РАСПРЕДЕЛЕННЫЙ ИНТЕРКОННЕКТ: СТРУКТУРА ПАРАЛЛЕЛЬНЫХ КАНАЛОВ И ИНСТРУМЕНТАРИЙ ИССЛЕДОВАНИЯ ТРАНСПОРТНЫХ ПОТОКОВ В НИХ

А.Г. Масич, Г.Ф. Масич, И.А. Сидоров

Описан распределенный Interconnect, соединяющий внутренние Interconnect территориально распределенных кластеров. Приведена структура параллельных скоростных каналов связи распределенного Interconnect и рассмотрены проблемы использования традиционных транспортных протоколов в них. Выбран инструментарий для исследования протокола TCP и проведены измерения, по результатам которых найдены оптимальной стратегии управления перегрузкой в зависимости от числа конкурирующих потоков. Показана область эффективного использования TCP стратегий в зависимости от числа параллельных соединений. Исследования проводятся при поддержке РФФИ (грант №11-07-96001-р_урал_а) и УрО РАН (грант 12-П-1-2012, проекты РЦП-13-Т1, РЦП-13-И3).

1. Введение. Пропускная способность национальных и региональных научно-образовательных (R&E - Research and Education) оптических сетей со спектральным уплотнением каналов (WDM - Wavelength Division Multiplexing) не является дефицитным ресурсом, вместо этого критическим является подключение к скоростной сети вычислителей и систем хранения данных. Различные длины волн света в одном оптическом волокне, называются «лямбда», используются для разделения каналов передачи данных и обеспечивают передачу по каждой лямбде на скорости 10-40-100 Гбит/с.

Lambda Grid парадигма распределенных вычислений была впервые сформулирована международной виртуальной организацией GLIF (Global Lambda Integrated Facility, http://www.glif.is), основывается на использовании параллелизма многочисленных лямбд в одном оптическом волокне и направлена на поддержку наиболее требовательных к скоростям и вычислительным ресурсам научных приложений. В этой модели сеть является системообразующим компонентом, своего рода материнской платой, объединяющей вычислители, хранилища, системы визуализации и установки, порождающие интенсивные потоки данных.

Несоответствие между вычислительной производительностью и компонентами ввода/вывода высокопроизводительных систем текущего поколения сделал ввод/вывод наиболее узким местом. Одним из основных источников ухудшения совокупной производительности территориально распределенных высокоскоростных приложений является плохая end-to-end производительность протокола TCP. Доступные измерительные инструменты (ping, Traceroute, Tcpdump, Pchar, Iperf) не позволяют выявить первопричину плохой производительности TCP и диагностировать эффективность результатов настройки в TCP стратегии управления перегрузкой.

Далее структура статьи выглядит следующим образом. В разделе 2 приведена разработанная инфраструктура распределенного Interconnect и структура параллельных каналов связи, связывающих внутренние Interconnect кластеров и систем хранения данных. В разделе 3 описываются проблемы TCP протокола при работе по скоростным сетям большой протяженности, а разделе 4 — исследовательский инструментарий и инсталляции для анализа транспортных потоков. В разделе 5 приведены и проанализированы результаты измерений, в разделе 6 сформулирована методология оценки эффективности стратегий управления потоком в TCP.

2. Инфраструктура.

Distributed Interconnect (рис.1) представляет собой распределенный суперкомпьютер из двух однородных кластеров и систем хранения, расположенных в ИМСС УрО РАН (Пермь) и ИММ УрО РАН (Екатеринбург). Цель - создание общей вычислительной инфраструктуры для исследователей, которые работают по различным аспектам параллельных, распределенных, сетевых и облачных вычислений. Отличительная особенность - использование лямбда каналов оптической магистрали (DWDM Backbone) для связи внутренних Interconnect кластеров и систем хранения данных.

DWDM Backbone создан в рамках Инициативы GIGA UrB RAS [1, 2]. Используется темное волокно (dark fiber) магистральных операторов связи (L=456км) и DWDM оборудование компании ECI-Telecom: платформы XDM-2000 на оконечных узлах и XDM-40 на промежуточных узлах. Оптические мультиплексоры (MUX) на 16 лямбда каналов обеспечивают возможность использования транспондеров со скоростью передачи 10-40 Гбит/с в каждой лямбде.

Структура параллельных каналов связи. Установленные транспондеры используют три лямбда канала со скоростью потоков по 10,7 Гбит/с (ITU-T G.709) в каждой лямбде, поддерживают механизм упреждающего исправления ошибок (FEC- Forward Error Correction) и имеют следующие характеристики портов ввода/вывода для подключения оконечного оборудования: 2 Ethernet порта по 10 Гбит/с (2x10GE) и 8 Ethernet портов по 1 Гбит/с (8x1GE). Такая совокупность портов образует слой гарантированных каналов связи

по схеме точка-точка, позволяет использовать распространенные в оконечных системах 1GE порты и осуществить постепенный переход на скорость 10 Гбит/с. Как показано на рис.1, гарантированными каналами по 1 Гбит/с связаны локальные сети (LAN) и Ethernet Interconnect(ы) вычислителей, а гарантированным каналом 10 Гбит/с соединены системы хранения данных. В стадии проработки сопряжение InfiniBand (IB) interconnect вычислителей. Оставшиеся 13 лямбд — потенциал масштабирования по 10-40 Гбит/с или когерентной DWDM 100 Гбит/с. Для большей гибкости проводимых исследований на площадках установлены L2 коммутаторы Cesar, имеющие по 24 порта 1GE и 4 порта 10GE, один из которых подключен к 10GE порту XDM2000. Эта совокупность портов образует слой не гарантированных каналов связи при нагрузке больше 10 Гбит/с. Система мониторинга и управления DWDM Backbone располагается в центре управления сетью в ИМСС УрО РАН (Пермь).

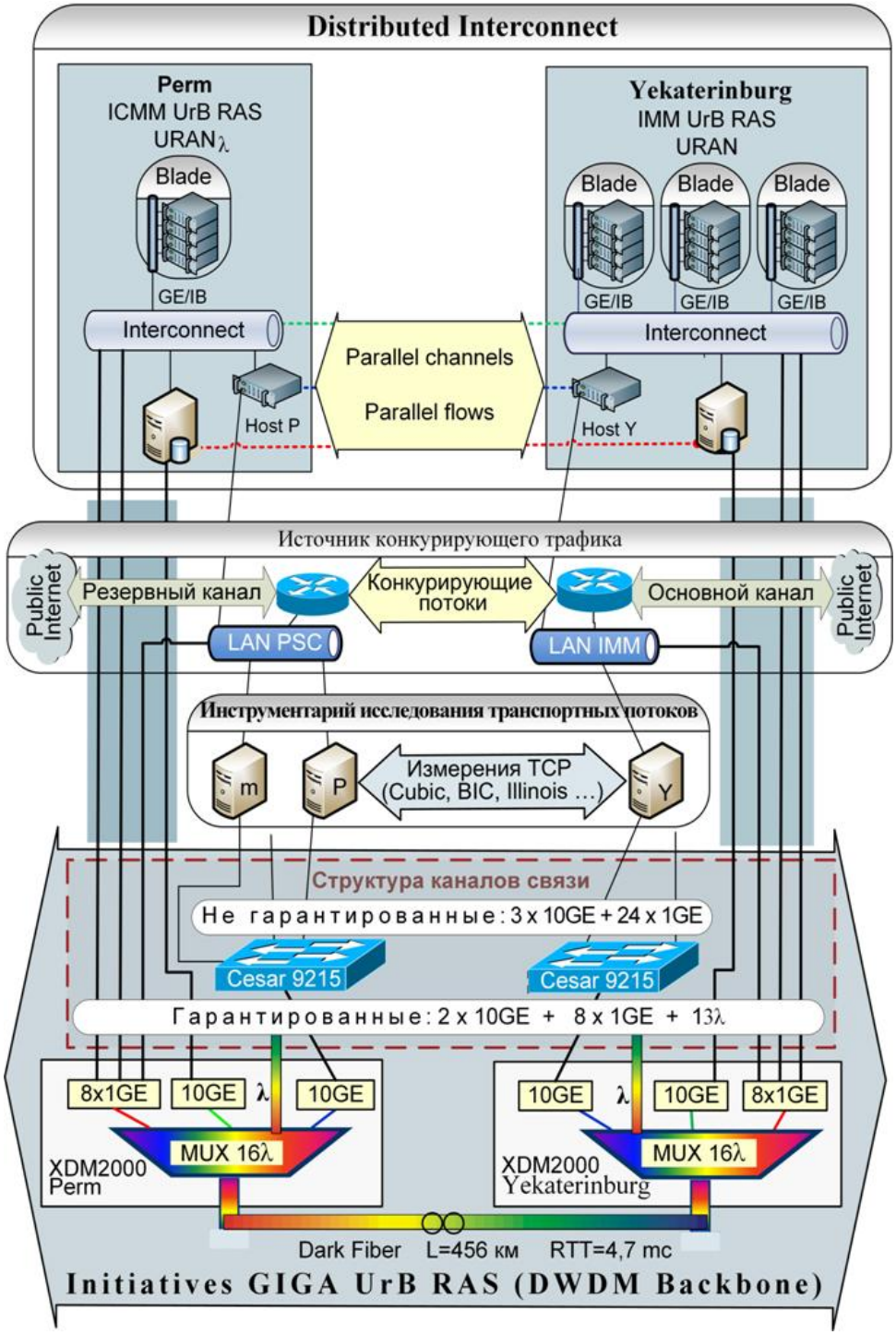

Рис.1 Инфраструктура: распределенный Interconnect и структура каналов связи

Инструментарий для исследования транспортных потоков. Компьютеры исследователей (P, Y) используются для порождения потоков данных транспортными протоколами (например, TCP) по каналам с присутствующими в них конкурирующими потоками данных (или без них) с целью изучения влияния стратегий управления потоком (например, Cubic, BIC, Illinois) на производительность. Зеркало m используется для

сохранения потоков передаваемых данных для последующего их временного анализа. Подключение сети Пермского научного центра (LAN PSC) к основному каналу доступа в Интернет в Екатеринбурге на скорости 1 Гбит/с порождает классический интернет трафик - более 1000 транспортных соединений с различными RTT. Этот трафик является конкурирующим для потока между P и HostY.

Аналогичная инфраструктура. DAS-4 (Distributed ASCI Supercomputer) - распределенный ASCI суперкомпьютер из шести территориально распределённых разнородных кластеров в Нидерландах, расположенных в голландских университетах и используемых для проведения Computer Science исследований (http://www.cs.vu.nl/das4/clusters.shtml). ASCI является аббревиатурой от "Advanced School for Computing and Imaging" - голландской магистерской школы, основанной в 1993 году и аккредитованной Нидерландской королевской академии наук и искусств (Royal Netherlands Academy of Arts and Sciences). DAS-4 разработана ASCI и финансируется NWO / NCF (Нидерландская организация по научным исследованиям). DAS-4 использует выделенный глобальный интерконнект, связывающий внутренние интерконнект кластеров по скоростным выделенным лямбда каналам, предоставляемым сетью SURFnet. SURFnet - голландская научно-образовательная сеть (NREN -National Research and Educational Networks), которая в рамках проекта StarPlane [3] предоставила для DAS-4 выделенные 10 Гбит/с лямбда каналы DWDM магистрали SURFnet.

3. Протокол TCP в сетях с большим BDP.

В RFC5681 (M. Allman, 2009) определены 4 связанных между собой алгоритма управления перегрузкой (congestion control algorithms) для протокола TCP: медленный старт (SS – slow start), предотвращения перегрузки (CA – congestion avoidance), быстрой повторной передачи (FR – fast retransmit) и быстрого восстановления (fast recovery). Алгоритмы SS и CA используются отправителем TCP для контроля передачи еще не переданных данных. Для отображения состояния TCP-соединения используются много переменных, важнейшими из которых являются:

1. Окно перегрузки cwnd (congestion window) задаваемое на стороне отправителя количество данных, которые отправитель может передать в сеть до получения подтверждения (ACK).
2. Анонсируемое получателем окно rwnd, которое определяет установленный на приемной стороне предел размера остающихся данных. Передачей управляет меньшее из двух значений cwnd и rwnd.
3. Порог медленного старта ssthresh (slowstart threshold) используется для определения момента, до которого следует использовать алгоритм медленного старта.
4. Максимальный размер передаваемого сегмента SMSS (Sender Maximum Segment Size).

Алгоритм SS используется для зондирования сети с целью определения пропускной способности после установления соединения или после восстановления потерянных пакетов по таймеру повторной передачи RTO. Начальное значение ssthresh следует устанавливать сколь угодно высоким, но оно должно быть уменьшено при возникновении перегрузки. В фазе SS TCP увеличивает размер окна cwnd не более чем на SMSS байтов для каждого пакета ACK, кумулятивно подтверждающего доставку новой порции данных. Хотя традиционные реализации TCP увеличивают cwnd ровно на SMSS байтов при получении ACK, подтверждающего новые данные, RFC5681 рекомендует увеличивать значение cwnd, как показано в уравнении (1):

$$cwnd{+}{=}min(N, SMSS) \quad (1)$$

где N - число ранее не подтвержденных байтов, подтверждаемых входящим ACK.

Отметим, что RFC3465 (M. Allman, 2003) разрешает увеличивать cwnd более чем на SMSS байтов для входящих подтверждений в процессе медленного старта в качестве экспериментов и не допустимо в качестве стандартного. Все TCP версии используют фазу SS после установления соединения (флаг Syn) до достижения cwnd < threshold и используют различные версии алгоритмов CA при cwnd ≥ threshold.

Версии TCP направлены на увеличение их эффективности (пропускной способности) в различных условиях функционирования. Эволюция TCP версий в условиях присутствия конкурирующих потоков приводят к таким метрикам как честность (Fairness) и стабильность (Stability). Характеристики честности определяют, насколько справедливо одновременные TCP потоки делят пропускную способность. А под «стабильностью» понимается описание колебательных характеристик потока данных. Плавный поток - это желательное поведение в большинстве ситуаций, и часто (хотя и не обязательно), ведет к увеличению пропускной способности. Так например, в протоколе TCP Reno (Jacobson, 1988) пересмотрена концепция SS после обнаружения ошибки путем введения различия между большими и маленькими событиями задержки. При обнаружении потерь по тайм-ауту повторной передачи отправитель устанавливает cwnd в единицу. Однако при получении duplicate ACK окно перегрузки уменьшается только до значения cwnd/2. Дальнейшие модификации TCP связаны с фазой предотвращения перегрузки при cwnd > threshold, в которой темп увеличения cwnd определяется вариантом TCP. По индикации перегрузки различные варианты TCP протокола делятся на три группы: по сигналу потери (loss-based), по задержке (delay-based) и использующие оба показателя (hibrid). На рисунке 2 приведены функций роста окна пяти вариантов высокоскоростных TCP, которые используют основанный на потере (loss-based) подход. HSTCP (Floyd 03) применяет линейную функцию роста, HTCP

(Shorten 04) - степенную функцию роста и STCP (Kelly 03) - функцию экспоненциального роста. В BIC (Xu 04) применяется и логарифмические функции и функции экспоненциального роста. Cubic (Rhee 05) применяет кубическую функцию. Как видно из рисунка 2, разница между BIC и Cubic заключается в их выпуклости. При нормальном рабочем диапазоне, Cubic больше вогнутых чем BIC. Cubic улучшает масштабируемость для протяженных скоростных сетей, имеет более справедливое распределение полосы пропускания между параллельными потоками с различными RTTs и обладает высокой степенью стабильности. Реализация Cubic в Linux прошла несколько модернизаций и является по умолчанию протоколом управления перегрузкой для TCP в Linux.

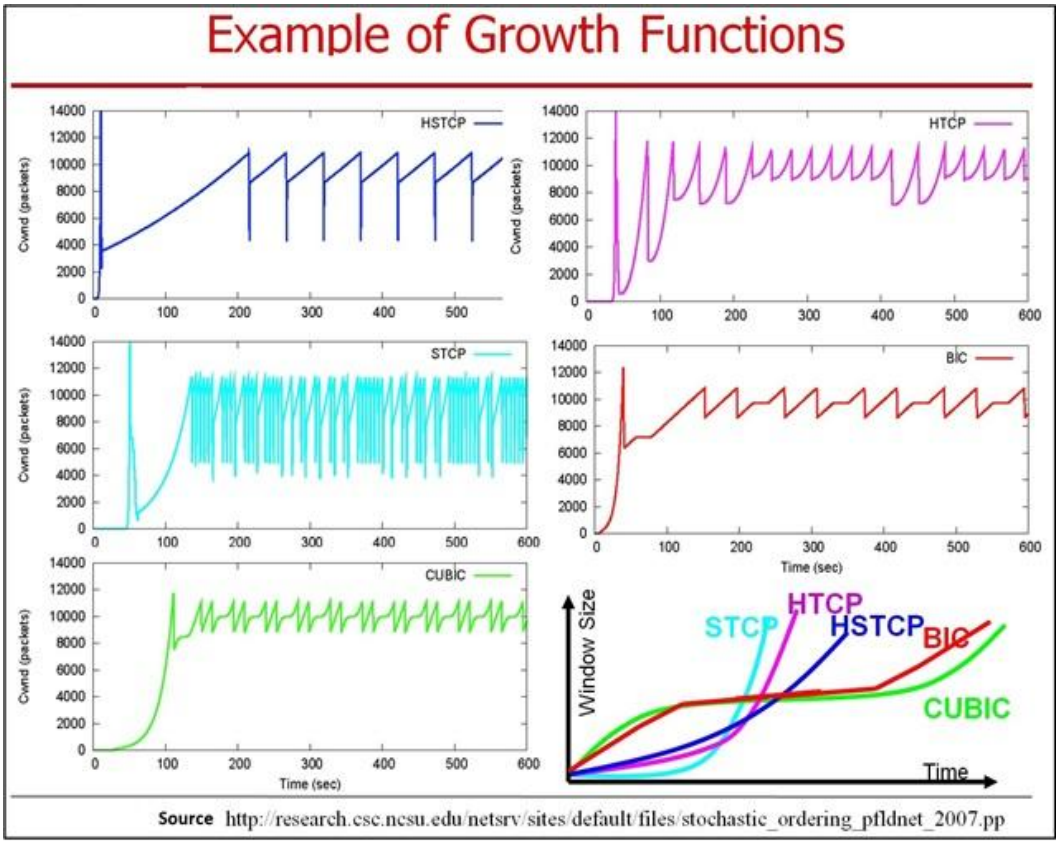

Рис.2. Примеры функций увеличения окна перегрузки cwnd в высокоскоростных TCP

Большое влияние на скорость передачи в протоколах, основанных на оконных принципах передачи, оказывают ошибки. M. Mathis [4] описал отношение между верхней границей пропускной способности TCP (BW), максимальным размером сегмента MSS, RTT пакетов и потерей пакетов p при помощи уравнения (1).

$$p < \left(\frac{MSS}{BW * RTT}\right)^2 \qquad (2)$$

Из уравнения (1) следует, что для достижения существенной производительности TCP при относительно большом RTT максимальный уровень потери пакетов p должен быть очень низким.

Отметим также, что произведение BW*RTT называется в англоязычной литературе параметром BDP (bandwidth-delay product), большая величина которой характерна для скоростных сетях большой протяженности. Например, для участка DWDM магистрали Пермь-Екатеринбург с оптической длиною 456 км и временем распространения сигнала по оптическому волокну 5 мкс/км параметр RTT равен 4,7 мс. С учетом задержек обработки сегментов в оконечных системах примем RTT=5мс. Тогда для скорости передачи 1 Гбит/с BDP=625000 байт, а для скорости 10 Гбит/с BDP равно 6250000 байт или 4130 пакетов длиною 1514 байтов.

Для TCP в фазе медленного старта чтобы увеличить окно от 1 до 4130 пакетов потребуется приблизительно 16 RTTs (12 + 3 на троекратное рукопожатие) ~ 90 мс. И если TCP соединение завершится раньше или будут одиночные ошибки с вероятностью p > 5,8 x 10-7 (согласно формуле 1), полоса пропускания канала будет существенным образом недоиспользована. Это делает передачу данных в рамках протокола TCP, используемого в большинстве операционных систем, включая Windows и Linux, достаточно вялой, не использующей всю имеющуюся полосу пропускания.

4. Инструментарий для измерений.

В проектах Network Weather Service, NetLogger и Gloperf занимаются проектированием и разработкой пользовательского уровня диагностики и управления пропускной способностью сети, полный список которых можно найти в [5]. Небольшой набор инструментов измерения производительности транспортных протоколов,

таких как Treno и TCP Testrig доступны, но использование этих инструментов требует обширных знаний сетей и характеристик TCP наряду с привилегированным доступом к сетевым устройствам в операционной системе.

По нашему мнению, наиболее удачным подходом решения проблемы тонкой настройки TCP является проект Web100 [6], разработанным и поддерживаемым командой Питтсбургского Центра Супервычислений. Web100 предоставляет доступ к расположенным в ядре переменным протокола TCP, позволяя считывать (и изменять) их. Знание динамики изменения параметров, таких как CurCwnd (текущее окно перегрузки), NonRecovDA (количество дубликатов подтверждений), RTTVar (значение RTT) и многих других, позволяет выявить первопричину недостаточной производительности и скорректировать стратегию управления перегрузкой.

Web10G [7]. Web10G, стартовал в 2011 году, является последней реализацией проекта Web100, который состоит из двух основных компонентов. Первый компонент – набор модификаций ядра Linux, которые экспортируют измерения TCP, переменные и настройки через Linux ‘/proc’ интерфейс. Он основан на наборе инструментов ядра (Kernel Instruments Set – KIS). Второй основной компонент Web100 – графическое пользовательское приложение – Инструмент Диагностики Разработчика (Diagnostic Tool Builder – DTB), который обеспечивает интерфейс для инструментария TCP Web100 в форме числовых индикаторов, гистограмм и круговых диаграмм значений данных, полученных Web100.

Наиболее существенным результатом этого проекта является разработанная в дополнении к KIS база управляющей информации TCP Extended Statistics MIB (RFC 4898, May 2007, M. Mathis, J. Heffner, R.). На рисунке 3 представлена структура проекта Web100, иллюстрирующая расположение и взаимодействие инструментария Web100 с сетевым стеком ядра Linux.

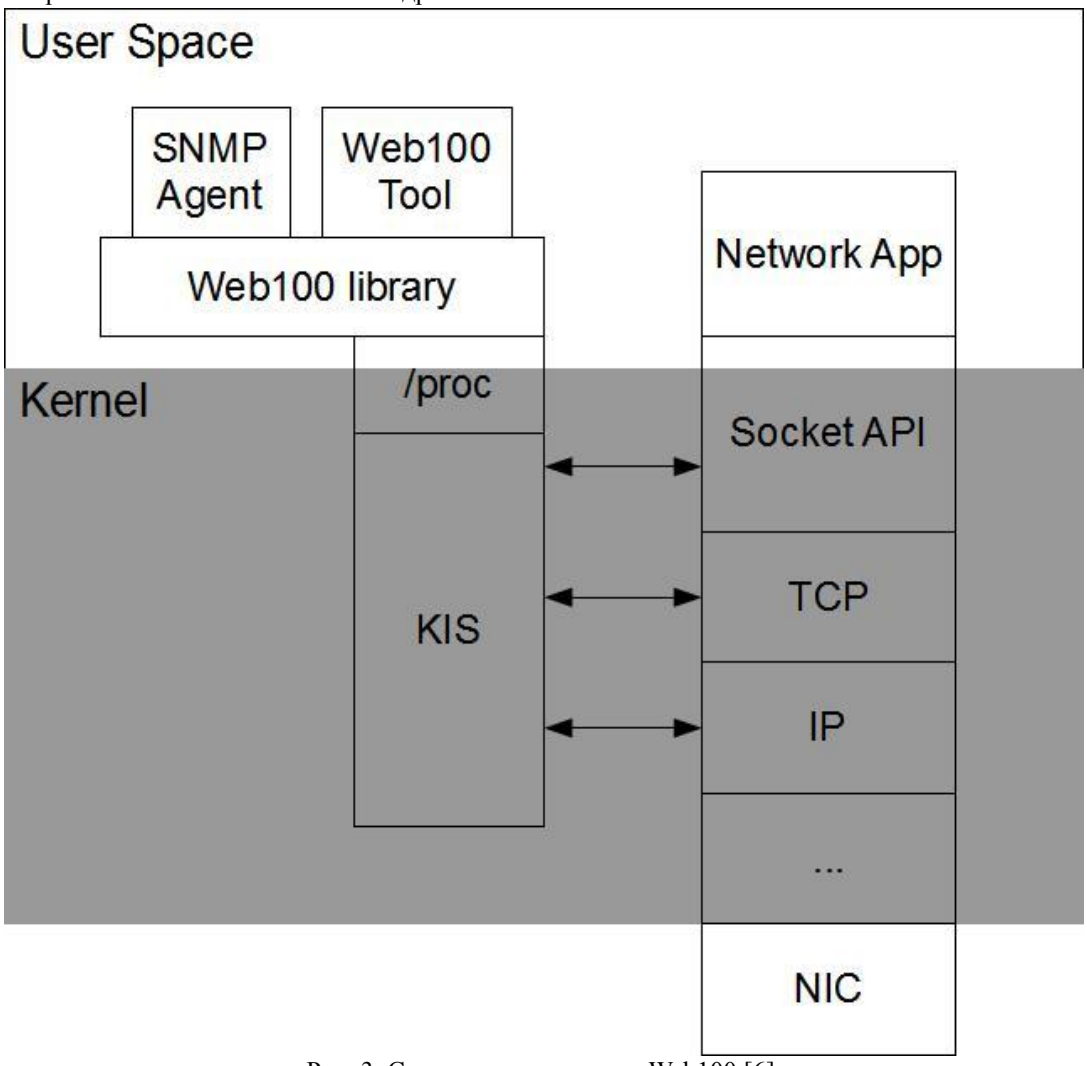

Рис. 3. Структура реализации Web100 [6].

Цель Web10G – обновления исходного кода ядра (доступного по адресу http://www.kernel.org/pub/linux/kernel/) и перемещения ABI ядра от /proc интерфейса к Netlink интерфейсу.

Web10G отделяет TCP инструментарий от /proc ABI (Application Binary Interface). Новый ABI реализуется как dlkm (Dynamically Loadable Kernel Module – динамически загружаемые модули ядра). У этого подхода есть много преимуществ, все из которых происходят от возможности поддерживать множественное ABI

и переключатель между ними по желанию. Прежний /proc ABI сохраняется для обратной совместимости. Тем не менее, использование dlkm позволяет пользователям отключать инструментарий и разрабатывать нестандартный ABI, и предоставляет гибкость разработчикам ABI. Новый ABI использует Netlink/Genetlink фреймворк, представленный в ядре. Netlink – IPC механизм, для связки процессов пространства пользователя и процессов уровня ядра. Netlink был разработан, чтобы стать более гибким преемником ioctl для обеспечения, главным образом, сетевой связки ядра и интерфейса контроля.

Конечной целью проекта Web10G является внедрение инструментария TCP в основной код ядра Linux так, чтобы он был распространен во все версии Linux и доступен всем пользователям. Однако самый важный шаг – включение в основной код Linux – не находится под прямым контролем разработчиков проекта и не может быть гарантирован. Поэтому основная стратегия разработчиков Web10G состоит в том, чтобы получить: качественный код, для включения в основной код Linux; комплект диагностических и измерительных инструментов с открытым исходным кодом; сообщество активных пользователей, которые желают получить эти инструменты для широкого спектра использования и могут подтвердить ценность инструментария остальной части сообщества Linux.

Инсталляция. Установку Web10G можно провести по инструкции для старых версий инструментария без использования Netlink (версия ядра Linux 3.0 или ниже), либо по инструкции, предоставляемой с более новой версией инструментария (версия ядра Linux 3.1 или выше). Инструментарий с патчем ядра доступен для скачивания по адресу: http://web10g.org/index.php?option=com_remository&Itemid=65&func=select&id=4. Была проведена установка двух версий инструментария: (1) ядро Linux 3.0.24 + инструментарий без Netlink; (2) ядро Linux 3.5.7 + инструментарий с Netlink.

В API инструментарии без использования Netlink использована возможность создания приложений для записи динамики изменения параметров соединения в лог-файл, а также чтения лог-файлов с этими данными.

Установка Web10G проводилась на компьютер P (рис. 1), со следующими характеристиками: CPU – Intel Core i3-2310M – 2x2,1GHz; RAM – 8Gb DDR3; Ethernet Controller – Atheros AR8151 GE Controller. Параметры HostY: CPU Intel Xeon E5520 – 4x2,27GHz; RAM – 18Gb DDR3; Ethernet Controller – Intel Corporation 82576 Gigabit Network Connection.

5. Измерения.

Все представленные в статье end-to-end измерения проводились по гарантированному каналу связи 1 Гбит/с между компьютером P и HostY (рис.1) . Сетевое конфигурирование позволяет добавлять или отключать существующий фоновый Интернет трафик. Интенсивность Интернет трафика оценена по MRTG и имеет величину не более 100 Мбит/с по причине его ограничения.

Импульсный поток на фоне Интернет трафика (рис. 4).

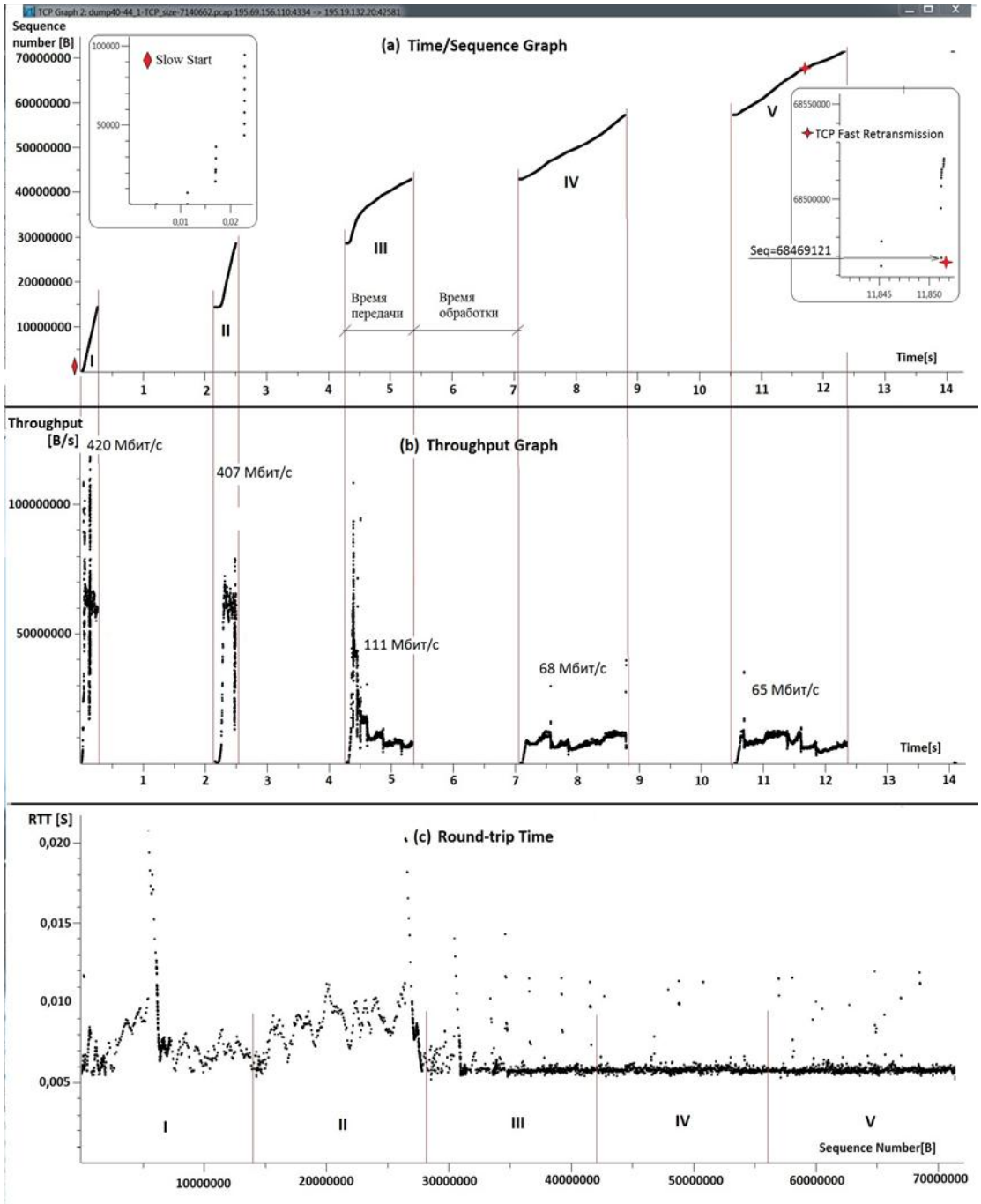

Рис. 4. Wireshark, импульсный трафик (Sequence number – порядковый номер байта в потоке)

На рис.4 приведен импульсный трафик, характерный для структурированного потока экспериментальных данных, передаваемого для обработки разным вычислительным узлам суперкомпьютера URAN [8]. Снятие, анализ и визуализация трафика проводились средствами Wireshark. Цель измерений — определение возможной потери темпа передачи при переходе от одной кванта времени передачи к другому. Наглядно видно уменьшение средней скорости передачи последовательности одинаковых порций данных, которое может быть объяснено хорошим дружелюбием Cubic к фоновому Интернет трафику. Проиллюстрировано сохранение темпа передачи в конце текущего и начале следующего интервалов передачи (макро анализ на графике не показан).

Множество параллельных потоков на фоне Интернет трафика. На рис. 5 показана измеренная зависимость производительности RENO и Cubic от числа параллельных TCP-соединений. Для измерения пропускной способности использована утилита Iperf. Видно, что Cubic эффективнее использует канал нежели Reno в диапазоне 2 - 16 параллельных потока. Количество параллельных потоков для Cubic рекомендуется делать не более 8, оставив полосу пропускания ~ 100 Мбит/с для Интернет трафика.

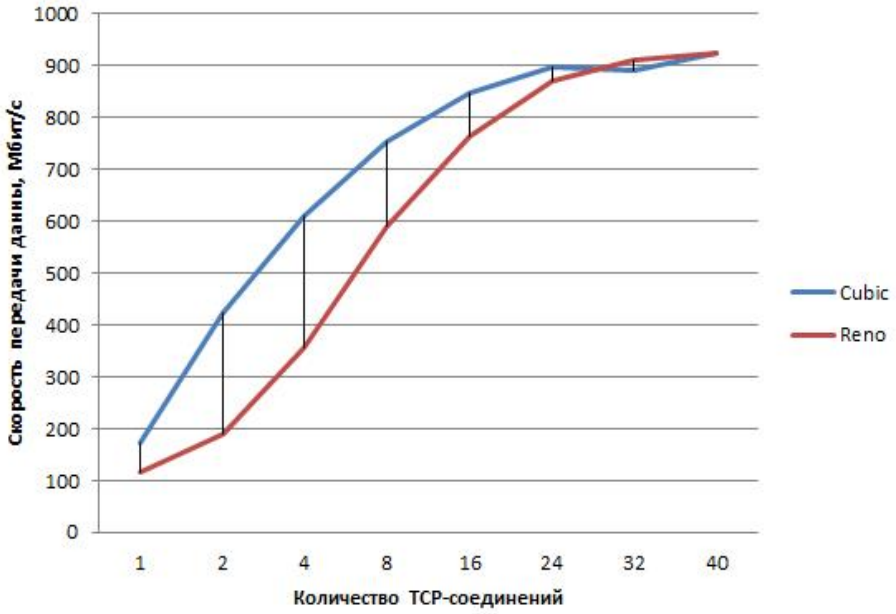

Рис. 5 – Зависимость скорости передачи данных от количества соединений

Один поток на фоне Интернет трафика (рис. 6). Для генерации трафика и измерения средней пропускной способности использована утилита Iperf. Параметр CurCwnd снимался с помощью Web10G. Результаты измерений: Cubic – 143 Мбит/с; Reno – 97,2 Мбит/с. На рис. 6 представлено сравнение работы алгоритмов управления перегрузкой Cubic и Reno. Видно, что стратегия управления перегрузкой Reno ведет себя более «осторожно» и не имеет большого числа перегрузок по сравнению с Cubic. Однако Cubic лучше Reno по скорости передачи данных. При этом один поток Cubic не использует доиспользует доступную полосу пропускания канала: 143Мбит/с (Cubic) + 100Мбит/с (Internet) < 1Гбит/с (канал связи)

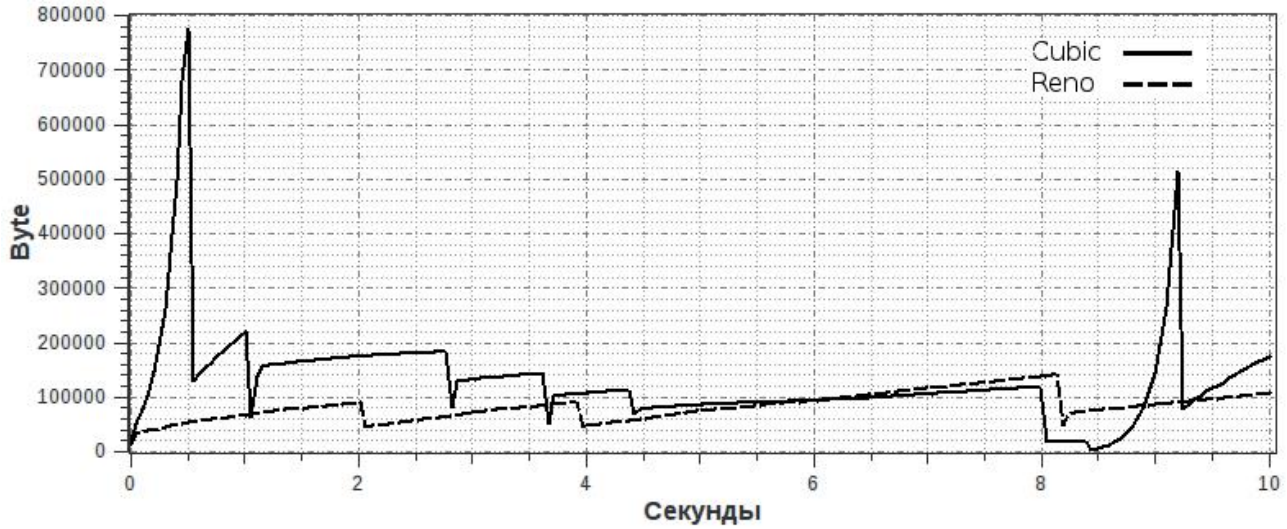

Рис. 6. Сравнение динамики параметра CurCwnd для Cubic и Reno.

Один поток без фонового Интернет трафика. На рис. 7 представлено сравнение работы алгоритмов управления перегрузкой BIC, Cubic и Illinois для непрерывного трафика. Используемый инструментарий: утилита Iperf для генерации трафика и Web10G для анализа и интерпретации. Условия: одно TCP соединение, гарантированный канал связи 1 Гбит/с с RTT=5мс (BDP=625000байт), фоновый интернет тьрафик отсутствует, периодичность снятия параметров 1мс, время измерения 1с, (на графике отображено 0,12с).

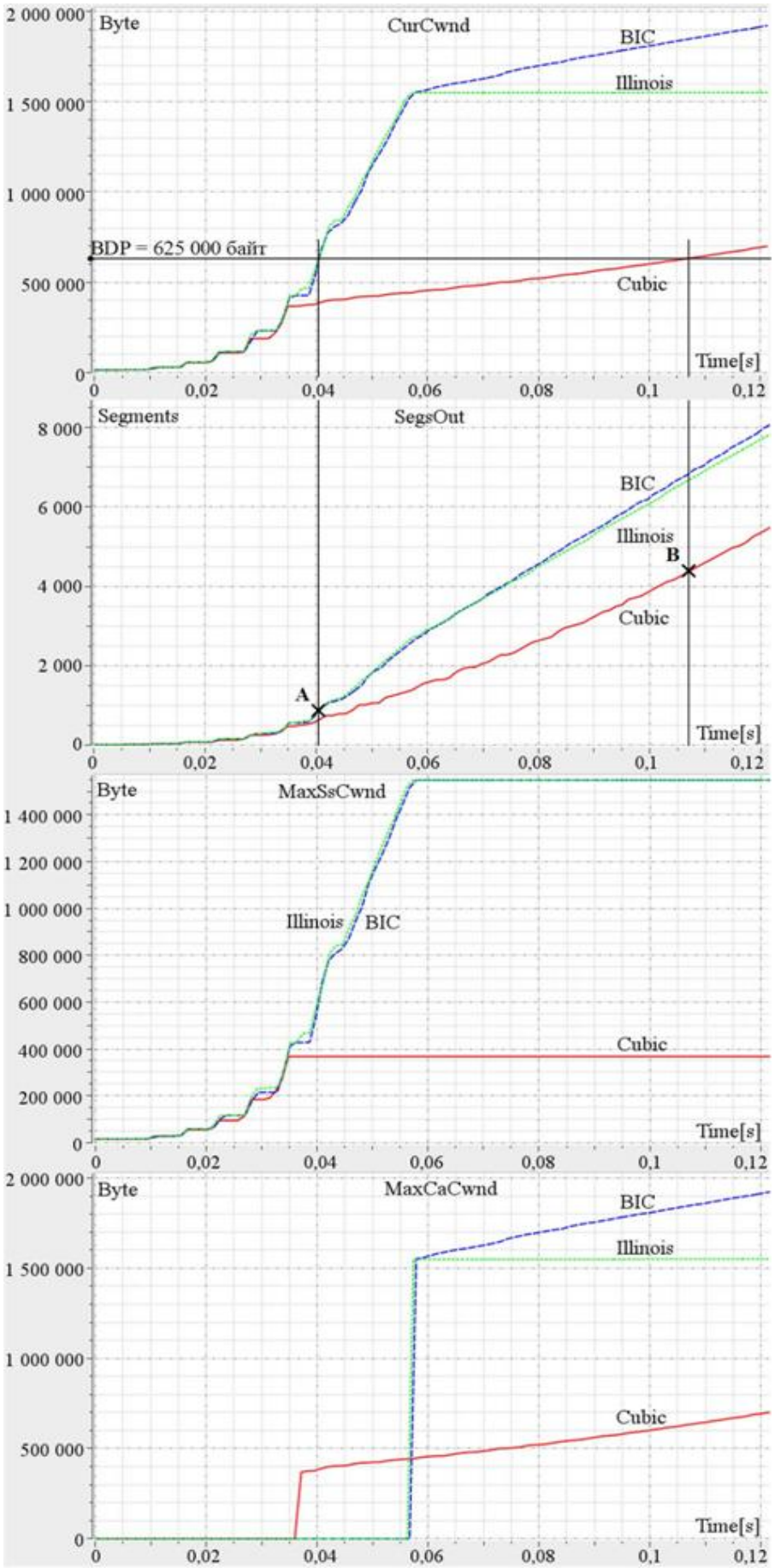

Рис. 7. Сравнение динамики изменения параметров TCP в области медленного старта

Описание параметров. CurCwnd - текущее значение окна перегрузки. SegsOut - общее количество переданных сегментов, наклон которой является текущей скоростью передачи. MaxSsCwnd – максимальное

окно перегрузки, используемое во время медленного старта (SS). MaxCaCwnd – максимальное окно перегрузки, используемое во время предотвращения перегрузки (CA). Интерпретация. Как видно из графиков все стратегии достигают максимальной скорости передачи после достижения CurCwnd ≥ BDP, что соответствует теории протоколов с обратной связью. Однако для BIC и Illinois это происходит в фазе SS в точке A на 40мс. A для Cubic в фазе CA в точке B на 110мс, поскольку MaxSsCwnd у Cubic на 35 мс завершил фазу SS и перешел в фазу CA, в которой темп увеличения CurCwnd происходит медленнее. Подтверждается негативное влияние фазы SS, которое лучше преодолевается в BIC и Illinous. Размер передаваемых данных при коротких TCP сессиях в должен быть существенно больше BDP.

6. Выводы.

Разработанная территориальная инфраструктура вычислителя существенно отличается от классических Grid, работающих через Public Internet, наличием распределенного Interconnect, соединяющим внутренние Interconnect кластеров и систем хранения данных. Созданная структура параллельных гарантированных каналов связи 1-10 Гбит/с позволяет создавать требуемые для исследователей end-to-end каналы связи.

Web10G инструментарий позволяет считывать переменные состояний TCP и объяснять истинные причины низкого темпа передачи. Желательное уменьшение времени фазы медленного старта возможно на созданной инфраструктуре путем увеличения cwnd более чем на SMSS байтов для входящих подтверждений. Однако принципиальным решением проблемы медленного старта при изначально известной гарантированной скорости в end-to-end канале связи и отсутствии фонового трафика в нем является установка начального значения параметра cwnd в оконечных системах равным BDP. Верхнюю границу числа параллельных конкурирующих потоков на фоне Интернет трафика в канале 1 Гбит/с с RTT=5мс для Cubic не целесообразно делать более 8. Как и ожидалось, в L2 каналах связи порядок следования пакетов не нарушается и не приводит к появлению ложных SACK. При передаче импульсного трафика в TCP соединении темп передачи в конце текущего и начале следующего интервалов передачи сохранятся.

Методология оценки эффективности параллелизма потоков данных должна содержать следующие компоненты. (1) Список и описание системных служб оконечных систем, (2) описание сетевой инфраструктуры, связывающей оконечные системы, (3) перечень и параметры нагрузки, (4) оцениваемые метрики параллельных потоков, (5) измерительные средства, (6) метод анализа и интерпретации данных, (7) представление результатов.

ЛИТЕРАТУРА:

1. Масич А.Г., Масич Г.Ф. Инициатива GIGA UrB RAS // Вычислительные технологии. 2008. Т. 13. Вестник КазНУ им. Аль-Фараби. Серия математика, механика, информатика 2008. №3(58). Совместный выпуск. - Ч.II. -С.413-418.
2. А.Г. Масич, Г.Ф. Масич. От «Инициативы GIGA UrB RAS» к Киберинфраструктуре УрО РАН. // Вестник Пермского научного центра (октябрь-декабрь 4/2009). – Пермь: Изд-во ПНЦ УрО РАН, 2009. С. 41-56
3. Paola Grosso, Damien Marchal, Jason Maassen, Eric Bernier, Li Xu, Cees de Laat. Dynamic photonic lightpaths in the StarPlane network // Future Generation Computer Systems, vol. 25, no. 2, 2009. Pages: 132-136.
4. M. Mathis. URL: http://staff.psc.edu/mathis/ (14.05.2013)
5. Alphabetical List of Software Available on PSC Systems. URL: http://www.psc.edu/index.php/users/software (14.05.2013).
6. Tom Hacker, Brian Athey, and Jason Sommerfield. "Experiences Using Web100 for End-To-End Network Performance Tuning." URL:http://web100.org/docs/ExperiencesUsingWeb100forHostTuning.pdf (14.05.2013).
7. Chris Rapier. Web10G: TCP Extended Statistics. URL: http://web10g.org/index.php?option=com_remository&Itemid=65&func=startdown&id=51 (14.05.2013).
8. В.А.Щапов, А.Г.Масич, Г.Ф.Масич. Модель потоковой обработки экспериментальных данных в распределенных системах // Вычислительные методы и программирование. 2012. Раздел 2. С. 139-145.