.pdf

[802.15]http://www.ieee802.org/15/

[zigbee]http://www.zigbee.org/

[FK03] Fall, Kevin, "A Delay-Tolerant Network Architecture for Challenged Internets" Sigcomm 2003.

[GAPANA]http://www.grid2002.org/ukescience/gapresources/GapAnalysis30June03.pdf

[GARA] Roy, A. and Sander, V., “GARA: A Uniform Quality of Service Architecture”, Resource Management: State of the Art and Future Trends, Kluwer Academic Publishers, pp. 135-144, 2003, Edited by Jarek Nabrzyski, Jennifer M. Schopf and Jan Weglarz.

[GAYNOR]Gaynor, M., and Welsh, M, and Moulton, S, and Rowan, A, and LaCombe, E, and Wynne, J, Integrating Wireless Sensor Networks with the Grid IEEE Internet Computing, special issue on the wireless grid, July/Aug 2004

[GRAAP] K. Czajkowski, S. Pickles, J. Pruyne, and V. Sander. “Usage Scenarios for a Grid Resource Allocation Agreement Protocol”, draft-ggf-graap-usagescenarios-01

[GRIDWELTEN] http://webdoc.gwdg.de/ebook/ah/dfn/gridwelten.pdf

[ORAM] Oram, A., Editor. Peer-to-Peer Harnessing the Power of Disruptive Technologies, March 2001, ISBN 0-596-00110-X

[HIBWD] J. Lee, D. Gunter, B. Tierney, W. Allock, J. Bester, J. Bresnahan, S. Tuecke, “Applied Techniques for High Bandwidth Data Transfers across Wide Area Networks”, Proceedings of Computers in High Energy Physics 2001 (CHEP 2001), Beijing China

[HILL] Hill, Ph.D. Thesis, UC Berkeley, May 2003

[ISCSI] V. Firoiu, F. Travostino, “Performance of iSCSI, FCIP and iFCP”, http://www.pdl.cmu.edu/mailinglists/ips/mail/msg02819.html

[Jac88] V. Jacobson, “Congestion Avoidance and Control,” Proceedings of the ACM SIGCOMM 88, August 1988.

[KA01] Karp, B., Challenges in Geographic Routing: Sparse Networks,Obstacles, and Traffic Provisioning, in the DIMACS Workshop on Pervasive Networking, Piscataway, NJ, May, 2001.

[KAPS02] R. Krishnan, M. Allman, C. Partridge, and J.P.G. Sterbenz, “Explicit Transport Error Notification (ETEN) for Error-Prone Wireless and Satellite Networks”, Technical Report TR-8333, BBN Technologies, March 2002.

[KEW02] B. Krishnamachari, D. Estrin, Deborah, and S. Wicker, "Modelling Data-Centric Routing in Wireless Sensor Networks" , IEEE infocom 2002

[KSEPA04] Rajesh Krishnany, James P.G. Sterbenzz, Wesley M. Eddy{, Craig Partridgey, Mark Allman, “Explicit Transport Error Notification (ETEN) for Error-Prone Wireless and Satellite Networks. Pre-print: Accepted for publication in Elsevier Computer Networks.

[KY02] F. Khafizov and M. Yavuz, "Running TCP over IS-2000," Proceedings of the IEEE International Conference on Communications, 2002, pp. 3444-3448

[LRKO99] R. Ludwig, B. Rathonyi, A. Konrad, K. Oden, and A. Joseph, “Multi-Layer Tracing of TCP over a Reliable Wireless Link,” ACM Sigmetrics, 1999

[MIDCOM]The MiddleBox Communication Working Group, http://www.ietf.org/html.charters/midcom-charter.html

[NDKG02]S. Nath, A. Deshpande, Y. Ke, P. Gibbons, Phillip, B. Karp, and S. Seshan, IrisNet: An Architecture for Compute-Intensive Wide-Area Sensor Network Services, IRP-TR-02-10, Dec 2002,http://www.intel-research.net/Publications/Pittsburgh/012820030729_117.pdf

[NLANR] Nordmark, E., „WAN packet size distribution“, http://www.nlanr.net/NA/Learn/packetsizes.html

[NSIS] The Next Steps in Signaling Working Group, http://www.ietf.org/html.charters/nsis-charter.html

[Pen00] K. Pentikousis, “TCP In Wired-Cum-Wireless Environments,” IRRR Communications Surveys, 2000, pp.2-14

[PG99] C. Parsa and J. J. Gacia-Luna-Aceves, “Improving TCP performance over Wireless Networks at the Link Layer,” Mobile Networks and Applications, 1999.

[PMS00] J. Pan, J. Mark, and X. Shen, “TCP Performance and Its Improvement over Wireless Links, Proceedings of IEEE Globecom 2000, pp. 62-66

[OGSACASES] Eds.: I. Foster, D. Gannon, H. Kishimoto, J. Von Reich, "Open Grid Services Architecture Use Cases", GWD-C (draft-ggf-ogsa-usecase-1.7)

[QUSTART] Quick-Start for TCP and IP, hhttp://www.icir.org/floyd/quickstart.html

[RFC147] Winett, G., "The Definition of a Socket", RFC147, 1971

[RFC1191] Mogul J. and Deering S.,"PATH MTU Discovery", RFC1191 November 1990

[RFC2018] M. Mathis, J. Mahdavi, S. Floy, and A. Romanow, "Selective acknowledgement option," RFC-2018, 1996

[RFC2401] Kent, S. and Atkinson, R., "Security Architecture for the Internet Protocol", RFC2401, November 1998

[RFC2409] Harkins, D. and Carrel, D., "The Internet Key Exchange (IKE)", RFC2409, November 1998

[RFC2616] Fielding, R. et. al., « Hypertext Transfer Protocol – http v1.1 », RFC2612, June1 1999

[RFC2960] Stewart, R., et.al., "Stream Control Transmission Protocol", RFC2960, October 2000

[RFC3168] K. Ramakrishnan, S. Floyd, and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP," Request for Comments: 3168, September 2001.

[RFC3260] Grossman, D., "New Terminology and Clarifications for Diffserv", RFC3260, April 2002

[SIMMOD] J. Padhye, V. Firoiu, D. Towsley and J. Kurose, "Modeling TCP Reno Performance: A Simple Model and its Empirical Validation.", IEEE/ACM Transactions on Networking, April 2000.

[FASTTCP] FAST Protocols for Ultrascale Networks http://netlab.caltech.edu/FAST/index.html

[RON] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient Overlay Networks", 18th ACM Symp. On Operating Systems Principles (SOSP), October 2001

[SCATCP]T. Kelly, Scalable TCP: Improving Performance in Highspeed Wide Area Networks http://www-lce.eng.cam.ac.uk/~ctk21/scalable/

[AGGFLOW] T. Hacker, B. Noble, B. Athey, "The Effects of Systemic Packet Loss on Aggregate TCP Flows," Proceedings of SuperComputing 2002

[RFC2914] S. Floyd, "Congestion Control Principles," RFC2914, September 2000.

[GTFWALL] Globus Toolkit Firewall Requirements http://www.globus.org/Security/v2.0/Globus%20Firewall%20Requirements-0.3.pdf

[SRELMUL] Scalable Reliable Multicasting projects http://dsonline.computer.org/middleware/MWprojects.htm#scal able

[SURTRAN] Suvery of Transport Protocols Other than Standard TCP http://www.evl.uic.edu/eric/atp/

[FBUFS] Druschel, P. and Peterson, L. L. High Performance Cross-Domain Data Transfer. Technical report, Department of Computer Science, University of Arizona, 1992. Technical Report 92-11

[WITLESS]Van Jacobson. Presentation about the “witless” host network interface. INTEROP, San Jose, California, October 1991

[UNET] U-Net A User-Level Network Interface for Parallel and Distributed Computing, Anindya Basu, Vineet Buch, Werner Vogels, Thorsten von Eicken. Proceedings of the 15th ACM Symposium on Operating Systems Principles (SOSP), Copper Mountain, Colorado, December 3-6, 1995

[VIARCH]The Virtual Interface Architecture, http//www.viaarch.org

[RDDP] The Remote Direct Data Placement Working Group, http//www.ietf.org/html.charters/rddp-charter.html

[RFC1928] M.Leech et al., SOCKS Protocol Version 5, RFC1928, March 1996

[RFC1122] R. Braden, Requirements for Internet Hosts -- Communication Layers, RFC1122, October 1989

[STOPAR]J. Stone, Craig Partridge, “When the CRC and TCP checksum Disagree”, Proceedings of SIGCOMM 2000, Stockholm, Sweden September 2000

[OIFUNI]The Optical Internetworking Forum’s UNI, http://www.oiforum.com/public/impagreements.html#UNI

[OPTIP] OptIPuter, http//www.calit2.net/news/2002/9-25-optiputer.html

[TCPFRD] http://www.psc.edu/networking/tcp_friendly.html

[WDD] CANARIE’s Wavelength Disk Drive (WDD) http//www.ccc.on.ca/wdd/header.htm

[DEE88] S. E. Deering and D. R. Cheriton. Multicast Routing in Datagram Internetworks and Extended LANs. In ACM SIGCOMM'88 and ACM Trans. on Comp. Sys., Vol. 8, No. 2

[GEM03] Jim Gemmell and al. The PGM Reliable Multicast Protocol. In IEEE Networks. Multicasting: An Enabling Technology. (previously an internet draft: T.~Speakman et~al. Pgm reliable transport protocol specification. internet draft, 1998)

[STO00] Ion Stoica, T. S. Eugene Ng and Hui Zhang. REUNITE: A Recursive Unicast Approach to Multicast. In IEEE INFOCOM 2000

[HUI00] Yang-hua Chu, Sanjay G. Rao, and Hui Zhang. A Case forEnd-System Multicast. In ACM SIGMETRICS 2000

[SAY03] A. El-Sayed, V. Roca and L. Mathy. A survey of Proposals for an Alternative Group Communication Service. In IEEE Network magazine, Special issue on ``Multicasting: An Enabling Technology',January/February 2003

[XPMS01] G. Xylomenos, G. Polyzos, P. Mahonen, and M. Saaranen, "TCP Performance Issues over Wireless Links," IEEE Communication Magazine, April 2001, pp. 52-58

[YAY95] R. Yavatkar, James Griffoen, and Madhu Sudan. A reliable dissemination protocol for interactive collaborative applications. In ACM Multimedia, 1995

[FLO97] Sally Floyd, Van Jacobson, Ching-Gung Liu, Steven McCanne, and Lixia Zhang. A reliable multicast framework for light-weight sessions and application level framing. In IEEE ACM Transactions on Networking, 5(6), 1997

[PAP98] Christos Papadopoulos, Guru M. Parulkar, and George Varghese. An error control scheme for large-scale multicast applications. In Proc. of the IEEE INFOCOM, March 1998

[PAU97] S. Paul and K. Sabnani. Reliable multicast transport protocol (RMTP). In IEEE JSAC, Spec. Issue on Network Support for Multipoint Communications, 15(3), April 1997

[PKB99] K. S. J. Pister, J. M. Kahn and B. E. Boser, "Smart Dust: Wireless Networks of Millimeter-Scale Sensor Nodes", Highlight Article in 1999 Electronics Research Laboratory Research Summary.

[TDB] http://telegraph.cs.berkeley.edu/tinydb/

[TOS] http://webs.cs.berkeley.edu/tos/

[XBOW] http//www.xbow.com/Products/Wireless Sensor Networks.htm

[XTP95] XTP Forum. Xpress Transport Protocol Specification, March 1995

[MSEC] "The Multicast Security Architecture",draft-ietf-msec-arch-01.txt

[RFC 3272] Overview and Principles of Internet Traffic Engineering. D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, X. Xiao. May 2002

[RFC 2702] Requirements for Traffic Engineering Over MPLS. D. Awduche, J. Malcolm, J. Agogbua, M. O'Dell, J. McManus. September 1999

[RFC 2475] An Architecture for Differentiated Services. S. Blake et. al.

[RFC 3346] Applicability Statement for Traffic Engineering with MPLS. J. Boyle, V. Gill, A. Hannan, D. Cooper, D. Awduche, B. Christian, W.S. Lai. August 2002

Annex A:

Moving a data set between two sites using multiple TCP sessions provides significantly higher aggregate average throughput than transporting the same data set over a single TCP session, the difference being proportional to the square of the number of TCP sessions employed. This is the outcome of a quantitative analysis detailed in annexe A.

using three simplifying assumptions:

1. the sender always has data ready to send

2. the costs of striping and collating the data back are not considered

3. the end-systems have unlimited local I/O capabilities.

It is well-known that 2) and 3) are not viable assumptions in real-life, therefore the outcome of the analysis has baseline relevance only.

Throughput dynamics are linked to the way TCP congestion control reacts to packet losses. There are several reasons for packet losses: network congestion, link errors, and network errors. Network congestion is pervasive in current IP networks, where the only way to control congestion is through dropping packets. Traffic engineering, admission control and bandwidth reservation are currently in early stages of definition. DiffServ-supporting QoS infrastructures will not be widely available in the near future.

Even in a perfectly engineered network, link errors occur. If we take an objective of 10**(-12) Bit Error Rate, for a 10Gbps link, this amounts to one error every 100 seconds. Network errors can occur with significant frequency in IP networks. [STOPAR] shows that network errors caught by TCP checksum occur between one packet in 1100 and 1 in 32000, and without link CRC catching it.

TCP throughput is impacted by each packet loss. Following TCP's congestion control algorithm existent in all major implementations (Tahoe, Reno, New-Reno, SACK), each packet loss results in the TCP sender's congestion window being reduced to half of its current value, and therefore (assuming constant Round Trip Time), TCP's throughput is halved. After that, the window increases linearly by roughly one packet every two Round Trip Times (assuming the popular Delayed-Acknowledgement algorithm). The temporary decrease in TCP's rate translates into an amount of data missing transmission opportunity. As shown below, the amount of data missing the opportunity to be transmitted due to a packet loss is (see [ISCSI] for mathematical derivations relative to TCP Reno):

D(N) = E**2/(N**2)*RTT**2/(256*M)

where

D = amount of data not transmitted due to packet loss, in MB

E = Total bandwidth of an IP "pipe", in bps

N = number of TCP streams sharing the bandwidth E, unitless

RTT = Round Trip Time, in ms

M = packet size, bytes

For example, for a set of N=100 connections totaling E=10Gbps, RTT=10ms, M=1500B, the data not transmitted in time due to a packet loss is D(N)=2.6MB.

To show this consider the following hypothetical graph of bandwidth versus time:

```
              |              Tr
      E/N     |------------     ----------------------
              |           |    /
```

```
                  |       | A /
                  |     E/2*N |  / slope
   Bandwidth      |           | /
     (bps)        |           |/
                  |
                  |
                  |
                  |
                  |
                  |
                  +------------------------------------------------
                  |              Time (seconds)
```

First, the area inside the triangle, A, is 1/2 base * height. The base has units of seconds and the height bps, and the product, bits. This represents the data not transmitted due to loss. The expression for the height is easily obtained since, as noted above, a dropped packet causes the bandwidth to be cut in half. TCP also specifies that the amount of data in-flight increases by one packet every 2 round trip times. We can calculate the corresponding increase in bandwidth from the equation for the bandwidth delay product [HIBWD].

This equation states buffer size = bandwidth * RTT, or rearranged the bandwidth = buffer size / RTT. So, our increase in bandwidth is M/RTT. We get this increase every x * RTT seconds, so the rate of recovery (the slope in the diagram) = M/RTT / xRTT or M/x*RTT^2 and has units of bps/s. We can now determine the recovery time(Tr), which is the base of the triangle, to be E/2N * x*RTT^2 / (8M). Finally, we can determine the equation for the area of the triangle. Using the units listed above and appropriate conversions:

```
        1 E (Mb) * E (Mb) *  x * RTT^2 (ms^2) (1 sec)^2
     = --------------------------------------------------- *
        2 2*N(s)   2*N(s) *                   (10^3 ms)^2

          * (byte) * 10^6 bits * MB
 ---------------------------------------
  M (bytes) * 8 bits *   Mb       8 Mb
```

In absence of Delayed-Acknowledgements (x=1) we get:

```
    E^2 * 2 * RTT^2           (1*10^4)^2 * 2 * (10)^2
 ==> ----------------------- =   -------------------------------
    N^2 * M * 256              (100)^2 * 1500 * 256
```

Using our previous example of a set of N=100 connections totaling E=10Gbps, RTT=10ms, M=1500B, the time interval for TCP to recover its sending rate to its initial value after a packet loss is I(N)= 0.833 seconds.

If N = 1, the time to recover its rate, I(1)=83.3s, is of the same order of magnitude as the time between two packet losses due exclusively to the link Bit Error Rate. In other words, a packet loss occurs almost immediately after TCP has recovered its rate. This means that N=1 delivers on average just about 3/4 of the required 10Gb/s rate, since 1/4 of rate is lost during the time TCP rate increases linearly from 1/2 to full rate. (More precisely, the effective rate is 8.27Gb/s because 1/4 of rate is lost during 83.3s, and the time between two errors is now 120.825s due to decreased sending rate).

Consideration of this equation also reveals another major issue with TCP on high latency networks. Notice that the recovery time is directly proportional to the square of the RTT. This means that doubling the RTT will result in a 4x increase in the recovery time, making dropped packet even more problematic, and multi-stream TCP even more valuable. The impact of packet losses on multi-stream TCP settings has been analyzed in [AGGFLOW].

GridFTP [DAMAT] is a real world application that uses multiple streams to obtain high performance during file transfers. There is no adequate data available to demonstrate the performance in the face of packet loss; however, it can be clearly shown that aggregate throughput is dramatically improved with multi-stream TCP. There are, as you would expect, differences from the much-simplified scenario used above. Differences include the inability to utilize full bandwidth in a single stream, and a distinct "knee" after which additional streams provide only limited additional improvement in performance. There are a host of complicating factors that could account for these differences. One of them is clearly the simplification in the model. However, other factors could include buffer copies from kernel space to user space, bus bandwidth, disk performance, CPU load, etc..