РОССИЙСКАЯ ФЕДЕРАЦИЯ

(19) **RU** (11) **2 635 220**(13) **C2**

(51) МПК
*H04L 29/08* (2006.01)

ФЕДЕРАЛЬНАЯ СЛУЖБА
ПО ИНТЕЛЛЕКТУАЛЬНОЙ СОБСТВЕННОСТИ

# (12) ОПИСАНИЕ ИЗОБРЕТЕНИЯ К ПАТЕНТУ

(21)(22) Заявка: **2015143010, 21.04.2014**

(24) Дата начала отсчета срока действия патента:
**21.04.2014**

Дата регистрации:
**09.11.2017**

Приоритет(ы):
(30) Конвенционный приоритет:
**23.04.2013 GB 1307340.8**

(43) Дата публикации заявки: **26.05.2017** Бюл. № **15**

(45) Опубликовано: **09.11.2017** Бюл. № **31**

(85) Дата начала рассмотрения заявки PCT на национальной фазе: **23.11.2015**

(86) Заявка PCT:
**EP 2014/001052 (21.04.2014)**

(87) Публикация заявки PCT:
**WO 2014/173521 (30.10.2014)**

Адрес для переписки:
**191036, Санкт-Петербург, а/я 24, "НЕВИНПАТ"**

(72) Автор(ы):
**КАРККАИНЕН Туомас Микаел (FI),**
**ХАККАРАЙНЕН Валттери (FI),**
**КАЛЕВО Осси (FI)**

(73) Патентообладатель(и):
**Гурулоджик Микросистемс Ой (FI)**

(56) Список документов, цитированных в отчете о поиске: **US 6892240 B1, 10.05.2005. US 2010/0042677 A1, 18.02.2010. US 2002/0156901 A1, 24.10.2002. US 2005/0246442 A1, 03.11.2005. RU 2417532 C2, 27.04.2011.**

RU 2635220 C2

## (54) Система двухсторонней связи в реальном времени с использованием протокола HTTP

(57) Реферат:

Изобретение относится к средствам установления канала связи посредством системы связи, выполненной с возможностью поддержки связи в соответствии с протоколом HTTP. Технический результат заключается в обеспечении двухсторонней связи в реальном времени с уменьшенным временем ожидания передачи данных. Используют систему связи для установления двухстороннего канала связи в реальном времени между двумя узлами системы посредством использования комбинации методов GET и POST, связанных с протоколом HTTP. Обменивают данные по каналу связи в форме порций и/или последовательности составных блоков данных. Устанавливают максимальный размер сегмента (MSS) для порций данных и/или составных блоков данных, передаваемых по каналу связи, равным максимальному размеру сегмента (MSS) для самого слабого канала передачи данных в сети связи, поддерживающей канал связи, при этом обмен данными по каналу (40) связи осуществляют с использованием кодирования передачи в виде порций данных и/или последовательности составных блоков данных. 3 н. и 9 з.п. ф-лы, 3 ил., 1 табл.

RU 2635220 C2


Фиг.2

RU 2635220 C2

RU 2635220 C2

RUSSIAN FEDERATION

FEDERAL SERVICE
FOR INTELLECTUAL PROPERTY

(19) **RU** (11) **2 635 220**(13) **C2**

(51) Int. Cl.
*H04L 29/08* (2006.01)

## (12) ABSTRACT OF INVENTION

(21)(22) Application: **2015143010, 21.04.2014**

(24) Effective date for property rights:
**21.04.2014**

Registration date:
**09.11.2017**

Priority:
(30) Convention priority:
**23.04.2013 GB 1307340.8**

(43) Application published: **26.05.2017** Bull. № **15**

(45) Date of publication: **09.11.2017** Bull. № **31**

(85) Commencement of national phase: **23.11.2015**

(86) PCT application:
**EP 2014/001052 (21.04.2014)**

(87) PCT publication:
**WO 2014/173521 (30.10.2014)**

Mail address:
**191036, Sankt-Peterburg, a/ya 24, "NEVINPAT"**

(72) Inventor(s):
**KARKKAINEN Tuomas Mikael (FI),**
**HAKKARAINEN Valtteri (FI),**
**KALEVO Ossi (FI)**

(73) Proprietor(s):
**Gurulodzhik Mikrosistems Oj (FI)**

RU 2635220 C2

### (54) TWO-WAY COMMUNICATION SYSTEM IN REAL TIME, USING HTTP PROTOCOL

(57) Abstract:

FIELD: radio engineering, communication.

SUBSTANCE: use the communication system to set the two-way communication channel in the real time between two system nodes by using the combination of the GET and POST methods, connected with the HTTP protocol. Exchange the data through the communication channel in the form of chunks and/or sequence of component data blocks. Set the maximum segment size (MSS) for the data chunks and/or component data blocks, transmitted through the communication channel equal to the maximum segment size (MSS) for the weakest data transmission channel in the communication network, supporting the communication channel, while data exchange through the channel (40) is performed, using transmission coding in the form of data chunks and/or the series of component data blocks.

EFFECT: provision of two-way communication in real time with reduced waiting time for data transmission.

12 cl, 3 dwg, 1 tbl

RU 2635220 C2


Фиг.2

RU 2635220 C2

RU 2635220 C2

Область техники

Настоящее изобретение относится к системам связи, например к системам связи, которые используют протокол HTTP реального времени (Real-Time Hypertext Transfer Protocol) для обмена данными различного типа, например графическими данными, данными изображений, видеоданными, аудиоданными и т.п. Настоящее изобретение относится также к способам функционирования указанных систем связи для обмена различными видами данных. Настоящее изобретение относится также к программным продуктам, хранящимся на машиночитаемых носителях информации, при этом программные продукты исполняются на компьютерном оборудовании для осуществления указанных способов.

Предпосылки создания изобретения

Протокол HTTP широко используется для реализации современной сети Интернет. Этот протокол является протоколом уровня приложения для распределенных совместных информационных гипермедиа-систем. На практике протокол HTTP представляет собой многолинейный набор объектов, обеспечивающих построение сети с использованием логических ссылок для задания сети. Эти ссылки, часто называемые гиперссылками, определяют сетевую взаимосвязь между узлами.

Протокол HTTP функционирует как протокол типа «запрос-ответ», например в модели клиент-сервер, как реализовано для сети Интернет. В этой модели может использоваться веб-браузер для реализации клиента, при этом программное приложение, исполняемое на сервере, размещает веб-сайт. В процессе работы конкретный клиент посылает сообщение с запросом HTTP на сервер, который отвечает, обеспечивая ресурсы, такие как файлы HTML и другой контент, или выполняет функции обработки данных от имени клиента, или возвращает клиенту ответное сообщение. Указанный веб-браузер может быть реализован различными способами, например как агент пользователя, поисковый агент или любая другая программа, исполняемая на компьютерном оборудовании, которая получает доступ, потребляет или отображает контент, полученный из сети Интернет.

Протокол HTTP разработан для того, чтобы разрешить промежуточным сетевым элементам осуществлять связь между клиентами и серверами. Веб-сайты сети Интернет с большим трафиком часто используют кэшевые веб-серверы, которые способны доставлять контент от имени верхних серверов для сокращения времени отклика для данных и/или доставки сервисов. Помимо этого, предпочтительно используются прокси-серверы HTTP на границах частных сетей для улучшения связи для клиентов без глобального IP адреса, в частности путем пересылки сообщений через внешние серверы.

Ресурсы HTTP идентифицируются и размещаются в конкретной сети с использованием универсальных идентификаторов ресурсов (URI, uniform resource identifier), называемых также универсальными указателями ресурса (URL, uniform resource locator). Помимо этого, идентификаторы URI и гиперссылки выражены на языке разметки гипертекста (HTML, hypertext markup language), с помощью которого можно создавать веб-страницы из взаимосвязанных гипертекстовых документов.

Сеанс HTTP осуществляется путем последовательности сетевых транзакций запросов-ответов. Например, клиент HTTP инициирует запрос путем установления соединения протокола управления передачей (TCP, transmission control protocol) с конкретным портом сервера. Сервер HTTP прослушивает сообщение с запросом клиента и отвечает, посылая строку состояния, например HTTP/1.1 200 OK, вместе с соответствующим сообщением. Телом этого сообщения обычно является запрашиваемый ресурс, хотя может быть возвращено сообщение об ошибке.

Протокол HTTP определяет методы, условно называемые «глаголами», для указания требуемой операции, которую нужно выполнить в отношении указанного ресурса. Ресурс может представлять собой, например, файл данных или вывод из исполняемого объекта, размещаемого на одном или более серверах. Примеры методов HTTP, называемых также «глаголами» HTTP, приводятся в таблице 1.

Таблица 1. Методы HTTP («глаголы» HTTP).

| Глагол | Описание |
|---|---|
| GET | Запрашивает представление конкретного ресурса, при этом запрос, использующий GET, может только извлекать данные. |
| HEAD | Запрашивает ответ, который является идентичным получаемому с помощью GET, но не имеет тела ответа. HEAD часто используется для эффективного извлечения метаданных. |
| POST | Запрашивает, чтобы данный сервер принял объект, вложенный в запрос, в качестве нового подчиненного объекта данного веб-ресурса, идентифицированного по URL. |
| PUT | Запрашивает, чтобы вложенный объект был сохранен в отношении указанного URI (URL). Если URI относится к уже существующему ресурсу, то этот ресурс изменяется. |
| DELETE | Запрашивает удаление указанного ресурса. |
| TRACE | Показывает результат в принятом запросе, на который отвечают данному клиенту. |
| OPTIONS | Возвращает методы HTTP, поддерживаемые сервером, связанным с данным URL. |
| CONNECT | Преобразовывает запрашиваемое соединение в прозрачный туннель TCP/IP, например для обеспечения связи с шифрованием TLS и SSL (HTTPS) через нешифрованный прокси-сервер, как указано выше; по умолчанию соединение HTTP не шифруется, а соединение HTTPS шифруется. |
| PATCH | Запрашивает применение частичных изменений для данного ресурса. |

Таким образом, основным протоколом передачи, используемым современными веб-браузерами, является протокол HTTP; при этом несколько связанных «экосистем» и программное обеспечение, которое они используют, в частности программные приложения браузера, не способны функционировать без использования протокола

HTTP. Как указано выше, протокол HTTP основан на передаваемых запросах (см. Таблицу 1) и на ответах на эти запросы, при этом страницы HTML или двоичные данные, такие как изображения или аудиопотоки/файлы обычно предоставляются в ответ на получение этих запросов.

Вследствие сложности сети Интернет могут возникать запаздывания при передаче, то есть задержки. Такие задержки могут вызывать проблемы в ситуациях, когда требуется обмен данными, например когда требуется двухсторонняя (полнодуплексная) связь или когда требуется ответ в реальном времени, например передача и прием видеоизображений и/или звука с очень малой задержкой. Двунаправленная передача через Интернет является известной из протокола передачи речи по сети Интернет (VoIP, Voice-over-Internet-Protocol) и видеоконференцсвязи на основе сети Интернет, например, как обеспечивается современным программным обеспечением Skype. Skype является зарегистрированным товарным знаком.

Известно использование протоколов, называемых WebSocket и описанных на сайте http://tools.ietf.org/html/rfc6455. для обращения к услугам связи определенного типа. Могут быть достигнуты следующие свойства связи:

(i) WebSocket используется внутри туннеля HTTP/HTTPS, в таком случае межсетевые экраны уже открыты для портов 80/443, поскольку в настоящее время они обычно используются в веб-браузерах;

(ii) WebSocket используется в режиме полнодуплексного соединения, при этом только одно соединение TCP способно осуществлять связь в двух направлениях в реальном времени, а именно способно передавать и принимать данные по одному соединению путем изменения направления передачи данных.

Однако протоколы WebSocket могут зависеть от порта, что является нежелательным ограничением.

Сущность изобретения

В настоящей заявке предлагается система связи, которая способна обеспечивать улучшенную двухстороннюю передачу данных по сети связи HTTP.

Кроме того, в настоящей заявке предлагается усовершенствованный способ функционирования системы связи для обеспечения двухсторонней передачи данных по сети связи HTTP.

Согласно первому аспекту настоящего изобретения, предлагается система связи, выполненная с возможностью поддержки связи в соответствии с протоколом HTTP и с возможностью установления двухстороннего канала связи в реальном времени между двумя узлами системы путем использования комбинации методов GET и POST, связанных с протоколом HTTP, причем обмен данными по каналу связи осуществляется порциями и/или в виде последовательности составных блоков данных, отличающаяся тем, что максимальный размер сегмента (MSS, maximum segment size) для порций данных и/или составных блоков данных, передаваемых по каналу связи, оптимизируется в зависимости от способности сети связи поддерживать канал связи, при этом обмен данными по каналу связи осуществляется с использованием кодирования передачи в виде порций данных и/или последовательности составных блоков данных.

Такая система связи имеет преимущество, заключающееся в способности обеспечивать двухстороннюю связь в реальном времени с уменьшенным временем ожидания.

Дополнительно, метод CONNECT может использоваться в трех различных типах сценариев:

(i) соединение туннелируется к объекту назначения, что является сценарием по умолчанию;

(ii) соединение туннелируется через локальный хост к объекту назначения, в результате чего данные передаются из процесса передачи в локальной службе в пересылающий прокси-процесс, откуда данные передаются в объект назначения; такой способ является предпочтительным, поскольку способен предотвратить анализ данных антивирусной программой и непреднамеренное блокирование, или другое вмешательство в данные;

(iii) соединение туннелируется к пересылающему прокси-серверу, который затем перенаправляет данные в объект назначения; такой подход предпочтительно используется в системах с балансированием нагрузки, в частности в системах, где нагрузка сети, вызванная клиентами, оптимально распределяется до объекта назначения. Например, данные быстрее передаются по магистральной сети, чем по прямому соединению.

Дополнительно, в системе связи канал связи включает соединение для приема и соединение для передачи для обеспечения двухсторонней связи, причем эти соединения остаются открытыми до тех пор, пока не будут приняты пустая порция данных и/или пустой составной блок данных.

Дополнительно, в системе связи канал связи выполнен с возможностью использования шифрования передаваемых по нему данных.

Дополнительно, в системе связи канал связи выполнен с возможностью обеспечения передачи по меньшей мере одного из следующего: графических данных, данных изображения, видеоданных, аудиоданных, неструктурированных данных.

Согласно второму аспекту настоящего изобретения, предлагается способ установления канала связи посредством системы связи, выполненной с возможностью поддержки связи в соответствии с протоколом HTTP, отличающийся тем, что данный способ включает:

(a) использование системы связи для установления двухстороннего канала связи в реальном времени между двумя узлами системы путем использования комбинации методов GET и POST, связанных с протоколом HTTP;

(b) обмен данными по каналу связи в виде порций данных и/или последовательности составных блоков данных и

(c) оптимизацию размера MMS для порций данных и/или блоков данных, передаваемых по каналу связи, в зависимости от способности сети связи поддерживать канал связи;

при этом обмен данными по каналу связи осуществляют с использованием кодирования передачи в виде порций данных и/или последовательности составных блоков данных.

Дополнительно, в данном способе канал связи включает соединение для приема и соединение для передачи для обеспечения двухсторонней связи, причем эти соединения остаются открытыми до тех пор, пока не будут приняты пустая порция данных и/или пустой составной блок данных.

Дополнительно, в данном способе канал связи выполнен с возможностью использования шифрования передаваемых по нему данных.

Дополнительно, в данном способе канал связи выполнен с возможностью обеспечения передачи по меньшей мере одного из следующего: графических данных, данных изображения, видеоданных, аудиоданных, текстовых данных, неструктурированных данных.

Согласно третьему аспекту настоящего изобретения, предлагается программный продут, хранящийся на машиночитаемом носителе информации и отличающийся тем,

что он исполняется посредством компьютерного аппаратного обеспечения для осуществления способа в соответствии со вторым аспектом настоящего изобретения.

Дополнительно, программный продукт выполнен в соответствии с протоколами HTTP и исполняется на сервере сети связи, работающей по протоколу HTTP.

Настоящее изобретение имеет преимущество в том, что данная система связи способна обеспечивать двухстороннюю полнодуплексную связь, с шифрованием или без шифрования, посредством использования известного протокола передачи HTTP, так что не требуется дополнительная конфигурация программных или аппаратных межсетевых экранов и/или антивирусных программных приложений, исполняемых в системе связи.

Помимо этого, настоящее изобретение имеет преимущество, которое заключается в улучшении функциональности и надежности приложений связи, и, таким образом, в упрощении технического обслуживания системы, например настройки безопасности данных.

Следует понимать, что технические признаки изобретения могут комбинироваться различным образом в пределах сущности изобретения, определяемой прилагаемой формулой изобретения.

Краткое описание чертежей

Далее варианты осуществления настоящего изобретения описываются посредством примеров со ссылкой на приложенные чертежи.

На фиг. 1 показана сеть связи, работающая по протоколу HTTP

На фиг. 2 показан набор шагов способа согласно изобретению.

На фиг. 3 показан другой набор шагов способа согласно изобретению.

На приложенных чертежах подчеркнутое число используется для представления элемента, на котором это подчеркнутое число располагается, или элемента, рядом с которым располагается это подчеркнутое число. Неподчеркнутое число относится к элементу, определяемому линией, которая связывает это неподчеркнутое число с элементом. Если число не подчеркнуто и сопровождается стрелкой, это неподчеркнутое число используется для обозначения общего объекта, на который указывает стрелка.

Подробное описание изобретения

Далее со ссылкой на фиг. 1 описывается система, часть которой обозначена цифрой 5, а также соответствующий способ, который способен уменьшить задержки, а именно «время ожидания» в отношении протокола HTTP для двухсторонней связи в реальном времени таким образом, что описание используемого протокола HTTP соответствует стандартам, таким как RFC2621, RFC2068 и RFC1945. В общем, протокол HTTP не предназначен для осуществления двухсторонней связи в реальном времени между первым и вторым узлами 10A, 10B, при этом конкретный клиент способен одновременно передавать и принимать данные в реальном времени, так что:

(i) соединение 20 связи, используемое между двумя узлами 10A, 10B, сконфигурировано для поддержки двухсторонней связи в шифрованном формате;

(ii) программа 30 защиты от вирусов не влияет на контент 40, передаваемый и принимаемый по соединению 20 связи;

(iii) межсетевые экраны 60 не способны препятствовать сетевому трафику, пока не будет полного блокирования трафика сети Интернет, в частности блокируется «WWW трафик», например в случае соединения между банками, используемого для защищенных финансовых операций;

(iv) сетевые устройства, например мосты и маршрутизаторы, не способны анализировать и вмешиваться в данные, подлежащие передаче по соединению 20 связи.

Варианты осуществления настоящего изобретения способны воплощать функциональные возможности от (i) до (iv) посредством следующего:

(a) используются разные виды методов, GET и POST (см. Таблицу 1), при этом метод GET устанавливает соединение для приема по соединению 20, а метод POST устанавливает соединение для передачи по соединению 20;

(b) оба соединения туннелируются с использованием метода CONNECT, как используется в современном протоколе HTTP;

(c) применяется механизм кодирования «порциями» или составной передачи, как подробно описано ниже.

Традиционно протокол HTTP используется для сеансов сети Интернет, при этом методы GET и POST используются независимо друг от друга. Например, метод GET используется для запроса контента HTML от веб-сервера, который работает в качестве хоста для веб-браузера клиента, при этом соединения для метода GET остаются открытыми, пока все ответные данные не будут доставлены от хоста клиенту. Кроме того, используется процедура соединения, которая является такой же, как в методе POST (см. Таблицу 1), за исключением того, что данные доставляются от клиента в хост.

В вариантах осуществления изобретения, раскрываемых далее, связь осуществляется так, что сокет используется в полудуплексном режиме, что отличает изобретение от известных подходов, например от вышеуказанной технологии WebSocket. В вариантах осуществления изобретения передача и/или прием данных являются более эффективными, чем при полнодуплексном соединении, поскольку сетевым интерфейсным картам не требуется переключать состояния ввода/вывода (I/O) между приемом и передачей. Такое переключение, используемое в известном уровне техники, потребляет системные ресурсы и соответственно уменьшает потенциальную скорость связи.

В вариантах осуществления изобретения, раскрываемых далее, сокет используется только после инициализации методов HTTP GET и POST, в режиме приема или в режиме передачи. Вследствие этого, используемому сетевому адаптеру требуется работать только в полудуплексном режиме, и, таким образом, экономятся ресурсы устройства и инфраструктуры сети, поскольку соединение работает исключительно либо в режиме передачи, либо в режиме приема после согласования заголовков методов HTTP GET и/или POST и до завершения соединения. Помимо этого, имеются также другие преимущества, например межсетевые экраны и маршрутизаторы, а именно концентраторы и коммутаторы, получают меньше нагрузки, связанной с переключениями, и поэтому не отказывают так быстро, как в известных современных полнодуплексных способах связи, в которых используется только одно полнодуплексное соединение. Таким образом, раскрываемые здесь варианты осуществления изобретения являются гораздо более эффективными в отношении ресурсов, чем, например, протокол WebSocket.

Известный протокол WebSocket может быть легко проанализирован межсетевыми экранами, отнесен к неопознанному типу соединения и отключен, что затрудняет или ограничивает его использование, независимо от того, туннелируется или нет соответствующее соединение. В вариантах осуществления изобретения, описываемых далее, соединение GET или POST функционирует в соответствии с протоколом HTTP, поэтому межсетевые экраны не могут ограничивать или прекращать связь, использующую эти методы.

В вариантах осуществления изобретения, как описано ниже, предпочтительно используется протокол UDP, который практически в три раза быстрее, чем протокол

TCP. Опционально, варианты осуществления изобретения могут использовать одноранговые соединения (P2P, pear-to-pear), которые позволяют осуществлять связь на уровне приложения.

Раскрываемые варианты осуществления изобретения отличаются от общеизвестных реализаций HTTP тем, что известные реализации HTTP не имеют связи между способами GET и POST, а раскрываемые варианты осуществления изобретения, наоборот, используют способы GET и POST, соединенные вместе новым способом для обеспечения передачи данных в полнодуплексном режиме в реальном времени. Указанная полнодуплексная передача данных осуществляется путем использования одного соединения для приема и одного соединения для передачи. Соединение для приема и соединение для передачи могут использовать полудуплексный режим соединения или полнодуплексный режим соединения.

Хотя варианты осуществления изобретения описываются далее на основе протокола TCP, необходимо отметить, что в качестве альтернативы может использоваться протокол UDP (User Datagram Protocol). Несмотря на то что оба протокола UDP и TCP опираются на лежащий в их основе протокол IP (Internet Protocol), и датаграмма UDP, и фрагмент TCP передаются в IP-пакете, протокол UDP отличается тем, что он представляет собой протокол без установления соединения, что позволяет осуществлять одноранговые соединения между приложениями не только в локальной сети (LAN, local area network), но также во внешней сети Интернет посредством технологии преобразования сетевых адресов NAT (Network Address Translation). При использовании такого подхода можно избежать необходимости передавать данные через серверы в системе 5, что приводит к значительной экономии ресурсов сети связи. Дополнительным преимуществом при использовании протокола UDP в системе 5 является то, что он практически в три раза эффективнее в использовании ресурсов сети связи, чем протокол TCP, поскольку протокол UDP не является управляемым протоколом. Кроме того, размер MSS, измеряемый в байтах в сетях IPv4 и IPv6, например используемых для реализации системы 5, больше, поскольку заголовки UDP меньше, чем соответствующие заголовки TCP.

Хотя далее описывается использование протокола TCP для обоих соединений GET и POST, следует отметить, что в некоторых случаях только одно из этих соединений использует протокол TCP, а другие соединения используют протокол UDP Кроме того, следует отметить, что оба соединения GET и POST могут использовать протокол UDP.

Следует отметить, что согласно настоящему изобретению данные на передающем или приемном конце также могут изменяться с данных с коммутацией каналов на IP-данные и соответственно с IP-данных на данные с коммутацией каналов.

В первом варианте осуществления изобретения выполняют ряд шагов, как показано на фиг. 2.

Шаг 1 (S1): клиент для соединения передачи данных формирует уникальный идентификатор потока (ID), который используется для сочетания методов GET и POST таким образом, чтобы сервер, используемый для осуществления соединения, знал, что эта пара методов GET и POST принадлежит одному клиенту. Используемый идентификатор ID будет далее рассмотрен более подробно. Однако следует отметить, что методы GET и POST не ограничивают настоящее изобретение, когда используется уникальная идентификация потока для объединения соединения для передачи и соединения для приема. Несмотря на то, что основной целью идентификатора ID потока является объединение клиентских соединений для передачи и для приема на сервере, он может одновременно использоваться для аутентификации и идентификации клиента.

Это означает, что сервер может отбрасывать вредоносные, ошибочные или неопознанные соединения прежде, чем продолжится их обработка. Такие функциональные возможности позволяют защитить сервер и снизить/сократить нагрузку на сервер, вызванную неопознанными запросами на соединение и ненужными вычислениями. Другими словами, это позволяет системе экономить ресурсы, что обеспечивает экономию энергии и уменьшение числа серверов, которые требуются для серверного оборудования, особенно в системах с балансированием нагрузки.

Шаг 2 (S2): клиент устанавливает два соединения TCP/IP с сервером, например с его портом по умолчанию «80», после чего клиент передает заголовок, связанный с методом CONNECT. В процессе работы метод CONNECT преобразовывает требуемое соединение передачи данных в прозрачный туннель TCP/IP, обычно, например, для того, чтобы обеспечить связь (HTTP) с шифрованием TLS и SSL через нешифрованный прокси-сервер, как упоминалось ранее.

При осуществлении шагов 1 и 2 опционально используются различные виды шифрования, например SSL 1.0, SSL 2.0, SSL 3.0, TLS 1.0, TLS 1.1, TLS 1.2, или шифрование аналогичного типа. При этом указанный туннель предпочтительно является прозрачным для обеспечения безопасной связи между различными «экосистемами». Помимо этого, предпочтительно также использовать аппаратные средства, защищенные от вредоносных атак или вмешательства. Такое прозрачное туннельное соединение, используемое для реализации вариантов осуществления настоящего изобретения, способно предотвратить вмешательство хакеров, вредоносного программного обеспечения, антивирусного программного обеспечения, программного обеспечения межсетевых экранов и других устройств и/или программного обеспечения, которые контролируют и анализируют трафик данных, в данные, передаваемые по туннельному соединению.

Шаг 3 (S3): в зависимости от соединения для приема или для передачи, используемого для туннеля связи, заголовок метода GET или метода POST по-прежнему передается и принимается. Заголовок содержит необходимую информацию для данного сеанса связи, обеспечиваемого туннелем связи. Кроме того, в заголовке предпочтительно используется принятая форма структуры данных, несмотря на то, что заголовок включает следующие параметры:

(i) вид ID потока информации для сцепленных/связанных соединений;

(ii) кодирование передачи в форме порций или составного формата.

Информация, включенная в заголовок, гарантирует, что передача и прием данных происходит в виде отдельных блоков данных. Предпочтительно, максимальный размер сегмента данных (MSS) оптимизируется согласно производительности сети, поддерживающей туннель связи, с учетом количества байтов, используемых для заголовка порций или составных блоков, так что байты не теряются при передаче и приеме данных, в результате чего обеспечивается надежный и безопасный обмен данными.

Такая оптимизация сети осуществляется, например, путем запроса значения максимального блока передаваемых данных (MTU, Maximum Transfer Unit) от сетей, связывающих подключенные клиентские устройства с сервером. Таким образом, возможно определить самый слабый канал связи в сети связи и затем установить размер MSS для передачи данных в клиентское устройство, связанное с самым слабым каналом, в соответствии со скоростью самого слабого канала. Опционально, сервер может сообщать размер MSS другим клиентским устройствам системы. Такая оптимизация сети предпочтительно осуществляется посредством способа, включающего следующие

шаги:

Шаг A: система определяет самый слабый канал передачи данных, связывающий сервер с клиентскими устройствами, например, размер MTU для конкретного канала данных составляет 1500 байтов. Если из значения MTU вычесть число байтов заголовка TCP, а именно 40 байтов, останется доступными 1460 байтов. Эти 1460 байтов соответствуют размеру MSS.

Шаг B: система определяет размер MSS для данного сеанса путем использования MSS самого слабого обнаруженного канала.

Шаг C: алгоритм Нейгла, используемый в системе, отключается, чтобы предотвратить управление перегрузкой в системе, а именно посредством установки для сокета системы параметра TCP_NODELAY, который отключает алгоритм Нейгла. Отключение алгоритма Нейгла требуется, поскольку алгоритм Нейгла ждет, пока определенное число байтов данных добавится в очередь передачи, прежде чем будет отправлен соответствующий пакет данных. Когда алгоритм Нейгла отключен, система способна передавать пакеты данных с размером, определяемым исключительно системой, как указано выше.

Шаг 4 (S4): как только передан заголовок запроса HTTP и успешно получен соответствующий ответ от сервера, начинается передача и прием данных в дуплексном режиме. Таким образом, успешно создается два соединения с сервером, а именно соединение для передачи и соединение для приема; эти соединения остаются в открытом состоянии, пока не будут приняты пустая порция данных или пустой составной блок данных.

Далее два варианта осуществления изобретения поясняются с помощью кода HTTP.

Пример 1: представлен код HTTP, который при исполнении создает простое туннельное соединение для приема данных между клиентом и сервером, при этом одноранговый узел с IP-адресом 192.168.0.101 подключается к хосту с IP-адресом 192.168.0.100. В коде HTTP используются методы GET и CONNECT совместно с кодированием порций данных при передаче (chunked transfer-coding).

```
<connect>
<send> CONNECT 192.168.0.100:80 HTTP/1.0 \r\n
<send> Mozilla/5.0 (Windows NT 5.0) Gurulogic \r\n
<send> \r\n
<send> GET
/readstream?streamid=12345&param1=value1&param2=value2 HTTP/1.1 \r\n
<send> Host: 192.168.0.100 \r\n
<send> Transfer-Coding: chunked \r\n
<send> User-Agent: Mozilla/5.0 (Windows NT 5.0) Gurulogic \r\n
<send> \r\n
<recv> HTTP/1.1 200 OK \r\n
<recv> 5AD\r\n
<recv> 1453 bytes of data… \r\n
<recv> 5AD\r\n
<recv> 1453 bytes of data… \r\n
…
<recv> 5AD\r\n
<recv> 1453 bytes of data… \r\n
<recv> 0 \r\n
<disconnect from 192.168.0.100>
```

Пример 2: представлен код HTTP, который при исполнении создает простое туннельное соединение для передачи данных между клиентом и сервером, при этом одноранговый узел с IP-адресом 192.168.0.101 подключается к хосту, который имеет соответствующий IP-адрес 192.168.0.100. В коде HTTP используются методы POST и CONNECT совместно с кодированием порций данных при передаче (chunked transfer-coding).

<connect to 192.168.0.100>
<send> CONNECT 192.168.0.100:80 HTTP/1.0 \r\n
<send> Mozilla/5.0 (Windows NT 5.0) Gurulogic \r\n
<send> \r\n
<send> POST /writestream?streamid=12345&param1=value1&param2=value2 HTTP/1.1 \r\n
<send> Host: 192.168.0.100 \r\n
<send> Transfer-Coding: chunked \r\n
<send> User-Agent: Mozilla/5.0 (Windows NT 5.0) Gurulogic \r\n
<send> \r\n
<send> 5AD\r\n
<send> 1453 bytes of data… \r\n
<send> 5AD\r\n
<send> 1453 bytes of data… \r\n
…
<send> 5AD\r\n
<send> 1453 bytes of data… \r\n
<send> 0 \r\n
<recv> HTTP/1.1 200 OK \r\n

В примерах 1 и 2 предполагается, что размер MSS равен 1460 байтов, поэтому фактически размер данных для оптимизированной порции данных составляет 1453 байта. Размер оптимизированной порции данных в системе вычисляется по формуле (1).

$$\text{MSS} = (\text{начало заголовка порции}) - (\text{конец заголовка порции}) \quad (1)$$

Начало заголовка порции состоит из длины фактических данных порции, например в шестнадцатеричном формате, и конца из одного или более символов строки, которые обычно представляют собой возврат каретки (CR) и перевод строки (LF). Конец порции является аналогичным концу символов строки, которые заканчивают эту порцию.

Как показано на фиг. 2, шаг 3 (S3), а именно создание туннеля соединения посредством метода CONNECT, может быть опционально пропущен, как показано на фиг. 3. Туннель соединения не создается, если нет необходимости в туннеле. Таким образом, когда связь не используется, применяются только шаги 1, 2 и 4. Кроме того, в отношении фиг. 2 следует отметить, что туннель соединения может быть создан только для соединения GET или соединения POST, а именно создается асимметричная туннельная конфигурация связи между множеством узлов; опционально, туннель связи используется только для соединений GET или POST.

Пример 3: оптимизация MSS зависит исключительно от конкретной полезной нагрузки, переносимой конкретной порцией данных, поскольку соответствующие HTTP-заголовки порций в этот момент уже удалены при обработке, при этом полезная нагрузка блока данных составляет 100%. Такая оптимизация MSS главным образом базируется на следующем принципе. Блок MTU представляет собой отдельный передаваемый пакет и соответственно самый большой блок данных протокола, который

данный уровень может передать дальше, например 1500 байтов, при этом MSS имеет размер, который равен размеру MTU минус заголовки протокола. В вариантах осуществления данной технологии в соответствии с настоящим изобретением в MSS передается точное количество данных в байтах, которое способен пересылать самый слабый канал в данной сети. Поэтому при использовании технологии, раскрываемой в данной заявке, разбиение данных на меньшие пакеты не производится, в результате увеличивается скорость и надежность передачи данных, что, в свою очередь, приводит к меньшему количеству конфликтов и потерь пакетов, например в сети WiFi.

Ниже приведен пример оптимизации MSS.

| Клиент 1 | Операторы между клиентом 1 и клиентом 2 | Клиент 2 |
|---|---|---|
| (MTU сети 1500 байтов) | (MTU самой слабой сети 600 байтов) | (MTU сети 1300 байтов) |

Начало создания соединения:

Для тестирования сети посылаются пакеты ICMP, при этом обнаруживают, что связь между клиентом 1 и клиентом 2 прекращается, если MTU>600. Поэтому размер MTU устанавливается равным 600 байтов, что означает, что MSS равен 560 байтов, поскольку 40 байтов отпускаются на заголовок TCP, то есть учитываются. Следует отметить, что заголовки в протоколе UDP меньше, поэтому полезная нагрузка будет больше при использовании протокола UDP

Клиент 1 посылает клиенту 2 пакет размером 3000 байтов, который разделяется на 6 частей. Такое разделение является простым и предпочтительно производится по следующей формуле: полная величина в байтах делится на самый меньший размер MTU в сети минус начальный и конечный заголовки порций, а именно 3000/(560-(5+2))=5,42 пакета, при этом результат округляется до ближайшего целого числа пакетов, если нет других данных в очереди для передачи.

Пакет 1: передается 560 байтов, из которых полезная нагрузка составляет 553 байта.

Пакет 2: передается 560 байтов, из которых полезная нагрузка составляет 553 байта.

Пакет 3: передается 560 байтов, из которых полезная нагрузка составляет 553 байта.

Пакет 4: передается 560 байтов, из которых полезная нагрузка составляет 553 байта.

Пакет 5: передается 560 байтов, из которых полезная нагрузка составляет 553 байта.

Пакет 6: передается 560 байтов, из которых полезная нагрузка составляет 235 байтов.

Если бы пакет был передан непосредственно без разделения, то есть как один пакет в 3000 байтов, то он был бы разделен, то есть фрагментирован, устройствами операторов в сети, что занимает время и потенциально может вызвать проблемы, при этом, возможно, потребуется повторная передача потерянных пакетов, из-за чего передатчик должен ждать, прежде чем передавать новые пакеты по причине задержки, вызванной нестабильной сетью получателя.

Возможны модификации вариантов осуществления изобретения, рассмотренных в данной заявке, в пределах сущности изобретения, определяемой приложенной формулой изобретения.

Такие выражения как «включает», «содержит», «объединяет», «состоит», «имеет» и «является», используемые для описания изобретения, не исключают объекты, компоненты или элементы, явно не упомянутые, хотя и присутствующие. Использование единственного числа не исключает множественного. Числа, заключенные в круглые

скобки в прилагаемой формуле изобретения, указаны для обеспечения понимания формулы изобретения и не ограничивают заявленное изобретение.

## (57) Формула изобретения

1. Система (5) связи, выполненная с возможностью поддержки связи в соответствии с протоколом HTTP и с возможностью установления двухстороннего канала (40) связи в реальном времени между двумя узлами (10A, 10B) системы (5) посредством использования комбинации методов GET и POST, связанных с протоколом HTTP, причем обмен данными по каналу (40) связи осуществляется в виде порций и/или последовательности составных блоков данных,

отличающаяся тем, что максимальный размер сегмента (MSS) для порций данных и/или составных блоков данных, передаваемых по каналу (40) связи, устанавливается равным максимальному размеру сегмента для самого слабого канала передачи данных в сети связи, поддерживающей канал (40) связи, при этом обмен данными по каналу (40) связи осуществляется с использованием кодирования передачи в виде порций данных и/или последовательности составных блоков данных.

2. Система (5) связи по п. 1, отличающаяся тем, что двухсторонний канал (40) связи туннелируется в соответствии с протоколами TCP/IP и/или UDP путем использования метода CONNECT, связанного с протоколом HTTP.

3. Система (5) связи по п. 1, отличающаяся тем, что канал (40) связи включает соединение для приема и соединение для передачи для обеспечения двухсторонней связи, причем эти соединения остаются открытыми до тех пор, пока не будут получены пустая порция данных и/или пустой составной блок данных.

4. Система (5) связи по п. 1, отличающаяся тем, что канал (40) связи выполнен с возможностью использования шифрования передаваемых по нему данных.

5. Система (5) связи по п. 1, отличающаяся тем, что канал (40) связи выполнен с возможностью обеспечения передачи по меньшей мере одного из следующего: графических данных, данных изображения, видеоданных, аудиоданных, текстовых данных, неструктурированных данных.

6. Способ установления канала (40) связи посредством системы (5) связи, выполненной с возможностью поддержки связи в соответствии с протоколом HTTP, отличающийся тем, что он включает:

(a) использование системы (5) связи для установления двухстороннего канала (40) связи в реальном времени между двумя узлами (10A, 10B) системы (5) посредством использования комбинации методов GET и POST, связанных с протоколом HTTP;

(b) обмен данными по каналу (40) связи в форме порций и/или последовательности составных блоков данных и

(c) установку максимального размера сегмента (MSS) для порций данных и/или составных блоков данных, передаваемых по каналу (40) связи, равным максимальному размеру сегмента (MSS) для самого слабого канала передачи данных в сети связи, поддерживающей канал (40) связи,

при этом обмен данными по каналу (40) связи осуществляют с использованием кодирования передачи в виде порций данных и/или последовательности составных блоков данных.

7. Способ по п. 6, отличающийся тем, что он включает туннелирование двухстороннего канала (40) связи в соответствии с протоколами TCP/IP и/или UDP путем использования метода CONNECT, связанного с протоколом HTTP.

8. Способ по п. 6, отличающийся тем, что канал (40) связи включает соединение для

приема и соединение для передачи для обеспечения двухсторонней связи, причем эти соединения остаются открытыми до тех пор, пока не будут получены пустая порция и/или пустой составной блок данных.

9. Способ по п. 6, отличающийся тем, что канал (40) связи выполнен с возможностью использования шифрования передаваемых по нему данных.

10. Способ по п. 6, отличающийся тем, что канал (40) связи выполнен с возможностью обеспечения передачи по меньшей мере одного из следующего: графических данных, данных изображения, видеоданных, аудиоданных, текстовых данных, неструктурированных данных.

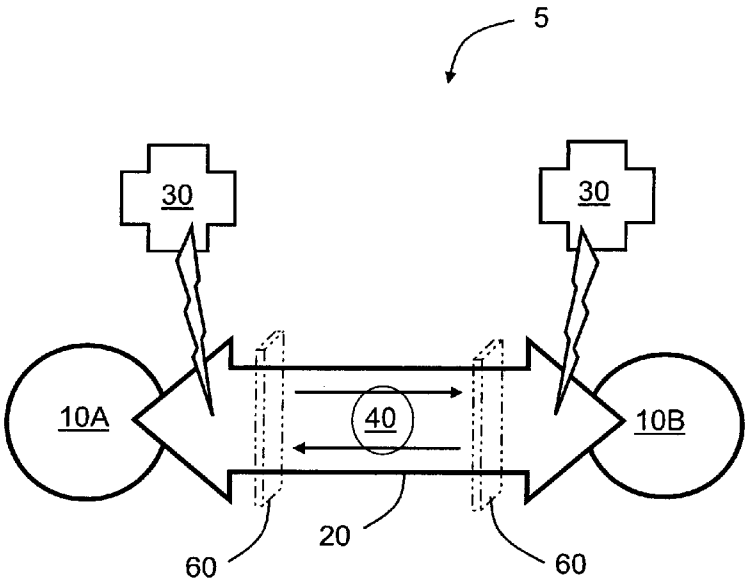
11. Машиночитаемый носитель информации, содержащий программный продукт и отличающийся тем, что программный продукт исполняется посредством компьютерного аппаратного обеспечения для осуществления способа по п. 6.

12. Машиночитаемый носитель информации по п. 11, отличающийся тем, что указанный программный продукт выполнен в соответствии с протоколом HTTP и исполняется на сервере сети связи, работающей по протоколу HTTP.

1

Система двухсторонней связи в реальном времени
с использованием протокола HTTP

1/3




Фиг.1

2


S1:
Задание ID, сочетание методов GET и POST
S2:
Создание двух TCP/IP-соединений посредством методов GET и POST
S3:
Создание туннеля связи посредством метода CONNECT
S4:
Начало приема и передачи данных в дуплексном режиме

Фиг.2

3


S1:
Задание ID, сочетание методов GET и POST
S2:
Создание двух TCP/IP-соединений посредством методов GET и POST
S4:
Начало приема и передачи данных в дуплексном режиме

Фиг.3