Grid High Performance Networking Research Group

GRID WORKING DRAFT

Document: draft-ggf-ghpn-netissues-1
Category: Informational Track

http://forge.gridforum.org/projects/ghpn-rg/

Volker Sander
(Editor)
Forschungszentrum
Jülich GmbH

William Allcock
Argonne National Lab.

Pham CongDuc
Ecole Normale
Superieure Lyon

Jon Crowcroft
Univ. of Cambridge

Mark Gaynor
Boston University

Doan B. Hoang
University of
Technology, Sydney

Inder Monga
Nortel Networks Labs

Pradeep Padala
University of Florida

Marco Tana
University of Lecce

Franco Travostino
Nortel Networks Labs

September 2003

Networking Issues of Grid Infrastructures

Status of this Memo

Comments

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

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. The vice-versa relationship between
the two communities is addressed by two documents, each of it
describing the relation from the particular view of either group.
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 heavily distributed infrastructure with
ambitious service demands to stress 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.   End-Systems

This section describes experienced issues related to End-Systems.

3.1   Communication Protocols and their Implementation

The evolution of the Transmission Control Protocol (TCP) is a good
example on how the specification of communication protocols evolves
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.   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.   This section
describes problems encountered with the implementation of
communication protocols, with a focus on TCP.

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 semantical gap between socket buffer interface
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
setsockopt()-call.

In fact, the window scale option is derived from the socket buffer
size used during the connect()- and listen()-call. Unfortunately,
this selection is 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.

3.2   Operating System Capabilities and Configuration Issues

Similarly to the above described influence of the selected socket
buffer size, widely deployed operating systems do have a strong
impact on the achievable level of service. They 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 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.

3.3   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, network bandwidth (be it in access, metro, core). A
chief role of an Operating System (OS) 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 OS 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 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 OS, 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 OS, 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 OS-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.

3.4   TCP Considerations

This section lists TCP related considerations.

3.4.1   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. If you
think you can use less than recent history rather than recent
measurements, look at the Congestion Manager and TCP PCB state
shearing work first!

3.4.2   Congestion Control

Congestion control is mandatory in best-effort networks. ISPs might
interrupt the service when congestion control is not performed.
AIMD and Equation Based

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.

3.4.3   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)

3.4.4   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.

3.5   Multi-Stream File Transfers with TCP

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 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)
```
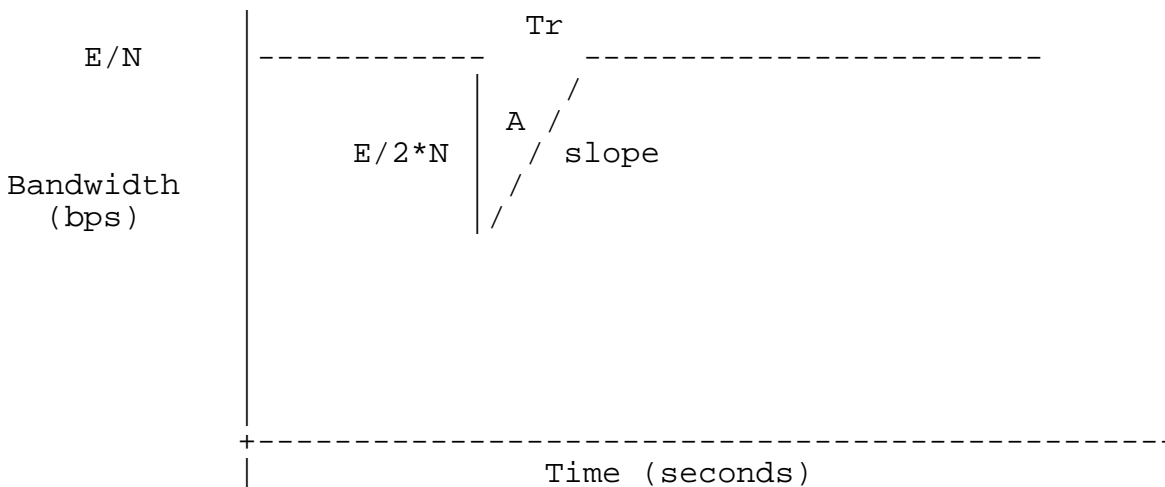
First, the area inside the triangle, A, is 1/2 base * height. The
base has units of seconds and the height bps, and the product,
bits. This represents the data not transmitted due to loss.   The
expression for the height is easily obtained since, as noted above,
a dropped packet causes the bandwidth 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..

In conclusion, 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.

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]. [RFC2914] documents key issues 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.

3.6   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 MTU is 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 size.

3.6.1   Multicast MSS

Sub-IP packet size is a 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 these are
anathema to good performance. More detailed information can be
found at [NLANR] and [RFC1191].

3.7   Miscellaneous

This section lists a variety of ongoing relevant efforts and issues.

3.7.1   RMT and Unicast

Reliable Multicast Transport protocols (PGM, ALC) use a variety of techniques to mimic TCP mainly.

3.7.2   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: The General Packet Radio Service (GPRS – a 2.5 Generation wireless networks) is a modification of the Global System for Mobile phone (GSM) protocol stack that makes it more closely match the TCP/IP stack. 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 vs 9.4 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 wire links. Wireless links still run at lower speeds, and as we saw, low data throughput on some of these wireless links means that the low bandwidth can make 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. 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.

3.7.3   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.

3.7.4   Economics, Fairness etc

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. A service differentiation at the access links are a potential solution to this, i.e. customers that paid less were bottlenecked at their access links in that case.

http://www.psc.edu/networking/tcp_friendly.html

3.7.5   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.

4.   IPv6

Does IPv6 Help (don't laugh!) - yes it might!

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.

5.   Routing

Priorities for good routing system design are:

5.1   Fast Forwarding

Packet classification and switched routers have come a long way
recently - we are unlikely in the software world to beat the h/w in
core routers, but we can compete nicely in access devices -
certainly, there is no reason why a small cluster couldn't make a
good 10Gbps router - but there's every reason why a PCI bus machine
maxes out at 1Gbps!

5.2   Faster Convergence

Routers and links fail. the job of OSPF/ISIS and BGP is to find the
alternate paths quickly - in reality they take a whole to converge
- IGPs take a while (despite being mainly link state nowadays)
because link failure detection is NOT obvious - sometimes you have
to count missed HELLO packets (since some links don't generate an
explicit clock). BGP convergence is a joke. But there are smart
people on the case.

5.3   Theory and practice

Most the problems with implementing routing protocols are those 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.

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

Equal cost Multipath OSPF and QOSPF have been dreamt up - are they
used a lot? Multipath in limited cases 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!

5.5   MPLS

The Multiprotocol Label Switching Architecture (MPLS) 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. Other experiences show, however, that some
functions (e.g. Multicast) are not well supported on an MPLS-
substrate.

5.6   BGP

Policies are hard - BGP allows one to express unilateral policies
to the planet. this is cute (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).

6.   Access Domains

This section describes experienced issues related to access
domains.

6.1   Firewalls

Firewalls pose interesting problems in grids. 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.

6.2   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.

6.3   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).

6.4   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 API.

7.   Transport Service Domains

This section describes experienced issues related to the core.

7.1   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.

7.1.1   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).

7.1.2   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**: So we are not sharing as we would if we
  adapted under congestion control

- **Scheduling** :So we do not experience 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.

7.1.3   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 the grid 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.

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 Fiber Channel
service 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.

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.

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

7.1.4   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.

7.1.5   On-demand SLS

One of the major values of the grid is the ability to form grid
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.

7.2   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.

8.   General Issues

So far, we listed issues which were related to a particular region
of the interconnecting network. This section lists the remaining
issues.

8.1   Service Orientation and Specification

In some sense, the Grid can be compared with the World Wide Web.
While the original goal of the World Wide Web was to offer location
independent access to information resources by using a simple
protocol, a common name space (the Universal Resource Locator), the
integration of the name space into a standardized hypertext
language, and a graphical user interface supporting these hypertext
documents, the Grid is about to offer individuals and institutions
the opportunity to build virtual organizations which facilitates
the access to the problem solving services of the community. A Grid
infrastructure facilitates the composition of existing services to
build more advanced, higher-level services.

Of course, when following this service oriented view of the Grid,
the question arises about network related services. Low-latency and
high-throughput communication are performance-critical in most
distributed environments. For Grid applications, the demanded level
of value-addedd services is basically as follows:

  • 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 (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.

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.

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.

8.2   Programming Models

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.

The 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.

8.3   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 you 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 - they do content sharing and have cute 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)

Peer-to-Peer Harnessing the Power of Disruptive Technologies

Edited by Andy Oram, March 2001, 0-596-00110-X

8.4   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 [DEERING] 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.

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

8.5   Sensor Networks

Sensors are a vital part of the grid because they allow integration
of real-time data into grid applications. Some sensors will be
complex, and wired directly into the grid, using standard grid
protocols. 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. 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.

9.   Macroscopic Traffic and System Considerations

Self similarity, so? traffic is self similar (i.e. arrivals are not
i.i.d) - 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 is a bad thing

9.1   Flash Crowds

e.g. genome publication of new result followed by simultaneous
dbase search with similar queries from lots of different places...

9.2   Asymmetry

Many things in the net are asymmetric - see ADSL lines, see client-
server, master-slave, see most NAT boxes. See BGP paths. beware -
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.

10.   Traffic Engineering

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 (IGP) 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.

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). One of the most
appealing features of MPLS-TE (Multiprotocol Label Switching-
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 ther is the risk of either an inefficient grid, a grid
that is incapable of handling its load, or of clients that start to
chronically overuse their available network resources.

11.   Security Considerations

Network security can be implemented at the link level (i.e., L2, as
in WEP or FrameRelay 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.

11.1   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 GSI 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.

Section 4.3 details other coordination issues between GSI users and
legacy VPNs devices (failures, timeouts).

11.2   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).

11.3   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 leverages the weakest policy definitions and

especially their implementations. Exposure to these risks is inherent to the way Grids work. Hence 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.

12.   Acknowledgments

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

13.   Author's Addresses

Volker Sander (Editor)   William Allcock
Forschungszentrum Jülich   Argonne National Lab.
GmbH   allcock@mcs.anl.gov
v.sander@fz-juelich.de

Pham CongDuc   Inder Monga
Ecole Normale Superieure   Nortel Networks
Lyon   imonga@nortelnetworks.com
Congduc.Pham@ens-lyon.fr
http://www.ens-
lyon.fr/~cpham
Pradeep Padala   Marco Tana
University of Florida   University of Lecce
ppadala@cise.ufl.edu   marco.tana@unile.it

Franco Travostino   Jon Crowcroft
Nortel Networks   University of Cambridge
travos@nortelnetworks.com   Jon.Crowcroft@cl.cam.ac.uk

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

14.   References

    [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

[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.

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

[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.

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

[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

[QUSTART] Quick-Start for TCP and IP,
        hhtp://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

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

[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%20Req
         uirements-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

[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 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

Intellectual Property Statement