Network Working Group
Request for Comments: 5357
Category: Standards Track

K. Hedayat
Brix Networks
R. Krzanowski
Verizon
A. Morton
AT&T Labs
K. Yum
Juniper Networks
J. Babiarz
Nortel Networks

October 2008

A Two-Way Active Measurement Protocol (TWAMP)

Протокол активных двухсторонних измерений (TWAMP)

Статус документа

Это документ содержит проект стандарта протокола Internet для сообщества Internet и служит приглашением к дискуссии в целях развития и совершенствования протокола. Текущее состояние стандартизации и статус протокола можно узнать из текущей версии документа Internet Official Protocol Standards (STD 1). Допускается свободное распространение документа.

Аннотация

Протокол односторонних активных измерений (One-way Active Measurement Protocol или OWAMP), описанный в RFC 4656, обеспечивает основу для измерения параметров пути между сетевыми устройствами в одном направлении. OWAMP можно использовать для измерений в обоих, запуская встречные односторонние измерения между двумя элементами сети. Однако протокол не поддерживает вруговых (round-trip) или двухсторонних измерений. Этот документ задает протокол двухсторонних активных измерений (Two-Way Active Measurement Protocol или TWAMP) на основе OWAMP, который добавляет возможности круговых или двухсторонних измерений. Архитектура измерений TWAMP обычно включает два хоста с конкретными ролями и это обеспечивает некоторое упрощение протокола, делающее его привлекательным во многих ситуациях.

Оглавление

1. Введение

В IETF[1] завершена разработка предлагаемого стандарта (Proposed Standard) для метрики задержек кругового обхода [RFC2681]. Также завершен протокол для управления и сбора односторонних измерений OWAMP [RFC4656]. Однако OWAMP не может выполнять круговых или двухсторонних измерений.

Двухсторонние измерения часто возникают в сетях IP в основном по причине того, что синхронизация между локальными и удаленными часами не требуется для задержки при круговом обходе (round-trip), а поддержка измерений на удаленной стороне может ограничиваться простой эхо-функцией (отражение). Однако наиболее распространенным средством круговых измерений остается ICMP Echo Request/Reply (применяется в ping) и проблемы этого метода рассмотрены в параграфе 2.6 [RFC2681]. Этот докумен задает протокол активных двухсторонних измерений TWAMP. Этот протокол использует методологию и архитектуру OWAMP [RFC4656] для определения открытого протокола измерений двухсторонних или круговых параметров (далее в документе термин двухсторонний указывает также и круговой) в дополнение к односторонним измерениям OWAMP. В TWAMP используются временные метки, применяемые на отражающем хосте (рефлектор) для повышения точности (позволяет учесть задержки при обработке). Архитектура измерений TWAMP обычно включает лишь два хоста с конкретными ролями и это позволяет упростить протокол, делая его в некоторых случаях привлекательным дополнением к OWAMP.

Ключевые слова **необходимо** (MUST), **недопустимо** (MUST NOT), **требуется** (REQUIRED), **нужно** (SHALL), **не нужно** (SHALL NOT), **следует** (SHOULD), **не следует** (SHOULD NOT), **рекомендуется** (RECOMMENDED), **возможно** (MAY), **необязательно** (OPTIONAL) в данном документе интерпретируются в соответствии с RFC 2119 [RFC2119].

1.1. Связь между протоколами тестирования и управления

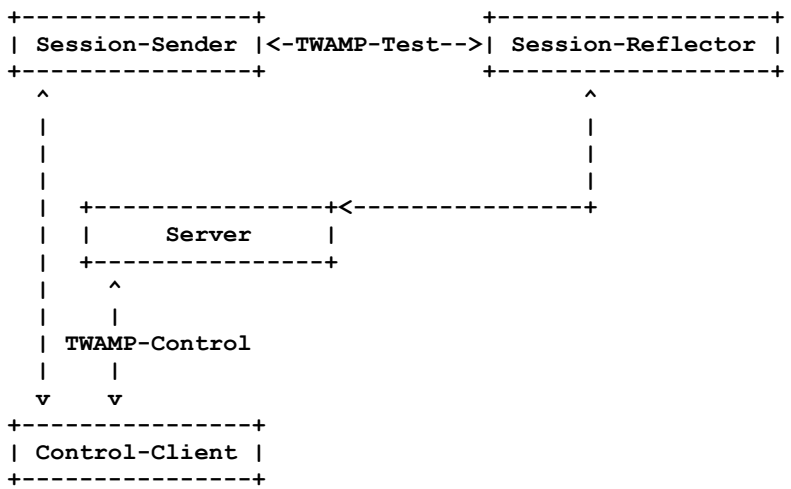
Подобно OWAMP [RFC4656], протокол TWAMP включает два связанных между собой протокола - TWAMP-Control и TWAMP-Test. Связь между этими протоколами определена в параграфе 1.1 спецификации OWAMP [RFC4656]. TWAMP-Control служит для организации, запуска и остановки тестовых сеансов, а TWAMP-Test - для обмена тестовыми пакетами между элементами TWAMP.

1.2. Логическая модель

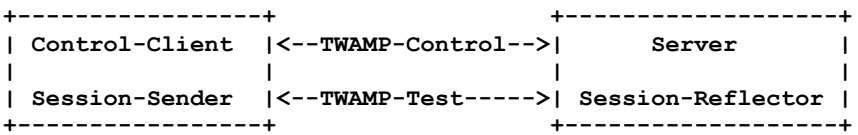
Роли и определения логических элементов заданы в параграфе 1.2 спецификации OWAMP [RFC4656], а ниже приведен ряд исключений.

- Session-Receiver называется Session-Reflector в архитектуре TWAMP. Рефлектор может создавать и передавать тестовые пакеты при получении другого тестового пакета. В отличие от Session-Receiver, рефлектор не собирает информацию из пакетов.
- Сервер является конечной системой, которая управляет одной или множеством сессий TWAMP и позволяет настраивать состояние конечных точек на уровне сессии. Однако сервер, связанный с Session-Reflector, не сможет возвращать результаты сеанса тестирования и в этом состоит отличие от OWAMP.
- Элемента Fetch-Client нет в архитектуре TWAMP, поскольку Session-Reflector не собирает информации из пакетов.

Пример возможных связей между элементами в разных ролях показан на рисунке. В этом примере роли распределены между разными хостами. Каналы без меток не задаются этой спецификацией и могут использовать фирменные (proprietary) протоколы.



Как и в OWAMP [RFC4656], разные логические роли могут быть связаны с одним хостом. В примере на приведенном выше рисунке можно обойтись двумя хостами, на одном из которых будут работать Control-Client и Session-Sender, на другом - Server и Session-Reflector, как показано ниже.



1.3. Произношение

Акроним OWAMP обычно произносится в два слога как Oh-wamp.

Акроним TWAMP обычно произносится в два слога как Tee-wamp.

[1]Internet Engineering Task Force.

2. Обзор протокола

TWAMP является открытым протоколом для двухсторонних измерений. Он основан на OWAMP [RFC4656] придерживается общих принципов архитектуры и устройства. Протоколы TWAMP-Control и TWAMP-Test выполняют свои задачи, как описано ниже.

- Control-Client инициирует соединение TCP через общеизвестный порт TWAMP и сервер (его роль установлена) отвечает сообщением Greeting, включающим желаемый режим защиты и контроля целостности.
- Control-Client отвечает выбранным режимом связи и информацией для поддержки защиты целостности и шифрования, если этого требует режим. Server отвечает восприятием режима и сообщает свое время старта. На этом организация управляющего соединения завершается.
- Control-Client запрашивает (и описывает) сессию тестирования уникальным сообщением TWAMP-Control. Сервер отвечает восприятием и поддерживающей информацией. Можно запросить дополнительные сессии, передав другие такие же сообщения.
- Control-Client инициирует все запросы тестирования в сообщении Start-Sessions, Server подтверждает их.
- Session-Sender и Session-Reflector обмениваются тестовыми пакетами в соответствии с протоколом TWAMP-Test в каждой сессии.
- При необходимости Control-Client передает сообщение для остановки всех тестовых сессий.

В протоколе TWAMP имеется два расширения.

1) Поле Modes служит для организации коммуникационных опций при создании соединения TWAMP-Control.
2) TWAMP-Control Command Number является другим механизмом расширения, позволяющим в будущем определить дополнительные команды.

TWAMP-Control позволяет распределять возможности между Control-Client и Server.

Многооктетные значения, определенные в этом документе, представляются целыми числами без знака с сетевым порядком байтов, если явно не указано иное.

Биты, помеченные символами MBZ (должен быть 0) в этом документе, отправитель **должны** устанавливать в 0, а получателия **должны** игнорировать.

3. TWAMP-Control

TWAMP-Control является адаптацией протокола OWAMP-Control для двухсторонних измерений. Все сообщения TWAMP-Control похожи по формату и следуют рекомендациям, походим на заданные в разделе 3 спецификации OWAMP [RFC4656], за исключением отмеченных в последующих параграфах отличий. Одним из таких отличий является отсутствие в TWAMP команды Fetch-Session.

3.1. Организация соединения

Организмция соединения TWAMP следует процедуре, определенной в параграфе 3.1 спецификации OWAMP [RFC4656]. Поле Modes является признанным механизмом расширения в TWAMP и текущие значения режимов идентичны принятым в OWAMP. Единственным исключением является номер общеизвестного порта для TWAMP-Control. Клиент открывает соединение TCP с сервером через порт 862. Хост, инициирующий соединение TCP, принимает роли Control-Client и (при работе на двух хостах) Session-Sender. Хост, подтверждающий соединение TCP принимает роли Server и (при работе на двух хостах) Session-Reflector.

Control-Client **может** установить желаемый код Diffserv (DSCP) в поле заголовка IP для **всех** пакетов в текущем управляющем соединении. Серверу **следует** использовать значение DSCP из пакета TCP SYN от Control-Client во **всех** последующих пакетах данного соединения (избегая двусмысленности в случае перемаркировки).

Существует возможность отказа Control-Client после организации соединения TWAMP-Control или на пути между Control-Client и Server может возникнуть сбой при работе соединения. Сервер **может** прекратить любое организованное соединение при отсутствии связанных с ним принятых пакетов в течение SERVWAIT секунд. Серверу **нужно** приостановить мониторинг управляющего соединения после получения команды Start-Sessions и **нужно** возобновить его после получения команды Stop-Sessions (если поддерживается опция SERVWAIT). Отметим, что тайм-аут REFWAIT (описан ниже) охватывает отказы в тестовых сессиях и при завершении REFWAIT во **всех** сеансах тестирования, инициированных соединением TWAMP-Control мониторинг SERVWAIT **нужно** восстановить (как по команде Stop-Sessions). Реализации, поддерживающей тайм-аут SERVWAIT, **следует** также реализовать тайм-аут REFWAIT. В качестве принятого по умолчанию значения SERVWAIT **нужно** установить 900 секунд и это время **может** быть настраиваемым. Этот тайм-аут позволяет серверу освободить ресурсы в случае отказа.

Клиент и сервер используют одно отображение KeyID на общие секреты. Сервер, готовящийся к сессиям с несколькими клиентами, использует KeyID для выбора подходящего секрета, а клиент обычно имеет разные секретные ключи для разных серверов. Общий секрет является парольной фразой (passphrase). Для максимальной совместимости парольных фраз символы в них **должны** кодироваться в соответствии с Прилоежением B из [RFC5198] (ASCII Network Virtual Terminal Definition). В них **должны** отсутствовать символы новой строки (любая комбинация CR и/или LF) и **следует** избегать символов управления.

3.2. Защита целостности

Защита целостности TWAMP использует процедуру, определенную в параграфе 3.2 спецификации OWAMP [RFC4656]. Как и в OWAMP, кахдый код HMAC (Hashed Message Authentication Code) учитывает все, передаваемое в данном направлении между предыдущим HMAC (не включая его) и началом нового HMAC. Таким образом, после организации шифрования каждый бит соединения TWAMP-Control аутентифицируется HMAC ровно 1 раз.

Отметим, что сообщение Server-Start (сервер передает его в начальном обмене управляющего соединения) не завершается полем HMAC. Поэтому HMAC в первом сообщении Accept-Session учитывает также сообщение Server-Start и включает поле Start-Time в расчет HMAC.

В режиме с аутентификацией и шифрованием HMAC в пакетах TWAMP-Control шифруется.

3.3. Значения поля Accept

Значения Accept в TWAMP используются в соответствии с параграфом 3.3 спецификации OWAMP [RFC4656].

3.4. Команды TWAMP-Control

Команды TWAMP-Control соответствуют правилам параграфа 3.4 в спецификации OWAMP [RFC4656]. Для Control-Client доступны команды Request-TW-Session, Start-Sessions, Stop-Sessions. Сервер может передавать конкретные сообщения в ответ на полученные команды (описаны ниже).

Команда OWAMP Request-Session заменена TWAMP Request-TW-Session, а команды Fetch-Session нет в TWAMP.

3.5. Организация тестовых сессий

Создание тестовой сессии следует процедуре параграфа 3.5 в спецификации OWAMP [RFC4656]. Команда Request-TW-Session основана на команде OWAMP Request-Session и использует формат, описанный в параграфе 3.5 спецификации OWAMP, но без полей Schedule Slot Descriptions и использует лишь один код HMAC. Описание формата Request-TW-Session приведено ниже.

В TWAMP первый октет называется номером команды и Command Number является признанным механизмом расширения. Читателям рекомендуется обратиться к реестру TWAMP-Control Command Number при возникновении потребности в дополнительных значениях.

Command Number = 5 указывает команду Request-TW-Session и сервер **должен** интерпретировать ее как запрос двухсторонней тестовой сессии с использованием протокола TWAMP-Test.

Если сервер TWAMP получает неожиданное значение Command Number, он **должен** ответить сообщением Accept-Session с полем Accept = 3 (некоторые аспекты запроса не поддерживаются). Неожиданными являются запрещенные (Forbidden) номера команд и возможно некоторые из числа резервных (Reserved).

В OWAMP для поля Conf-Sender устанавливается значение 1, когда сообщение Request-Session описывает задачу, где сервер будет настраивать отправителя односторонних тестовых пакетов. Аналогичным образом устанавливается Conf-Receiver = 1, когда сообщение описывает конфигурацию Session-Receiver. В TWAMP обе стороны передают и принимают пакеты, причем Session-Sender сначала передает, затем принимает тестовые пакеты, а Session-Reflector сначала принимает, потом передает.

В полях Conf-Sender и Conf-Receiver **должно** быть указано значение 0, поскольку Session-Reflector будет принимать и передавать пакеты, а роли устанавливаются в соответствии с тем, какой из хостов инициирует соединение TCP для управления. Сервер **должен** интерпретировать ненулевое значение как команду с некорректным форматом и **должен** отвечать сообщением Accept-Session с полем Accept = 3 (некоторые аспекты запроса не поддерживаются).

Session-Reflector в TWAMP не обрабатывает входящие тестовые пакеты для определения параметров производительности, поэтому ему не нужно знать номера пакетов или тактирование их передачи. Поэтому поля Number of Scheduled Slots и Number of Packets **должны** устанавливаться в 0.

Sender Port указывает порт UDP, из которого будет переданы пакеты TWAMP-Test и в который Session-Reflector будет отправлять пакеты TWAMP-Test (Session-Sender использует один порт UDP для передачи и приема). Receiver Port указывает желаемый порт UDP, в который Session-Sender будет направлять пакеты TWAMP-Test (порт, запрошенный у Session-Reflector для приема пакетов). Receiver Port также является портом UDP, из которого Session-Reflector будет передавать пакеты TWAMP-Test (Session-Reflector использует один порт UDP для передачи и приема).

Поля Sender Address и Receiver Address содержат адреса отправителя и получателя в конечных точках пути Internet, через который запрошена сессия TWAMP-Test. В них **можно** указать 0 и в таком случае для тестовых пакетов **должны** использоваться адреса IP из обмена сообщениями TWAMP-Control между Control-Client и Server.

Идентификатор сессии (Session Identifier или SID) соответствует определению в OWAMP [RFC4656]. Поскольку SID всегда генерируется принимающей стороной, сервер определяет SID и поле SID в сообщении Request-TW-Session **должно** иметь значение 0.

Поле Start Time соответствует определению в OWAMP [RFC4656].

Поле Timeout интерпретируется не так как в OWAMP [RFC4656]. В TWAMP это поле указывает интервал, в течение которого Session-Reflector должен ждать перед отправкой сообщения Stop-Sessions. Когда тестовые пакеты продолжают поступать, Session-Reflector **должен** отражать их, если пакеты поступают в интервале Timeout с момента получения Stop-Sessions. Рефлектору **недопустимо** отражать пакеты по достижении тайм-аута.

Дескриптор Type-P соответствует определению в OWAMP [RFC4656]. Единственным назначением этого поля является установка кода дифференцированного обслуживания (DSCP) [RFC2474]. Это же значение DSCP **должно** указываться в отраженных пакетах от Session-Reflector.

Поскольку в протоколе не применяются Schedule Slot, сообщение Request-TW-Session завершается полями MBZ (0) и HMAC. Это завершает логическое сообщение, называемое командой Request-TW-Session.

Рефлектор **должен** отвечать на каждую команду Request-TW-Session сообщением Accept-Session, как указано в спецификации OWAMP [RFC4656]. При Accept = 0 поле Port подтверждает (повторяет) номер порта, в который Session-Sender передает пакеты TWAMP-Test для Session-Reflector. Иными словами, поле Port указывает номер порта, на котором ожидается прием рефлектором пакетов от Session-Sender.

Когда запрошенный Receiver Port не доступен (например, уже занят), сервер на стороне Session-Reflector **может** предложить другой доступный порт для этой сессии в поле Port. Session-Sender воспринимает этот порт или создает

новое сообщение Session-Request с подходящими параметрами. В противном случае сервер использует поле Accept для передачи иного способа отклонения сессии или отказа клиенту управления (Control-Client) и предлагать другой порт **недопустимо**[1]. При этом в поле Port **должно** устанавливаться значение 0.

3.6. Планирование передачи

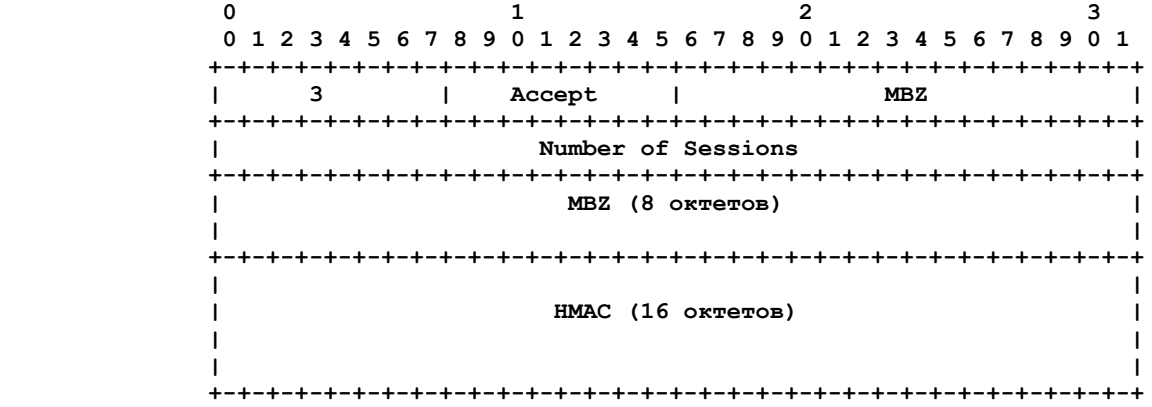
Планирование передачи тестовых пакетов, заданное в параграфе 3.6 спецификации OWAMP [RFC4656], не применяется в TWAMP. Control-Client и Session-Sender **могут** самостоятельно планировать передачу. Рефлектору **следует** возвращать каждый тестовый пакет его Session-Sender как можно скорее.

3.7. Запуск сеанса тестирования

Процедуры и рекомендации для запуска тестовых сессий соответствуют параграфу 3.7 в OWAMP [RFC4656].

3.8. Команда Stop-Sessions

Процедуры и рекомендации для остановки сеансов тестирования отличаются от заданных в параграфе 3.8 спецификации OWAMP [RFC4656]. Команду Stop-Sessions может подавать лишь Control-Client. В сообщение **недопустимо** включать какие-либо описания сессии или диапазоны пропуска. Сообщение завершается одним блоком HMAC (завершение команды Stop-Sessions). Команда TWAMP Stop-Sessions не включает SID и применяется ко всем сессиям запрошенным и созданным командами Start-Sessions. Формат Stop-Sessions показан на рисунке.



Command Number = 3 в первом октете указывает команду Stop-Sessions.

Ненулевое значение Accept указывает причину отказа, а 0 говорит о нормальном (возможно преждевременном) выполнении. Список возможных значений Accept приведен в параграфе 3.3 [RFC4656]. Если поле Accept отлично от 0, все сессии TWAMP-Test, созданные этим сеансом TWAMP-Control, **следует** считать недействительными. Если сообщение Accept-Session не было передано (по любой причине, включая сбой соединения TCP для TWAMP-Control), результаты всех сессий TWAMP-Test, созданных этой сессией TWAMP-Control **можно** считать недействительными.

Number of Sessions указывает число сессий, которые Control-Client должен остановить. Это поле **должно** указывать число сеансов передачи, запущенных Control-Client и не прерванных ранее командой Stop-Sessions (т. е. Control-Client **должен** учитывать каждое воспринятое сообщение Request-Session). Если сообщение Stop-Sessions не указывает точного числа действующих сессий, оно будет считаться недействительным, а соединение TWAMP-Control **следует** закрывать, считая все полученные результаты недействительными.

После получения команды TWAMP-Control Stop-Sessions рефлектор **должен** отбрасывать все пакеты TWAMP-Test, принятые по истечении интервала Timeout (из команды Request-TW-Session) с момента получения команды.

3.9. Команда Fetch-Session

Одной из целей TWAMP является двухстороннее измерение. Методы такого измерения не требуют сборки рефлектором данных на уровне пакетов (таких как порядковые номера, временные мети, TTL), поскольку они передаются в «отраженных» тестовых пакетах. Поэтому протокол не требует от сервера извлечения данных из пакетов и команда OWAMP Fetch-Session не применяется в TWAMP.

4. Протокол TWAMP-Test

Протокол TWAMP-Test похож на OWAMP-test [RFC4656] за исключением того, что рефлектор передает тестовые пакеты отправителю в ответ на каждый пакет от Session-Sender. TWAMP определяет 2 формата пакетов, один из которых передает Session-Sender, другой - Session-Reflector. Как и в OWAMP-test [RFC4656], применяется 3 режима: без аутентификации, с аутентификацией, с шифрованием.

4.1. Поведение отправителя

Поведение отправителя определяется конфигурацией Session-Sender и не задается этим стандартом. Кроме того, рефлектору не нужно знать поведение Session-Sender так детально, как требуется в OWAMP [RFC4656]. Session-Sender собирает и записывает всю требуемую для двухсторонних измерений информацию, получая ее из пакетов от Session-Reflector. Записываемая информация зависит от реализации.

4.1.1. Тактирование пакетов

Поскольку информация о планировании пакетов не передается рефлектору, не возникает необходимости в стандартизованном тактировании пакетов. Независимо от задержек планирования, каждый переданный пакет **должен** включать в свою временную метку как можно более точное значение реального времени отправки.

[1]В исходном документе это предложение содержит ошибку. См. https://www.rfc-editor.org/errata/eid1587. *Прим. перев.*

4.1.2. Формат и содержимое пакетов

Формат и содержимое пакетов Session-Sender следуют процедурам и рекомендациям параграфа 4.1.2 с спецификации OWAMP [RFC4656] (за исключением планирования передачи).

Отметим, что тестовые пакеты от рефлектора по размеру превышают пакеты от Sender. Session-Sender **может** добавлять в конец пакета заполнение (Packet Padding) для выравнивания размера данных в пакетах IP (payload) для обоих направления (обычно это желательно). Для компенсации большего размера тестовых пакетов от рефлектора Sender добавляет по меньшей мере 27 октетов заполнения в режиме без аутентификации и 64 октета в режимах с аутентификацией и шифрованием.

4.2. Поведение рефлектора

TWAMP требует от Session-Reflector передачи пакета отправителю в ответ на каждый пакет от Session-Sender.

Ниже указаны действия Session-Reflector при получении пакета.

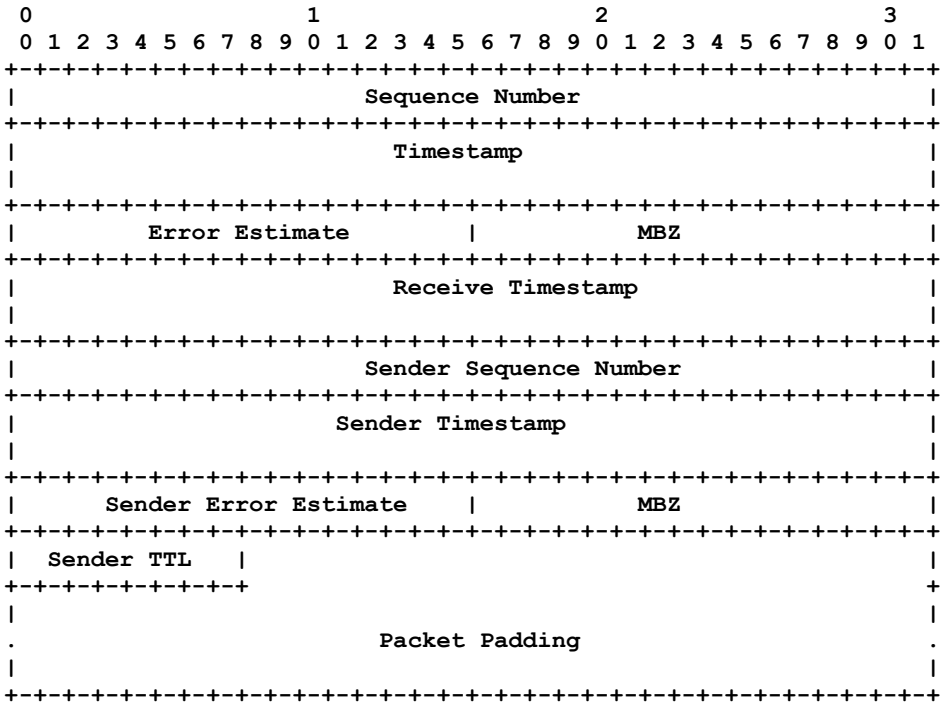
- Для пакета создается временная метка. Каждый принятый пакет должен иметь как можно более точное время реального прибытия в его Received Timestamp (в пакете).
- В режимах с аутентификацией и шифрованием расшифровываются соответствующие части тела пакета (первый блок из 16 октетов октетов для режима с аутентификацией и 96 для режима с шифрованием), затем проверяется целостность разделов, учитываемых в HMAC.
- Копируется порядковый номер пакета в соответствующий отраженный пакет для Session-Sender.
- Извлекается значение Sender TTL из поля TTL/Hop Limit в принятом пакете. Реализациям Session-Reflector **следует** брать TTL/Hop Limit из заголовка IP, заменяя им значение 255, установленное Session-Sender. Если реализация не извлекает реальное значение TTL (единственной разумной причиной является невозможность доступа в полю TTL в принятых пакетах), она **должна** установить Sender TTL = 255.
- В режимах с аутентификацией и шифрованием сначала **должно** рассчитываться значение HMAC, затем нужные части тела пакета шифруются.
- Тестовый пакет передается Session-Sender в ответ на каждый принятый пакет. Отклик **должен** создаваться незамедлительно. Формат и содержимое пакета заданы в параграфе 4.2.1. Перед отправкой тестового пакета рефлектор **должен** указать максимально точно время реальной отправки в поле Timestamp (в пакете). Это позволяет определить время между приемом и передачей пакета.
- Пакеты, не принятые в интервале Timeout (после команды Stop-Sessions), рефлектор **должен** игнорировать. Рефлектору **недопустимо** генерировать тестовые пакеты для игнорируемых пакетов от Session-Sender.

Существует вероятность отказа Session-Sender или сбоя на пути между Session-Sender и Session-Reflector во время сессии. Рефлектор **может** прекратить любую начатую сессию, если в течение REFWAIT секунд не было получено связанных с ней пакетов. По умолчанию для REFWAIT **нужно** устанавливать значение 900 секунд и это значение **можно** делать настраиваемым. Этот тайм-аут позволяет рефлектору освободить ресурсы при возникновении отказа.

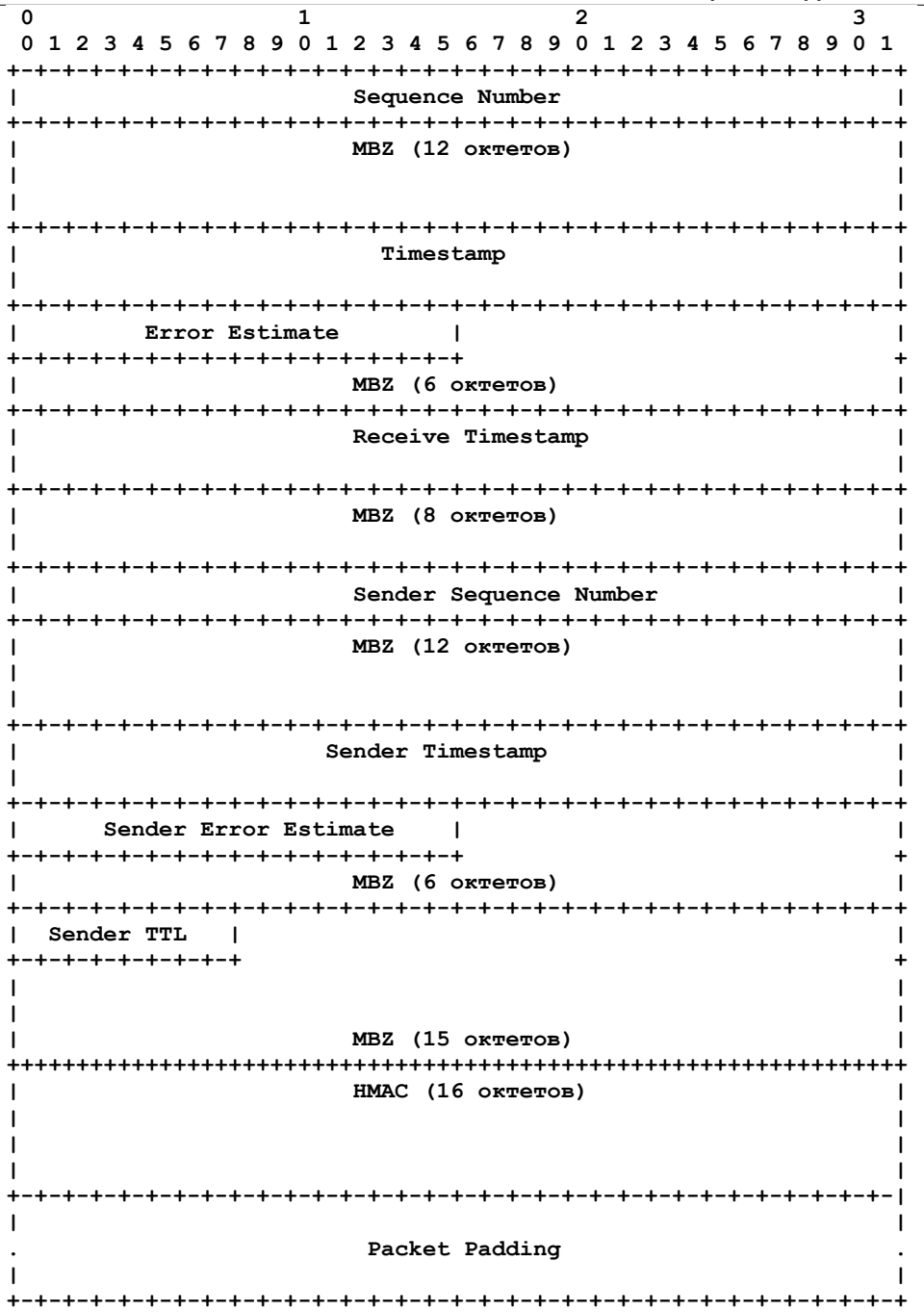
4.2.1. Формат и содержимое пакетов TWAMP-Test

Session-Reflector **должен** передавать свой пакет в ответ на каждый принятый от Session-Sender пакет. Рефлектору **следует** передавать пакеты как можно скорее. Рефлектору **следует** устанавливать в поле TTL для IPv4 (или Hop Limit для IPv6) пакета UDP значение 255.

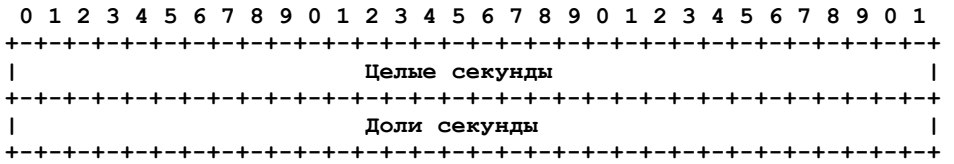
Тестовые пакеты включают информацию, требуемую для расчета двухсторонних параметров на стороне Session-Sender. Формат тестовых пакетов зависит от режима. На рисунке показан формат пакетов в режиме без аутентификации.



На следующем рисунке приведен формат пакетов в режимах с аутентификацией и шифрованием.



Отметим, что временные метки имеют такой же формат как в OWAMP [RFC4656], показанный на рисунке.



Sequence Number указывает номер тестового пакета в соответствии с порядком передачи в сессии TWAMP. Номера начинаются с 0 и значение инкрементируется при отправке каждого пакета в этой сессии[1]. Номера, создаваемые рефлектором, не зависят от номеров в прибывающих пакетах.

Поля Timestamp и Error Estimate содержат временную метку передачи и оценку ошибок на стороне Session-Reflector для отраженных пакетов. Формат полей соответствует определению OWAMP, параграф 4.1.2 в [RFC4656].

Sender Timestamp и Sender Error Estimate являются точными копиями одноименных полей из тестового пакета от Session-Sender, соответствующего данному пакету.

Sender TTL имеет значение 255 при передаче от Session-Sender. В этом поле Session-Reflector устанавливает значение TTL (Hop Count) из заголовка IP полученного пакета.

Receive Timestamp указывает время приема тестового пакета рефлектором. Разница между Timestamp и Receive Timestamp составляет время пребывания пакета в Session-Reflector. Значение Error Estimate, связанное с полем Timestamp, применяется и к Receive Timestamp.

Sender Sequence Number содержит копию поля Sequence Number из пакета, переданного Session-Sender, который вызвал генерацию и отправку рефлектором данного пакета.

[1] В оригинале это предложение содержит неточности. См. https://www.rfc-editor.org/errata/eid1590. *Прим. перев.*

Поле HMAC в пакетах TWAMP-Test учитывает те же поля, что и шифрование AES (Advanced Encryption Standard). Таким образом, в режиме с аутентификацией HMAC охватывает первый блок (16 октетов), а в режиме с шифрованием - первые 6 блоков (96 октетов). В пакетах TWAMP-Test **недопустимо** шифровать поле HMAC.

В поле заполнения (Packet Padding) пакетов TWAMP-Test **следует** помещать псевдослучайные значения (они **должны** генерироваться независимо от других псевдослучайных чисел, упоминаемых в этом документе). Однако реализация **должна** обеспечивать параметр конфигурации, опцию или иной способ выбора для поля Packet Padding заполнения нулями. Поле Packet Padding **недопустимо** учитывать в HMAC и **недопустимо** шифровать.

Минимальный размер сегмента данных в пакетах TWAMP-Test составляет 41 октет в режиме без аутентификации и 112[1] октетов в режимах с аутентификацией и шифрованием.

Отметим, что тестовые пакеты от Session-Reflector превышают по размеру пакеты от Session-Sender. Рефлектору **следует** снижать размер поля заполнения Packet Padding для выравнивания размера пакетов IP в обоих направлениях при наличии достаточного объема заполнения. Session-Reflector **может** повторно использовать содержимое Packet Padding (те же требования к заполнению) и в этом случае рефлектору **следует** просто отсечь ненужное заполнение.

В режиме без аутентификации **недопустимо** применять аутентификацию и шифрование.

Схемы пакетов TWAMP-Test в режимах с аутентификацией и шифрованием идентичны. Операция шифрования для пакетов Session-Sender следует правилам из параграфа 4.1.2 спецификации OWAMP [RFC4656].

Основное различие между режимами с аутентификацией и шифрованием заключается в охватываемых HMAC и шифрованием частях пакета. Режим с аутентификацией позволяет получать временную метку после шифрования части пакета, а в режиме с шифрованием порядковые номера и временные метки извлекаются перед шифрованием для обеспечения максимальной защиты целостности данных.

В режиме с аутентификацией шифруется лишь порядковый номер в первом блоке, а последующие временные метки и номера передаются в открытом виде. Передача временных меток без шифрования позволяет сократить время между получением рефлектором временной метки и отправкой пакета. Это может повышать точность временных меток, поскольку шифрование можно выполнить до получения метки.

В режиме с шифрованием рефлектор **должен** извлечь временные метки, рассчитать HMAC и зашифровать пакет до его отправки.

Получение ключей и методы шифрования следуют таким же процедурам OWAMP, как описано ниже. Каждая сессия TWAMP-Test имеет 2 сеансовых ключа - AES и HMAC, которые выводятся из ключей TWAMP-Control и SID.

TWAMP-Test AES Session-key получается путем шифрования ключа TWAMP-Control AES Session-key (тот же AES Session-key, что и для соответствующей сессии TWAMP-Control) с 16-октетным (SID) в качестве ключа с помощью одного блока AES-ECB. Полученный в результате TWAMP-Test AES Session-key служит для шифрования и расшифровки пакетов в конкретной сессии TWAMP-Test. Отметим, что TWAMP-Test AES Session-key, TWAMP-Control AES Session-key и SID имеют размер 16 октетов.

TWAMP-Test HMAC Session-key получается путем шифрования TWAMP-Control HMAC Session-key (тот же HMAC Session-key, что применяется в сессии TWAMP-Control) с использованием AES-CBC (Cipher Block Chaining) с 16-октетным SID в качестве ключа. Это 2-блочное шифрование CBC, всегда выполняемое с IV=0. Отметим, что TWAMP-Test HMAC Session-key и TWAMP-Control HMAC Session-key имеют размер 32 октета, а SID - 16 октетов.

В режиме с аутентификацией первый блок (16 октетов) каждого пакета TWAMP-Test шифруется в режиме AES ECB (Electronic Codebook). Этот режим не использует цепочек, а потеря, дублирование и нарушение порядка пакетов не создает проблем при расшифровке в сессии TWAMP-Test.

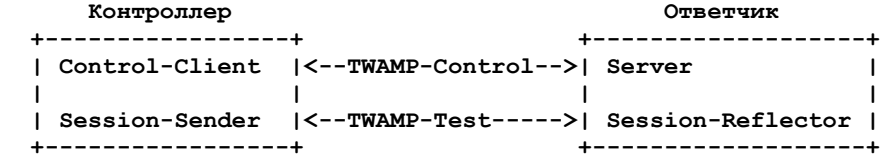
В режиме с шифрованием первые 6 блоков (96 октетов) шифруются в режиме AES-CBC. Сеансовый ключ AES получается так же как в режиме с аутентификацией. Каждый пакет TWAMP-Test шифруется как отдельный поток без включения в цепочку нескольких пакетов, что позволяет предотвратить проблемы с расшифровкой при потере, дублировании или нарушении порядка пакетов. Вектор инициализации для шифрования CBC содержит значение 0.

Следует отметить, что планирование ключей в каждой сессии TWAMP-Test **должно** выполняться один раз в течение сессии, а не для каждого пакета.

5. Рекомендации для разработчиков

В этом разделе приведены рекомендации для разработчиков TWAMP. Представленный пример не является требованием. Подобно OWAMP [RFC4656], протокол TWAMP обеспечивает достаточную гибкость для поддержки разной архитектуры и системных требований.

В примере роли Control-Client и Session-Sender реализованы на одном хосте, названном контроллером, а роли Server и Session-Reflector - на другом хосте, названном ответчиком.



Пример представляет архитектуру, полностью поддерживающую стандарт TWAMP. Контроллер организует сеанс тестирования по протоколу TWAMP-Control. После организации сессии контроллер передает ответчику тестовые пакеты, а ответчик работает в соотвесттвии с поведением Session-Reflector, описанным в параграфе 4.2.

В Приложении I с информационными целями представлен другой пример, предлагающий постепенное внедрение TWAMP путем реализации сначала протокола TWAMP-Test.

[1]В оригинале ошибочно указано 104. См. https://www.rfc-editor.org/errata/eid5045. *Прим. перев.*

6. Вопросы безопасности

По сути в TWAMP и OWAMP используется один протокол для организации управления и тестирования. Основное различие между TWAMP и OWAMP заключается в поведении Session-Reflector для TWAMP и Session-Receiver для OWAMP. Это различие не создает известных уязвимостей, которые уже не решены защитными средствами OWAMP. Все рассмотрение вопросов безопасности OWAMP [RFC4656] применимо к протоколу TWAMP.

Сообщение Server-Greeting (параграф 3.1 в [RFC4656]) включает поле Count для задания счетчика итераций, используемого в PKCS #5 при создании ключей из общих секретов. OWAMP рекомендует нижний предел в 1024 итерации, но не задает верхний предел. Поле Count создает уязвимость к DoS-атакам[1], поскольку имеет размер 32 бита. Если атакующий установит для Count максимальное значение 2^{32}-1, атакованная система «остановится» на продолжительное время для генерации ключей. Поэтому совместимым с TWAMP системам **следует** включать конфигурационное ограничение для поля Count. По умолчанию **следует** устанавливать максимальное значение 32768. Как предлагается в OWAMP, это значение **можно** будет увеличить по мере роста вычислительной мощности. Если Control-Client получит сообщение Server-Greeting с Count больше максимального значения, ему **следует** закрыть управляющее соединение.

7. Благодарности

Спасибо Nagarjuna Venna, Sharee McNab, Nick Kinraid, Stanislav Shalunov, Matt Zekauskas, Walt Steverson, Jeff Boote, Murtaza Chiba и Kevin Earnst за их комментарии, предложения, рецензии, полезное обсуждение и вычитывание документа. Lars Eggert, Sam Hartman и Tim Polk предоставилии полезные обзоры на уровне AD и авторы признательны им за вклад в документ.

8. Взаимодействие с IANA

Агентство IANA выделило номер порта TCP 861 для протокола OWAMP-Control в составе OWAMP [RFC4656].

```
...
owamp-control   861/tcp    OWAMP-Control
owamp-control   861/udp    OWAMP-Control
#                          [RFC4656]
```

В IANA также выделены порты TCP и UDP для протокола TWAMP-Control.

```
...
twamp-control   862/tcp    Two-way Active Measurement Protocol
                           (TWAMP) Control
twamp-control   862/udp    Two-way Active Measurement Protocol
                           (TWAMP) Control
#                          [RFC5357]
```

Поскольку в TWAMP добавлена дополнительная команда управления, отсутствующая в спецификации OWAMP-Control, и описано поведение при использовании этой команды, в IANA создан реестр значений TWAMP Command Number. Поле не названо явно в [RFC4656], но используется для каждой команды. Это поле признано механизмом расширения для TWAMP.

8.1. Спецификация реестра

Агентство IANA создало реестр TWAMP-Control Command Number. Команды TWAMP-Control указываются первым октетом в сообщении OWAMP-Control, как показано в параграфе 3.5 [RFC4656] и обновлено в данном документе. В результате реестр может включать 256 различных значений.

8.2. Управление реестром

Поскольку реестр может включать лишь 16 значений, а OWAMP и TWAMP являются протоколами IETF, реестр должен обновляться лишь по процедуре IETF Consensus, описанной в [RFC5226], с выпуском RFC, документирующего использованием значения и одобренного IESG. Предполагается, что новые значения будут выделяться по возрастанию из диапазона [0-255], если не будет предложено иного разумного решения.

8.3. Экспериментальные значения

[RFC3692] рекомендует выделять некоторое количество значений для экспериментов и тестирования. Авторам неясно, сколько значений можно с пользой выделить для этих целей или разумно просто предоставить для экспериментов «верхний» край диапазона. Могут быть полезны 2 значения - одно для управления, другое для тестовой сессии. С другой стороны, один номер можно расширять неограниченно, используя для этого формат остальной части сообщения, а другие номера выделять по мере осознания их полезности. Поэтому в документе для экспериментов и тестирования выделен один номер (6).

8.4. Начальное содержимое реестра

Реестр TWAMP-Control Command Number.

Значение	*Описание*	*Определение семантики*
0	Резерв	
1	Forbidden	
2	Start-Sessions	RFC 4656, параграф 3.7
3	Stop-Sessions	RFC 4656, параграф 3.8
4	Резерв	
5	Request-TW-Session	Данный документ, параграф 3.5
6	Experimentation	Не определено, см. параграф 8.3.

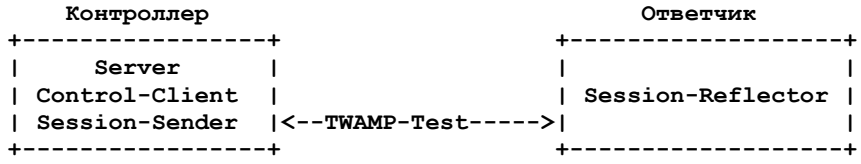
[1]Denial-of-service - отказ в обслуживании.

9. Использование других языков

Протокол не передает какой-либо информации на естественных языках, за исключением разве что KeyID в TWAMP-Control, где применяется кодировка UTF-8 [RFC3629, RFC5198].

Приложение I - TWAMP Light (информационное)

В этом примере роли Control-Client, Server и Session-Sender реализованы на одном хосте (контроллер), а роль Session-Reflector - на другом (ответчик).



Пример представляет простую архитектуру для ответчиков, где их роль заключается лишь в создании тестовых точек в сети. Контроллер организует тестовую сессию с сервером нестандартным путем. После организации сессии контроллер передает тестовые пакеты ответчику, а тот следует поведению Session-Reflector, описанному в параграфе 4.2, с указанными ниже исключениями.

В случае TWAMP Light рефлектор не обязательно знает состояние сессии. Если это состояние неизвестно, Session-Reflector **должен** копировать Sequence Number из принятого пакета в одноименное поле отраженного пакета. Контроллер получает отраженные пакеты и собирает результаты двухсторонних измерений.

Пример исключает необходимость протокола TWAMP-Control и предполагает, что Session-Reflector настраивается и сообщает свою конфигурацию серверу нестандартным путем. Session-Reflector просто отражает входящие пакеты контроллеру, копируя нужную информацию и генерируя порядковые номера и временные метки в соответствии с параграфом 4.2.1. TWAMP Light порождает некоторые дополнительные вопросы безопасности. Нестандартным путям управления ответчиком и организации тестовых сессий **следует** поддерживать указанные ниже свойства.

Протоколу управления ответчиком **следует** поддерживать режим работы с аутентификацией. Ответчику **следует** поддерживать настройку на восприятие лишь сеансов управления с аутентификацией.

Протоколу управления ответчиком **следует** поддерживать способы активации режимов с аутентификацией и шифрованием для протокола TWAMP-Test.

При работе тестовой сессии TWAMP Light в режиме с аутентификацией и шифрованием Session-Reflector **должен** поддерживать тот или иной механизм генерации ключей (поскольку обычно применяемый для этого протокол TWAMP-Control здесь отсутствует). Спецификация механизма генерации ключей выходит за рамки документа.

Нормативные документы

[RFC4656] Shalunov, S., Teitelbaum, B., Karp, A., Boote, J., and M. Zekauskas, "A One-way Active Measurement Protocol (OWAMP)", RFC 4656, September 2006.

[RFC2681] Almes, G., Kalidindi, S., and M. Zekauskas, "A Round-trip Delay Metric for IPPM", RFC 2681, September 1999.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.

[RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO 10646", STD 63, RFC 3629, November 2003.

[RFC5198] Klensin, J. and M. Padlipsky, "Unicode Format for Network Interchange", RFC 5198, March 2008.

Дополнительная литература

[RFC3692] Narten, T., "Assigning Experimental and Testing Numbers Considered Useful", BCP 82, RFC 3692, January 2004.

Адреса авторов

Kaynam Hedayat

Brix Networks

285 Mill Road

Chelmsford, MA 01824

USA

EMail: khedayat@brixnet.com

URI: http://www.brixnet.com/

Roman M. Krzanowski, Ph.D.

Verizon

500 Westchester Ave.

White Plains, NY

USA

EMail: roman.krzanowski@verizon.com

URI: http://www.verizon.com/

Al Morton

AT&T Labs

Room D3 - 3C06

200 Laurel Ave. South

Middletown, NJ 07748

USA

Phone +1 732 420 1571

EMail: acmorton@att.com

URI: http://home.comcast.net/~acmacm/

Kiho Yum

Juniper Networks

1194 Mathilda Ave.

Sunnyvale, CA

USA

EMail: kyum@juniper.net

URI: http://www.juniper.com/

Jozef Z. Babiarz

Nortel Networks

3500 Carling Avenue

Ottawa, Ont K2H 8E9

Canada

Email: babiarz@nortel.com

URI: http://www.nortel.com/

Полное заявление авторских прав

Интеллектуальная собственность

Перевод на русский язык

Николай Малых

nmalykh@protocols.ru