

IP Quality Monitor

Демон ССС и работа внутри NAT

Руководство пользователя

Версия 1.01 редакция 2024-05-14



ООО «Нетпроб»
123557, г.Москва,
пер. Электрический, дом 3/10 стр. 3,
офис 306А

Москва, 2024

1 Введение

IP Quality Monitor (далее в тексте IQM) – это аппаратно-программный комплекс, предназначенный для расчёта качественных метрик сетей TCP/IP. АПК состоит из двух частей — некоторого числа установленных на сети IQM-агентов (IQMA) и ядра системы (IQM менеджер, IQMM). Между агентами проводятся тесты, итоги которых поступают в ядро. Основные понятия приведены в отдельной документации.

IQMA способен выступать как в роли агента-инициатора, так и в роли сопряжённого для проведения тестов и сбора метрик. В данном руководстве рассматриваются способы установки агентов-инициаторов внутри сегментов сети, включая клиентские, в которых используется трансляция адресов (NAT). Вопросы возможной установки сопряжённых агентов, закрытых к доступу сетевой трансляцией следует рассматривать отдельно.

Настоящее руководство предназначено для системных администраторов, сопровождающих IQM. От администратора требуются следующие навыки:

- уверенное понимание принципов работы IQM,
- опыт работы со стеком протоколов TCP/IP,
- знание операционной системы Linux на уровне системного администратора.

2 Схемы включения IQM-агентов

Типовая схемой включения IQM-агента в сеть TCP/IP представлена на [рисунке 1](#). Где Q1 это агент-инициатор, а Q2 — сопряжённый. Возможны иные варианты включения, например, сопряжённый так же может включаться T-образно, либо агент может быть встроен в саму аппаратную платформу в случае использования стандартных протоколов. Однако при любом исполнении агента основным методом включения без использования специальных средств является T-образное при доступности агентов по обычной маршрутизируемой TCP/IP-сети.

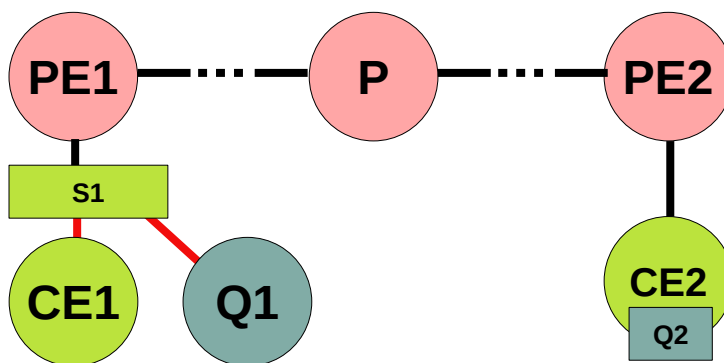


Рисунок 1. Агент-инициатор включен безопасно T-образно, сопряжённый ему использует пользовательское оборудование как программную платформу.

В настоящее время в сетях как операторов связи, так и у клиентов, даже физических лиц часто применяется технология сетевой трансляции. Причины мы опустим, здесь только заметим, что за счёт относительной дешевизны исполнения технологии, это позволяет включать развитые сегменты сетей с приемлемым уровнем безопасности, при этом экономно. Однако, этот же уровень созданной безопасности затрудняет доступ к агентам-инициаторам со стороны системы управления, если она находится не в том же сегменте.

Кроме того, у операторов связи существует особое требование к реализации т. н.

«выездного тестирования», когда персонал оператора временно находится на площадке клиента, осуществляя с помощью агента проверку метрик качества предоставляемых услуг. В этом случае, как правило, у клиента уже есть сетевая трансляция, доступ к системе управления агентами, конечно, реализовать возможно. Однако для доступа самой системы управления к агенту придётся проводить специализированные настройки, причём на оборудовании клиентской площадки. Это может быть неудобно как по техническим причинам (проще иметь типовые комплекты и типовые настройки), так и по организационным (оборудование может находиться под управлением клиента).

Конечно, можно для выездного тестирования использовать режим REMOTE_IQM, можно использовать синхронное тестирование с помощью соответствующей утилиты, но такие решения предполагают высокую квалификацию тестировщика, а на сегодня в связи существенный дефицит кадров. И дополнительно нагружать опытный персонал неразумно. Кроме того, удобный режим «тестирование по запросу», доступный в IQMM, в этом случае всё-таки недоступен по понятным причинам.

Приведём типовую схему включения клиента с возможным доступом к IQMM для выездного тестирования.

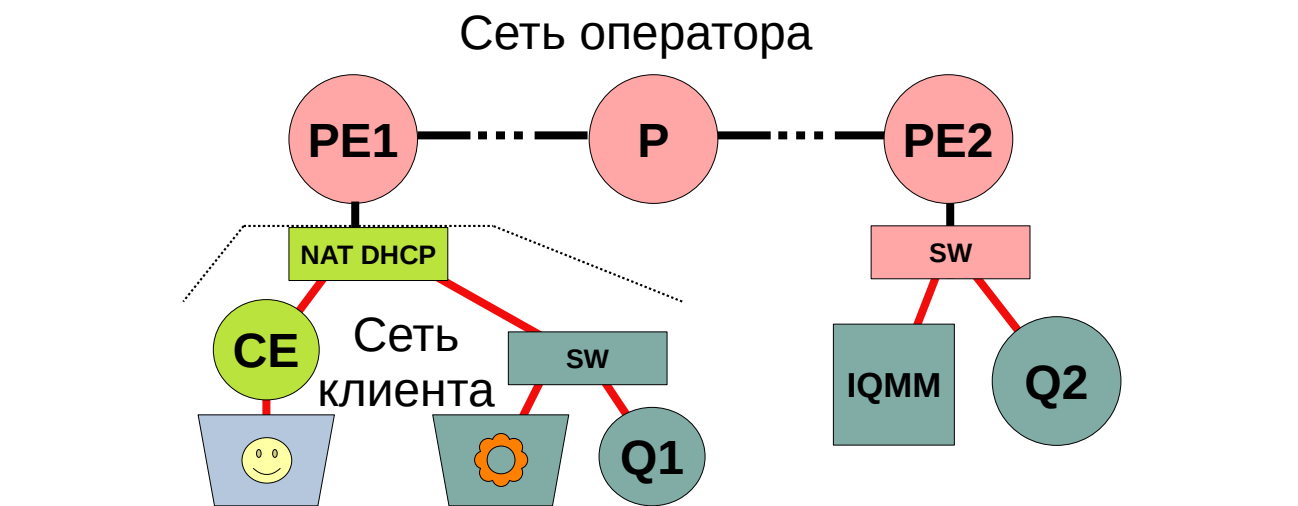
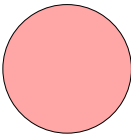







Рисунок 2. Агент-инициатор временно установлен клиенту для выездного тестирования, сопряжённый и система управления — в сети оператора.

где

- 

Сетевое оборудование оператора связи — маршрутизаторы P и PE, коммутаторы.
- 

Сетевое оборудование клиента — маршрутизаторы CE, коммутаторы, NAT-трансляторы, DHCP-серверы. Наиболее часто всё объединено в одном устройстве.
- 

Система управления и выделенный агент, выполняющий функции сопряжённого для большого числа выездных тестирований.



Рабочие места клиента.



Комплект выездного тестирования, включающий рабочее место инженера, агент-инициатор и факультативный коммутатор на случай, если у клиента не хватает портов доступа.

Поскольку на схеме присутствует несколько устройств, опишем более подробно какие функции ожидаются от каждого. Опустим сетевое оборудование, так как мы предполагаем высокую квалификацию сотрудников оператора связи и клиента. Эти устройства должны быть знакомые и вопросов не вызывать.

Рабочее место клиента — как правило настольный компьютер или переносной ноутбук (возможно, планшет). Именно он в большинстве случаев используется клиентом как своего рода «доказательство» того, что в сети оператора есть проблемы. Например, при просмотре любимых сетевых мемов клиент испытывает фрустрацию оттого, что не может вовремя ознакомиться с актуальным контекстом и ему приходится стыдиться собственной неготовности поддержать разговор в социальных сетях. Задача выездного тестирования — показать ему обратное, что вся сеть оператора связи работает 24/365 бесппроблемно, а возможный источник «помех» находится за пределами зоны ответственности оператора.

IQMM — это наша знакомая уже система управления. Именно на ней проводится финальный сбор данных, анализ, сигнализация о проблемах и, конечно, там есть web-интерфейс администратора и личный кабинет пользователей, которые могут не совпадать с клиентами оператора, а быть, допустим, филиалами или отделами или просто отдельными сотрудниками.

Q2 — это агент на одной из центральных площадок оператора. Он может быть как рядом с IQMM, так и на значительном расстоянии. Его задача в рамках выездного тестирования — отвечать на тесты, созданные агентами-инициаторами на клиентских площадках, впрочем, при запасе по аппаратным возможностям, он может выполнять и иные функции, от агента-инициатора магистрали до консольного сервера.

SW на площадке клиента — это коммутатор младшего класса. Предназначен для того, чтобы инженеру-тестировщику не создавать проблем, если у клиента отсутствуют свободные порты для включения.

Q1 — агент-инициатор на площадке клиента. Именно он проводит тестирование. Это ключевое оборудование. Для особо важных клиентом мы бы порекомендовали устанавливать его не на временной, а на постоянной основе.

Трапезия с шестерёнкой — это рабочее место выездного инженера, например, ноутбук. В основном используется для доступа к IQMM, к агенту Q1. В некоторых случаях возможна установка рабочего места, настройка через него агента или тестов от него и последующий вывоз на то время, пока агент будет проводить тестирование. После окончания такого тестирования возможна новая установка рабочего места для фиксации результата.

И последнее, но тоже важное. Не забывайте, что тестирование сетей следует осуществлять по кабельным соединениям. Wi-Fi даже на «последнем дюйме» вносит такое большое число проблем, что его смело можно не рекомендовать для тестирования метрик качества. Только кабель!

Итак, схема включения ясна, теперь опишем потоки трафика, которые ожидаются при тестировании и возможные проблемы, которые могут нас ожидать при их создании.

Сеть оператора

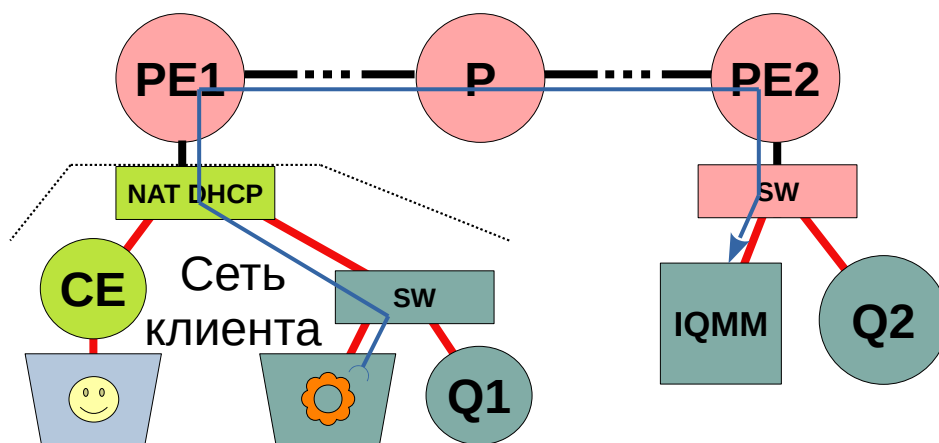


Рисунок 3. С рабочего места инженера проводится запрос на доступ к IQMM.

где



Направление web-трафика между рабочим местом инженера и системой управления.

Поскольку система управления доступна на сети оператора для рабочих задач, несложно сделать так, чтобы она была доступна и для нужных клиентских включений (допустим, путём прописывания адресов клиента в «белый список», разрешающий доступ). Поэтому в целом, для web-трафика выездного инженера по обращению в IQMM на [рисунке 3](#) препятствий нет. То есть, можно проводить конфигурацию агентов и тестов, просматривать отчётность и так далее. Однако, не будем забывать, что адреса внутри сети клиента не доступны с системы управления. Поэтому конфигурацию пробника Q1 невозможно выполнить по типу агента IQM. Так как такая конфигурация предполагает прямой (или косвенный, при дополнительных настройках сетевых устройств) доступ от IQMM к агенту.

Поэтому в качестве временного решения до установки агента, поддерживающего технологию CCC (Core Control Connector), возможно применение уже знакомого метода обновления конфигурации агента по типу REMOTE_IQM. Напомним, что в этом случае на IQMM в процессе создания или изменения конфигурации создаётся файл, который в дальнейшем по cron-планировщику агент способен получить командой `update_cfg.sh`. Правда, данная технология предполагает высокую квалификацию персонала для создания нужного набора ПО на Q1 и привязана к имени агента, которое так же нужно исправить в конфигурации вручную. Тем не менее, в качестве первого рабочего метода проведения выездного тестирования она может быть рекомендована. Разумеется, без использования тестов по требованию, что мы покажем наглядно в этой главе ниже.

Пример того, как направлен трафик конфигурации по REMOTE_IQM мы не приводим, так как он слабо отличается от [рисунка 3](#) и достаточно очевиден. Так же мы не приводим схему трафика по отравке CDR (файлов результатов), как аналогичную.

А вот то, как движется трафик при попытке запуска теста U0 следует показать. Это сделано на [рисунке 4](#). TCP-сессия контрольного протокола между Q1 и Q2 установится без проблем. Так как инициатор за NAT. Так же, без проблем будет осуществлён поток тестового трафика по UDP от Q1 до Q2. Однако, встречный поток (подробности работы протокола U0 в отдельной документации) от Q2 до Q1 встретит препятствие — точку трансляции адресов. Так как штатный NAT от самых разных производителей обычно поддерживает

специализированные протоколы только с помощью специальных программных модулей, можно смело утверждать, что с 99% вероятностью трафик от Q2 до Q1 остановится на точке NAT, не будет доставлен в точку Q1 и, таким образом, будет зафиксировано 100% потерь, хотя в реальности трафик прошёл почти по всей сети за исключением сегмента клиента.

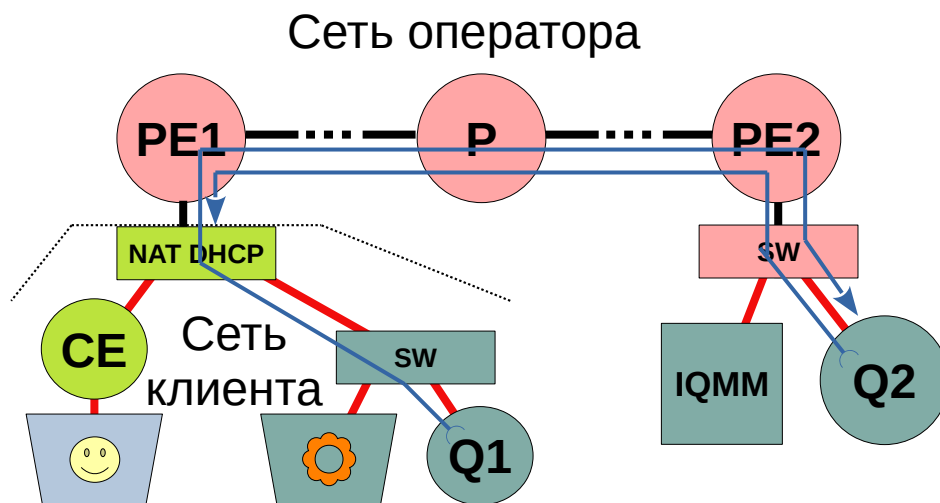


Рисунок 4. Агент Q1 пытается провести тест с агентом Q2 по протоколу U0.

где

- Q1 → Q2 Направление тестового трафика TCP и UDP от Q1 до Q2
- Q2 → Q1 Направление тестового трафика UDP от Q2 до Q1
(TCP трафик от Q2 до Q1 проходит по первому пути, так как инициатор установления соединения внутри NAT-сегмента)

Это хороший пример т. н. ложного срабатывания, когда определённые сетевые технологии блокируют проход тестового трафика и это следует каким-либо образом исправить. Можно, конечно, использовать иные протоколы, допустим U7, или I0, или протоколы, привязанные к производителю, или наконец RFC-5257 (T1/TWAMP). Однако, надо сразу отметить, что в протоколе U0 предусмотрены асимметрия генераторов, что отсутствует в большинстве альтернатив. А в случае включений клиента по ADSL или xPON разница в скоростях генерации очень полезна, так как позволяет верно тестировать входящую и исходящую полосы пропускания. Поэтому первой задачей для упрощения выездного тестирования должна стать следующая:

Задача 1.

Следует внести такие исправления в протокол U0 (либо создать аналог), чтобы при сохранении тех же настроек можно было проводить тестирование из клиентской сети, находящейся за сетевой трансляцией (NAT) без применения дополнительных настроек на сетевом оборудовании.

Теперь рассмотрим возможность тестирования по запросу из-за NAT. Поскольку тестирование по расписанию или по cron-шаблону предполагает некоторое время для работы, включая периоды ожидания, тестирование прямо в момент запуска было бы удобно как для инженера-тестировщика, так и для клиента.

Рассмотрим получающиеся потоки трафика при тестировании по запросу инженера. Они приведены на [рисунке 5](#).

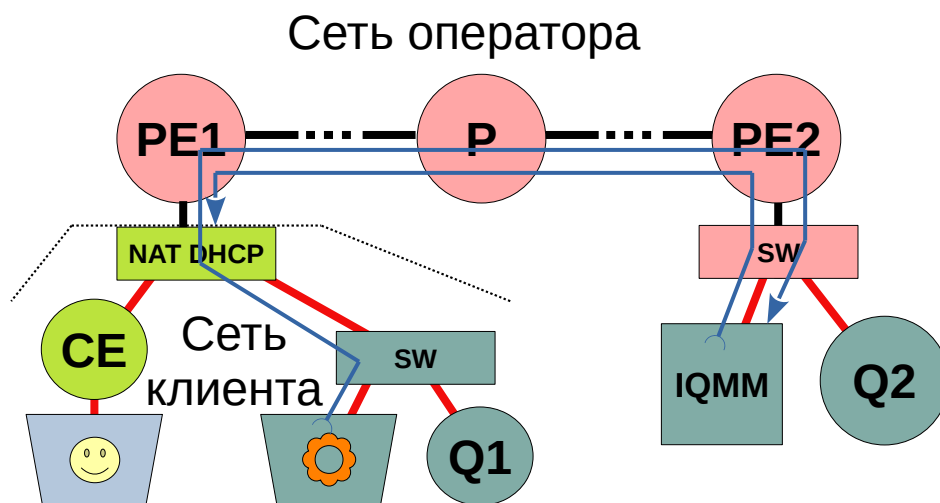
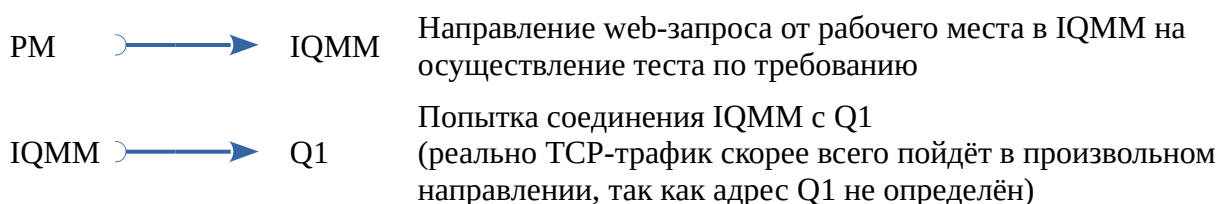


Рисунок 5. С рабочего места инженера проводится попытка провести тест по требованию между Q1 и Q2.

где



Запрос от рабочего места тестировщика до IQMM на [рисунке 5](#) в целом ничем не отличается от [рисунка 3](#). И тоже будет выполнен успешно, если вопросы безопасности уже решены (см. выше). А вот обратный запрос от IQMM до Q1 будет неудачным. В первую очередь потому, что адресация клиента обычно произвольная, более того, часто при умолчательных настройках оборудования может пересекаться с другим клиентом. Таким образом, штатно, без дополнительных настроек или написания нужного ПО, мы не получим доступа от IQMM к агенту-инициатору, а без этого тестирование по запросу невозможно. Следовательно нам нужно сформулировать ещё одну задачу:

Задача 2.

Следует создать программный модуль, который позволит подсоединяться к управляющему каналу агента-инициатора с тем, чтобы управлять им как если бы он был установлен в обычной маршрутизируемой сети TCP/IP.

Итак, теперь, когда наши цели ясны, а задачи определены, опишем то, как они были решены не только для работы по выездному тестированию, но и вообще для успешного функционирования агента-инициатора за NAT.

3 Протокол U1

Задачу номер один мы решили путём создания протокола U1, аналога по поведению U0, только умеющего применять специальное поведение для создания в NAT т.н. «транслированной пары», позволяющей прозрачно пропустить UDP-трафик. Для пользователя всё выглядит абсолютно прозрачно — просто выбирается нужный протокол в тесте.

Покажем, какие потоки трафика создаёт протокол U1 между агентами.

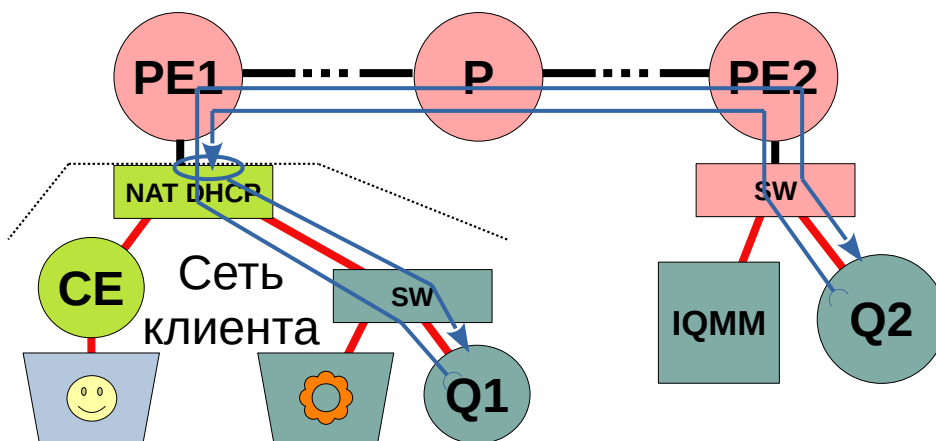
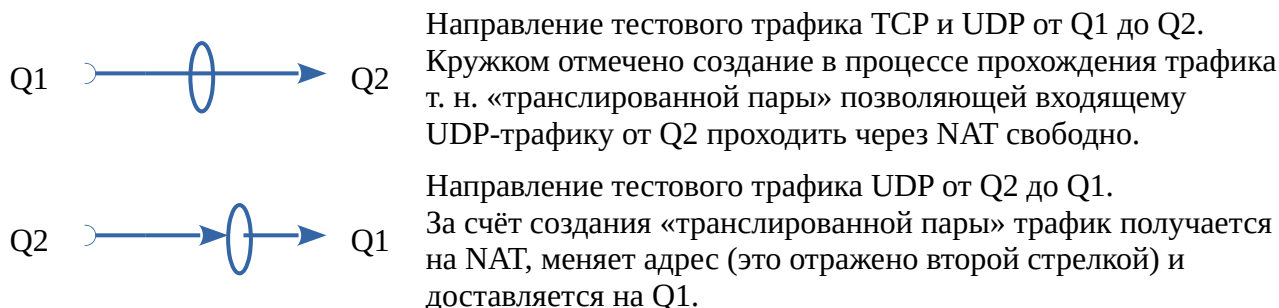


Рисунок 6. Агент Q1 пытается провести тест с агентом Q2 по протоколу U1.

где



Подробнее поведение агентов можно описать так:

1. Агент Q1 создаёт управляющее соединение по TCP-протоколу между Q1 и Q2. За счёт того, что оно создаётся изнутри NAT-сегмента, сложностей обычно нет.
2. В процессе создания двунаправленных потоков UDP-трафика первым создаётся поток от Q1 до Q2, дополнительно сообщая нужные настройки на Q2, позволяющие синхронизировать поток от Q2 до Q1. Тем самым создаётся «транслированная пара».
3. Включается поток трафика от Q1 до Q2.
4. Включается поток трафика от Q2 до Q1. Поскольку фактически этот поток направляется на исходящий адрес NAT, где «транслированная пара» уже есть, происходит дополнительная к TCP трансляция и пропуск трафика от NAT-устройства до Q1.
5. Время от времени потоки сравнивают синхронизацию с тем, чтобы сохранялась «транслированная пара».

В итоге для пользователя агента (инженера выездного тестирования, либо клиента, либо иного) пропуск трафика от Q1 до Q2 выглядит практически так же, как если бы эти

агенты находились в обычной маршрутизируемой сети TCP/IP и применяли протокол U0. Что позволяет получать все те же метрики качества, что и для U0 за исключением параметров G.107 (синтетический MOS и R-factor), которые требуют слишком маленького размера пакета, что невозможно реализовать в U1.

В случае, если по пути следования пакетов применяется несколько точек трансляции адресов (NAT) т. н. «транслированные пары» будут создаваться на каждой. Таким образом, прозрачность пропуска UDP-трафика будет сохранена.

В случае, если NAT по пути следования пакетов не применяется, работа будет проводиться почти аналогично протоколу U0 за вычетом несколько больших накладных расходов на трафик и несколько иначе выстроенных таймеров ожидания. Для рядового пользователя разница не будет видна.

4 Схема включения ССС-демона

Задача номер два была решена иначе. Так как штатно агент является входящей точкой управления (то есть слушает запросы от системы управления и отвечает на них), то неразумно менять уже установившееся знакомое поведения на что-то новое. Гораздо правильнее реализовать альтернативу, чтобы у пользователя всегда был выбор.

Поэтому было выбрано следующее решение. На хосте, находящемся в обычной маршрутизируемой сети запускается специальный прокси-демон **cccd** (далее ССС-демон). Мы рекомендуем запускать его на системе управления, так как создаваемая нагрузка невелика, но это не строго обязательно. Демон умеет принимать запросы как от агентов, так и от системы управления. В целях безопасности предусмотрено использование разделяемых секретных слов, аналогичных по поведению т. н. SNMP community.

Опишем поведение агента с модулем ССС, который позволяет подсоединяться к ССС-демону.

Сразу после запуска и начальной конфигурации агент при наличии нужных настроек (модуль включен, IP-адрес демона известен) проводит попытку соединиться по системному TCP-порту 1192, который слушает ССС-демон. В случае удачного соединения и совпадения секретных слов, агент проводит регистрацию своего имени (SID) внутри демона. Будьте аккуратны, пожалуйста, пересечение имён агентов недопустимо! После регистрации агент готов принимать команды управления по установленному виртуальному каналу. Если за время таймаута команда не поступила, агент закрывает соединение и создаёт его вновь. Если команда поступила, агент выполняет их так, как будто бы получил по порту управления 1189. По завершении сессии и закрытию соединения, агент создаёт его вновь.

Поведение на стороне системы управления иное. Если система управления видит, что агент, на который нужно либо отослать конфигурацию, либо выполнить команды тестов по требованию, имеет тип IQM, она поступает так, как и прежде. Если же тип CIQM, то система управления проводит подсоединение к ССС-демону, регистрирует себя со своим разделяемым секретным словом (это сделано специально, чтобы агент не мог подделывать запросы системы управления), и даёт запрос на соединение с агентом по его имени. Будьте внимательны, не по IP-адресу, как в случае с IQM-типом. Если такой агент зарегистрирован, ССС-демон производит проключение виртуального канала на установленный агентом с указанным именем, чтобы тот мог принять команды системы управления. Если такого агента нет, возвращается ошибка и система управления отображает её пользователю.

На рисунке 7 приведены потоки трафика, которые создаёт агент при регистрации на ССС-демоне.

Сеть оператора

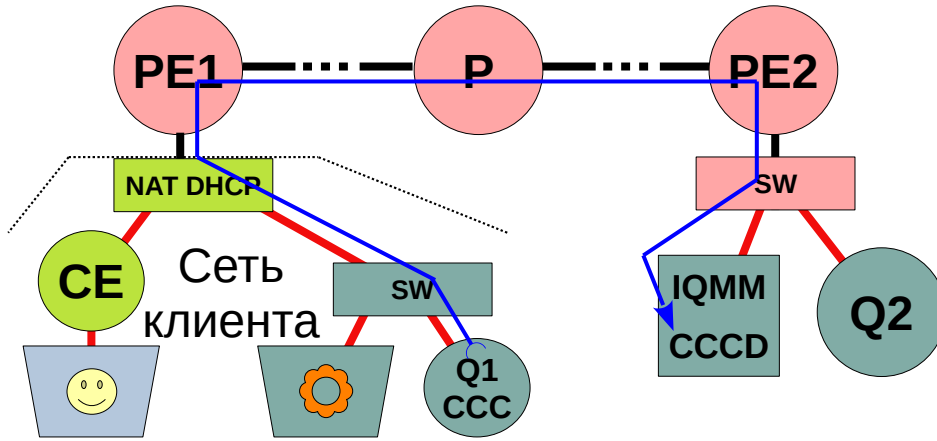



Рисунок 7. Агент Q1 с модулем CCC автоматически подсоединяется к CCC-демону на хосте IQMM.

где

 Установка виртуального канала между агентом Q1 и CCC-демоном

Параллельно с установленным соединением, агент может выполнять все свои обычные функции. Допустим, если в вашей настройке используется прежний режим REMOTE_IQM, то дополнительное соединение хотя и устанавливается, но фактически для работы не используется. Это проиллюстрировано на [рисунке 8](#).

Сеть оператора

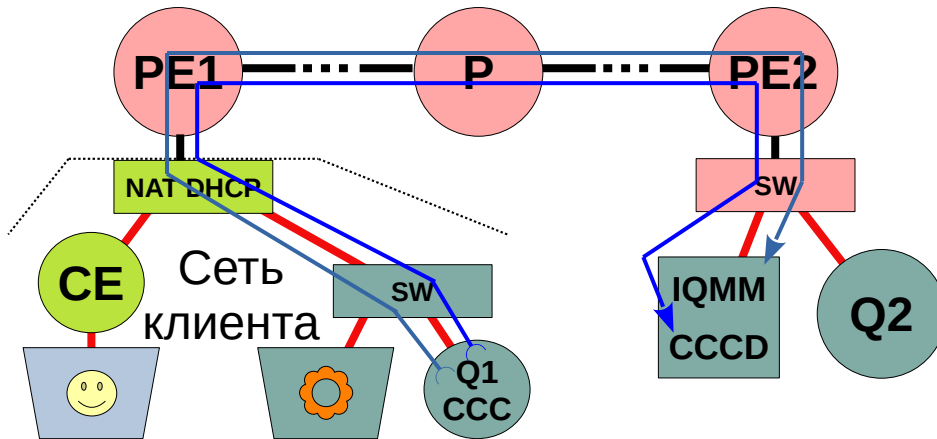



Рисунок 8. Агент Q1 с модулем CCC автоматически подсоединяется к CCC-демону на хосте IQMM. Параллельно проводится обновление конфигурации по технологии REMOTE_IQM.

где

 Установка виртуального канала между агентом Q1 и CCC-демоном.

 Обновление конфигурации Q1 с IQMM по технологии REMOTE_IQM.

А теперь покажем, как ведут себя потоки трафика, когда инженер запускает тест по требованию U1 от агента Q1 на сопряжённый Q2. Начало работы показано на [рисунке 9](#).

Сеть оператора

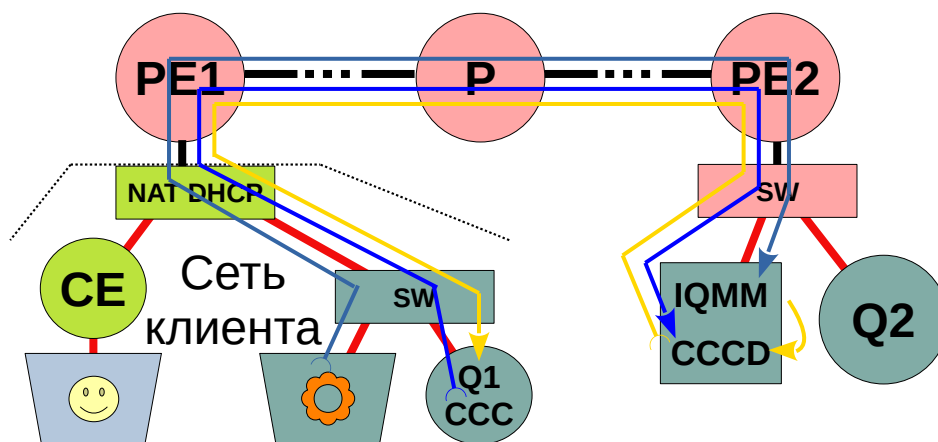





Рисунок 9. Агент Q1 с модулем CCC автоматически подсоединяется к CCC-демону на хосте IQMM. После подсоединения инженера на IQMM, система управления даёт указанную команду.

где

-  Установка виртуального канала между агентом Q1 и CCC-демоном.
-  Направление web-запроса от рабочего места в IQMM на осуществление теста по требованию
-  Использование виртуального канала между Q1 и CCC-демоном для выдачи команды на агента для исполнения теста по требованию или запроса итогов теста.

Последовательность действий:

1. Агент типа CIQM регистрирует своё имя (например AL) на CCC-демоне.
2. Инженер выдаёт команду на IQMM «исполнить тест по требованию» с типом U1.
3. IQMM видит в базе данных тип агента CIQM и для обращения к нему регистрируется на CCC-демоне как управление.
4. CCC-демон читает запрос на соединение с агентом с именем AL от IQMM и находит в пуле соединений соответствующий виртуальный TCP-канал. IQMM уведомляется об открытии канала и дальнейшие команды IQMM «исполнить тест по требованию» направляются в канал агента Q1 (с именем AL).
5. Агент Q1 увидев команды «исполнить тест по требованию», находит такой в конфигурации и начинает исполнение. Уведомление о старте отправляется на IQMM через всё ещё открытый канал. Канал закрывается.
6. Далее, как показано на [рисунке 6](#), проводится тестирование от агента Q1 до агента Q2. Параллельно с исполнением агент вновь открывает соединение с CCC-демоном и регистрируется там.
7. Инженер видит итоги запуска и в той странице, которая в IQMM отображает тесты по требованию, у него начинает работать т. н. «бегунок», показывающий примерный процент готовности результатов тестов.
8. После того как бегунок завершился либо ранее инженер выдаёт команду на IQMM

«выдать итоги теста по требованию».

9. IQMM видит в базе данных тип агента CIQM и для обращения к нему регистрируется на ССС-демон как управление.

10. ССС-демон читает запрос на соединение с агентом с именем AL от IQMM и находит в пуле соединений соответствующий виртуальный TCP-канал. IQMM уведомляется об открытии канала и дальнейшие команды IQMM «выдать итоги теста по требованию» направляются в канал агента Q1 (с именем AL).

11. Агент Q1 увидев команды «выдать итоги теста по требованию», находит таковые и выдаёт, либо создаёт ответ с ошибкой. Ответ отправляется на IQMM через всё ещё открытый канал. Канал закрывается.

12. IQMM отображает на странице для инженера итоги теста по требованию.

На деле всё выглядит существенно проще и естественнее, мы расписали все этапы подробно, чтобы у сетевых инженеров было правильное представление о характере трафика, создаваемого агентами, ССС-демоном и IQMM.

5 Ключи управления ССС-демона

ССС-демон обычно не требует настроек. Мы приводим здесь все ключи командной строки для тонкой настройки данного программного обеспечения.

Ключ командной строки	Описание
--help	Выдать подсказку по поддерживаемым ключам командной строки с указанием формата, а так же значений по умолчанию и завершить работу
-h	—" —
--version	Выдать версию агента, список модулей, протоколов, версий сторонних библиотек и завершить работу
-V	—" —
--copyright	Выдать лицензионный договор присоединения и завершить работу
--pidfile=/path/file.pid	Сохранять цифровой идентификатор процесса в файле /path/file.pid Если путь к файлу не указан, будет использоваться /run/cccd.pid Данный ключ рекомендуется использовать с именем, так как файл с идентификатором создаётся в любом случае.
-P /path/file.pid	—" —
--level=N	Уровень журналирования (1-7), чем больше, тем подробнее.

Если администратору не нужен начальный вывод предупреждений и ошибок, можно на старте указать следующий ключ:

Ключ	Значение по умолчанию	Описание
<code>--nostdout</code>	Выводить на начальном этапе ошибки и предупреждения.	Вывод на начальном этапе запуска ошибок

Если в системе доступно ведение журнала в файл, можно на старте использовать следующий ключ:

Ключ	Значение по умолчанию	Описание
<code>--log=S</code>	Не вести журналирование.	Записывать журнал в файл

Если в системе доступно ведение журнала через системный демон, можно на старте использовать следующий ключ:

Ключ	Значение по умолчанию	Описание
<code>--syslog</code>	Не вести журналирование.	Записывать журнал в syslog

Если в системе есть `fork(2)`, можно на старте использовать следующий ключ:

Ключ	Значение по умолчанию	Описание
<code>--nodaemon</code>	Становиться демоном	Не отсоединяться от контрольного терминала, сохраняя работу агента без выхода в shell-сессию. Полезно для отладки либо на экзотических ОС. При штатной работе не применяется.

Если в системе есть `IP_RECVTOS` либо `IPV6_RECVTCLASS`, можно на старте использовать следующий ключ:

Ключ	Значение по умолчанию	Описание
<code>--notos</code>	Обрабатывать поле TOS или TCLASS каждого пакета UDP либо ICMP.	Не использовать чтение TOS и TCLASS. Как правило это не нужно, но возможно для отладки либо в экзотических конфигурациях.

Если в системе есть IP_RECVTTL либо IPV6_RECVNOPLIMIT, можно на старте использовать следующий ключ:

Ключ	Значение по умолчанию	Описание
--nottl	Обрабатывать поле TTL или NOPLIMIT каждого пакета UDP либо ICMP.	Не использовать чтение TTL и NOPLIMIT. Как правило это не нужно, но возможно для отладки либо в экзотических конфигурациях.

Если в системе есть возможность фиксации входящего адреса на каждом пакете, можно на старте использовать следующий ключ:

Ключ	Значение по умолчанию	Описание
--nosamesend	Отсылать ответы с того же адреса, на который они направлены.	Отсылать ответы согласно маршрутной таблице. Как правило это не нужно, но возможно для отладки либо в экзотических конфигурациях.

Если в системе есть поддержка IPv6, можно на старте использовать следующие ключи:

Ключ	Значение по умолчанию	Описание
--ipv4-only -4	Работать с двумя типами адресов одновременно.	Включить работу только для адресов IPv4.
--ipv6-only -6	Работать с двумя типами адресов одновременно.	Включить работу только для адресов IPv6.

Общесистемные ключи закончились, теперь приводим те, от которых зависит поведение ССС-демона.

Ключ	Значение по умолчанию	Описание
--ra-port=N	1192	TCP-порт на котором происходит приём входящих сообщений от агентов и IQMM
--ra-secret=S	Сообщается по безопасным каналам связи	Разделяемая секретная строка для регистрации агентов

Ключ	Значение по умолчанию	Описание
<code>--mgr-secret=S</code>	Сообщается по безопасным каналам связи	Разделяемая секретная строка для регистрации системы управления
<code>--ra-sessions=N</code>	5	Максимальное число одновременных сессий с агентами. Рекомендуется выставить примерно в 1.5 раза больше, чем действующих агентов типа CIQM в целях запаса.
<code>--ra-time-res=N</code>	5000 миллисекунд	Частота таймера проверок на начало регистрации агентом по вновь открытому каналу
<code>--ra-timeout=N</code>	30000 миллисекунд	Таймаут для исполнения регистрации агентом. Если за это время по вновь открытому каналу от агента не пришёл запрос на регистрацию, сессию закрыть.
<code>--ra-access=S</code>	Ограничений нет	Список IP-адресов и масок через запятую, с которых разрешена регистрация агентов. Например: X.X.X.X/24,Y.Y.Y.Y/8
<code>--mgr-access=S</code>	Ограничений нет	Список IP-адресов и масок через запятую, с которых разрешена регистрация систем управления. Например: X.X.X.X/24,Y.Y.Y.Y/8

6 Примеры запуска тестов из-за NAT

Поскольку указанные примеры могут быть сложны для понимания, мы приведём непосредственные примеры запуска тестов изнутри клиентской сети при управлении через систему IQMM в сети Интернет.

Для начала покажем настройки сети на самом пробнике и на ближайшем маршрутизаторе, выступающем в роли как шлюза по умолчанию, так и NAT-транслятора.

```

192.168.32.70 - PuTTY
Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
root@wtplug:~# ip addr
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 16436 qdisc noqueue state UNKNOWN
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP> mtu 1500 qdisc mq state UNKNOWN qlen 1000
    link/ether 00:e1:75:50:22:cc brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.1/24 brd 192.168.1.255 scope global eth0
3: eth1: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc mq state UP qlen 1000
    link/ether 00:e1:75:50:22:cd brd ff:ff:ff:ff:ff:ff
    inet 192.168.32.70/24 brd 192.168.32.255 scope global eth1
    inet6 fe80::2e1:75ff:fe50:22cd/64 scope link
        valid_lft forever preferred_lft forever
4: sit0: <NOARP> mtu 1480 qdisc noop state DOWN
    link/sit 0.0.0.0 brd 0.0.0.0
root@wtplug:~# ip route
default via 192.168.32.1 dev eth1
192.168.1.0/24 dev eth0 proto kernel scope link src 192.168.1.1
192.168.32.0/24 dev eth1 proto kernel scope link src 192.168.32.70
root@wtplug:~#

```

Рисунок 10. Адресация и маршрутизация пробника plug-tushino.

На [рисунке 10](#) пробнику настроено получение на порту eth1 адресов по протоколу DHCP. Получен адрес из сети 192.168.32.0/24. В качестве шлюза по умолчанию выступает маршрутизатор по адресу 192.168.32.1.

```

192.168.32.1 - PuTTY
arp - display system ARP table
dhcp - display DHCP server status
ftp - display FTP server status
hotspot - display hotspot hosts
http - display HTTP server status
name-server - display DNS resolver configuration
nat - display network address translation table
neighbour - display system neighbour table
policy - display network policy table
route - display system routing table
service -

(config)> show ip route
=====
Destination          Gateway              Interface            Metric
=====
0.0.0.0/0            10.2.██████████     ISP                  0
10.2.██████████      0.0.0.0             ISP                  0
10.██████████        0.0.0.0             Guest                0
192.168.32.0/24     0.0.0.0             Home                 0
217.10.32.4/32     10.2.██████████     ISP                  0
217.10.36.5/32     10.2.██████████     ISP                  0
217.10.44.35/32    10.2.██████████     ISP                  0
(config)>

```

Рисунок 11. Маршрутизация на шлюзе по умолчанию.

На [рисунке 11](#) у шлюза и транслятора, к которому подключен агент, настроено несколько сетей. Первая сеть — ISP с адресом 10.2.X.X/X через которую осуществляется подключение к оператору связи. Обращаем ваше внимание, что адреса выделены из т. н. «серой» сети 10.0.0.0/8, то есть по факту после доступа в интернет, трансляция адресов будет исполняться дважды — первый раз в сети клиента 192.168.32.0/24, второй раз — где-то у оператора связи.

Вторая сеть — Guest, то есть гостевая с адресацией 10.Y.Y.Y/Y, она нас не интересует.

Третья сеть — Home, то есть внутренняя сеть клиента, куда подключен пробник plug-tushino.

Дополнительные адреса предназначены для доступа к DNS-серверам оператора связи и получают по DHCP от его сети автоматически.

```
root@wtplug:~# ps -ax|grep iqm_a
Warning: bad ps syntax, perhaps a bogus '-'? See http://procps.sf.net/faq.html
 550 pts/0    S+        0:00    grep iqm_a
1231 ?        Ssl      12:20   /usr/local/iqm_agent/iqm_agent --pidfile=/var/run/iqm_agent.pid --level=5 --syslog --config=/usr/local/iqm_agent/iqm_agent.conf @/usr/local/iqm_agent/iqm_agent.args
root@wtplug:~# more /usr/local/iqm_agent/iqm_agent.args
--server-config=http://[REDACTED].net/iqm/getconfig?login=loadcfg&pass=[REDACTED]
&agent_ip=$AGENTIP --srv-cfg-first --srv-cfg-cli-only --ccc-core-ip=[REDACTED].net
root@wtplug:~#
```

Рисунок 12. Ключи запуска агента на пробнике plug-tushino.

Теперь рассмотрим командную строку запуска агента. Она представлена на [рисунке 12](#). Ключи по настройке файла с идентификатором процесса, журналированием и именем файла с конфигурацией для вас уже должно быть знакомо. Остановится на особенностях с модулями агента FOPT, SRV-CFG и CCC.

В командной строке запуска, агенту передан файл с ключами командной строки @/usr/local/iqm_agent/iqm_agent.args. За это отвечает модуль FOPT. Сделано так для удобства, чтобы не править системный файл запуска, так как и сервер конфигурации и адрес CCC-демона могут в процессе эксплуатации не только меняться, но и клонироваться внутри развитой сети.

В файле с ключами указаны ключи модуля SRV-CFG для автоматической синхронизации с системой управления. Это --server-config=URL. Обращаем внимание, что на картинке в URL закрыты критичные для безопасности параметры — имя хоста с системой управления (может употребляться и адрес) и пароль пользователя loadcfg. Кроме того, для автоматического получения конфигурации по внешнему IP-адресу пробника (в данном случае 192.168.32.70, на DHCP-сервере он зафиксирован) употребляется переменная \$AGENTIP. В момент создания запроса на систему управления вместо этой переменной будет подставлен текущий IP-адрес (это описано в отдельной документации подробнее). Возможно употребление переменной \$AGENTNAME и иных, если вам это необходимо. Дополнительно указаны ключи --srv-cfg-first и --srv-cfg-cli-only, задающие нужные режимы получения конфигурации («Вначале читать конфигурацию из системы управления, а затем из файла» и «использовать только URL, указанный в командной строке»).

Для подсоединения к CCC-модули указан ключ модуля CCC --ccc-core-ip=HOST. Обращаем внимание, что имя хоста в целях безопасности закрыто.

В результате применения всех этих ключей, агент работает так:

1. на старте, а так же время от времени (подробности в сторонней документации) происходит обновление конфигурации от системы управления. Файл с конфигурацией используется в крайнем случае, если система управления недоступна.

2. Агент подсоединяется к ССС-демону и продолжает поддерживать соединение в ожидании команд от системы управления (в данном примере на исполнение файлов по требованию). Если ССС-демона нет, к агенту обратиться от системы управления можно только традиционным методом — через порт 1189.

```
[cae@ [REDACTED] ~]$ w
 12:44:49 up 140 days, 12:47,  1 user,  load aver.
USER      TTY      FROM          LOGIN@   IDLE
cae       pts/0    46.39.34.105  12:44   0.00s
[cae@ [REDACTED] ~]$ █
```

Рисунок 13. Внешний адрес plug-tushino при доступе в интернет.

На [рисунке 13](#) видно, что внешний IP-адрес после выхода в интернет через сеть оператора связи будет не из блока 10.0.0.0/8, и не из блока 192.168.32.0/24. Таким образом, видно, что трансляция осуществляется.

Рассмотрим конфигурацию агента plug-tushino, занесённую в систему управления. Она представлена на [рисунке 14](#). Как обычно, критичные по безопасности значения намеренно закрыты.

Parameter	Value	Default
Multiconf	<input type="text"/> Separation char <input type="text"/> Quotation char	Enter CSV configuration file
Agent ID	plug-tushino	
Agent name *	plug-tushino	
Agent IP *	192.168.32.70	
Agent password *	xyz	xyz
Control password *	ZCZC	ZCZC
Agent type	CIQM	IQM
Network layer	CORE	CORE
SNMP profile		
Zone *	MГTC	1
Core IP *	212	127.0.0.1
CDR filesize (NoR)	60	60
CDR file timeout (min)	5	5
CDR send timeout (min)	5	5
Spool dir	/tmp	/tmp
CDR transport script	FTP	/usr/local/iqm_agent/sender.pl
FTP user		iqm
FTP password		sla
Config source		
Listen port	1189	1189
Server timeout (sec)	3	

Рисунок 14. Конфигурация агента в системе управления.

Видно, что агент имеет тип CIQM. Это означает доступ к управлению через CCC-демон (в данном случае запущенный на IQMM). В качестве адреса используется внутренний адрес в сети клиента. Остальные настройки сделаны обычными. То есть, фактически, при указании типа CIQM агент ведёт себя так же, как и IQM или REMOTE_IQM, отличается только доступ управления. SRV-CFG можно получать для всех агентов указанных типов, разницы между ними не делается. Дополнительно отметим, что при заведении агента типа CIQM невозможно получить по SNMP настройки интерфейсов, поэтому в этом случае они проводятся по контрольному соединению через CCC-демон. То есть настройки SNMP-профиля для CIQM-агентов можно не указывать.

Теперь покажем настройки тестов (их будет два, U1 и DNS). Первый на рисунках 15 и 16.

Multiconf	
Agents	
Separation char	
Quotation char	
Test ID	plug-tushino_IQMM_U1_od OnDemand
Test name *	plug-tushino_IQMM_U1_od
Class (IP Precedence or DSCP)	RT
SLA policy profile	
Service	Internet
Provider	Default
SRC agent *	plug-tushino
DST agent *	IQMM
Source IP	NAT
DST agent IP *	212
Local port	
Remote port	
Control port	
Test frequency (sec)	600
Cron-like template	
Number of probes	2000
Number of probes to ignore	
URL	
URL-test query interval (ms)	
Cookie file	
HTTP User-Agent	

Рисунок 15. Конфигурация теста U1 между plug-tushino и IQMM. Начало

Видно, что тест проводится между агентом plug-tushino и агентом IQMM. Адреса выбираются стандартные, число пакетов указывается 2000, с тем чтобы тест исполнялся некоторое время (точные формулы смотрите в «Основах тестирования сетей TCP/IP для служб эксплуатации»), частота запуска не влияет, отметки трафика RT (5 класс).

Number of probes	2000
Number of probes to ignore	
URL	
URL-test query interval (ms)	
Cookie file	
HTTP User-Agent	
Content download timeout	
DST agent type	A
Test type	U1
VoIP profile	
Test command (for LOCAL or CMD tests)	
CMD options	
CMD timeout kill signal	
Test timeout (sec)	
Test-specific parameters	
Send config to CMD's STDIN	
Log CMD's STDOUT	
Redirect CMD's STDERR to STDOUT	
Enabled	1
Packet size (B)	256
On demand test	1
Bandwidth for UDP tests (Kb/s)	128
Reverse bandwidth for UDP tests (Kb/s)	1024
Expected bandwidth for UDP tests (Kb/s)	
Expected reverse bandwidth for UDP tests (Kb/s)	

Рисунок 16. Конфигурация теста U1 между plug-tushino и IQMM. Завершение

Тип теста U1, тест включен, признак «по запросу» так же включен, размер пакета 256 байт (помимо заголовков), скорость генератора от plug-tushino до IQMM 128 килобит в секунду, от IQMM до plug-tushino 1024 килобита в секунду. Никаких незнакомых настроек нет, всё для администраторов IQM выглядит привычно.

Теперь настройки для теста DNS (в сети оператора связи, на котором проводилось тестирование, доступны только его серверы). Они представлены на [рисунках 17 и 18](#).

Agents	Interface	Test
Quotation char		
Test ID	plug-tushino_akado-dns1_DNS_od	OnDemand
Test name *	plug-tushino_akado-dns1_DNS_od	
Class (IP Precedence or DSCP)	RT	
SLA policy profile		
Service	Internet	
Provider	Default	
SRC agent *	plug-tushino	
DST agent *	akado-dns1	
Source IP	NAT	
DST agent IP *	217.10.32.4	
Local port		
Remote port		
Control port		
Test frequency (sec)	600	
Cron-like template		
Number of probes	20	
Number of probes to ignore		
URL	dns://www.net-probe.ru	
URL-test query interval (ms)		
Cookie file		
HTTP User-Agent		
Content download timeout		

Рисунок 17. Конфигурация теста DNS между plug-tushino и оператором. Начало

Видно, что тест проводится между агентом plug-tushino и dns-сервером оператора связи. Адреса выбираются стандартные, число пакетов указывается 20 (оператор не выдерживает большой нагрузки), частота запуска не влияет, отметки трафика RT (5 класс), впрочем, они скорее всего отключатся. Запрашиваем адрес сервера www.net-probe.ru (он точно есть). Подробнее про тесты DNS в отдельной документации.

Number of probes	20
Number of probes to ignore	
URL	dns://www.net-probe.ru
URL-test query interval (ms)	
Cookie file	
HTTP User-Agent	
Content download timeout	
DST agent type	A
Test type	DNS
VoIP profile	
Test command (for LOCAL or CMD tests)	
CMD options	
CMD timeout kill signal	
Test timeout (sec)	
Test-specific parameters	
Send config to CMD's STDIN	
Log CMD's STDOUT	
Redirect CMD's STDERR to STDOUT	
Enabled	1
Packet size (B)	60
On demand test	1
Bandwidth for UDP tests (Kb/s)	64
Reverse bandwidth for UDP tests (Kb/s)	
Expected bandwidth for UDP tests (Kb/s)	
Expected reverse bandwidth for UDP tests (Kb/s)	
Creation date	2024-02-05 18:52:49.125822

Рисунок 18. Конфигурация теста U1 между plug-tushino и оператором. Завершение

Тип теста DNS, тест включен, признак «по запросу» так же включен, размер пакета 60 байт (помимо заголовков), скорость генератора от plug-tushino до оператора 64 килобит в секунду. Тоже ничего непривычного.

Теперь запускаем тест по запросу. Начало на [рисунке 19](#).

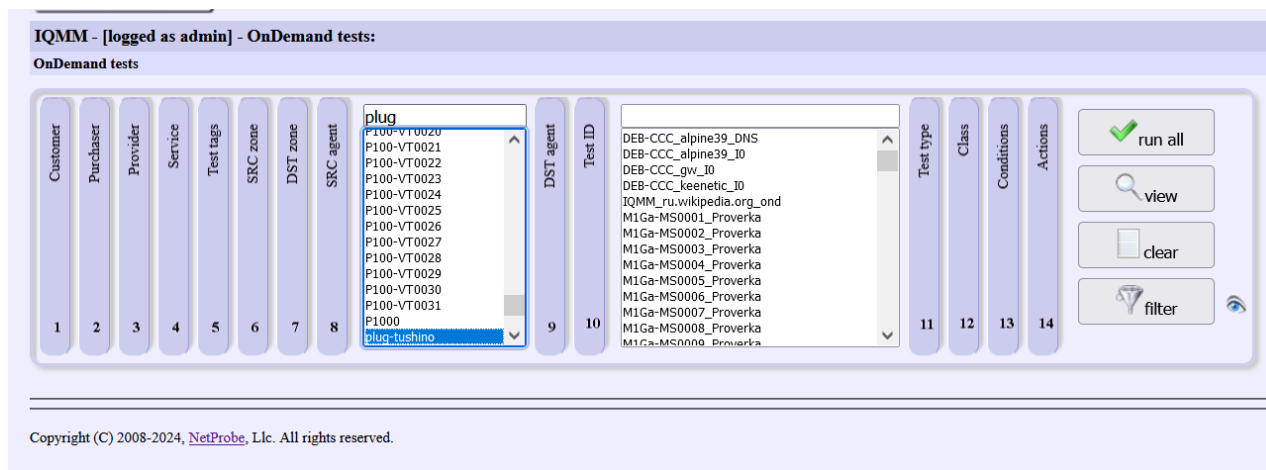


Рисунок 19. Начало работы с тестами по требованию. Выбираем агент-инициатор, набирая начало имени в верхнем поле и нажимаем Tab для отмены.

Теперь нажимаем кнопку Filter для отбора среди всех доступных тестов только тех, кто создан на plug-tushino. Результат виден на [рисунке 20](#).

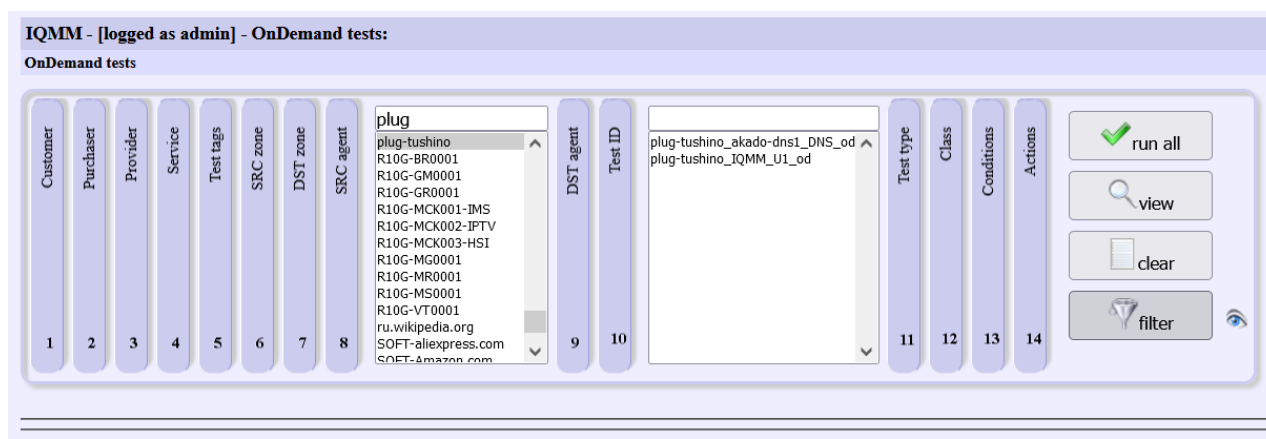


Рисунок 20. Фильтрация тестов только на те, что принадлежат агенту.

Теперь заметим, что в отличие от многих других режимов, в меню запуска тестов по требованию следует обязательно выбрать тесты для исполнения в списке. Если этого не сделать, будет результат, который показан на [рисунке 21](#). Так сделано намеренно, так как при большом числе тестов в списке (для более-менее развитой сети это норма), будет попытка исполнить сразу множество тестов, а это может привести к деградации сервиса на самой сети. В то же время, если пользователь отметил нужные тесты, предполагается, что он знает, что именно делает. Дополнительно отметим, что более 10 тестов выбирать не разрешается. Данный лимит может быть скорректирован по запросу пользователя системы.

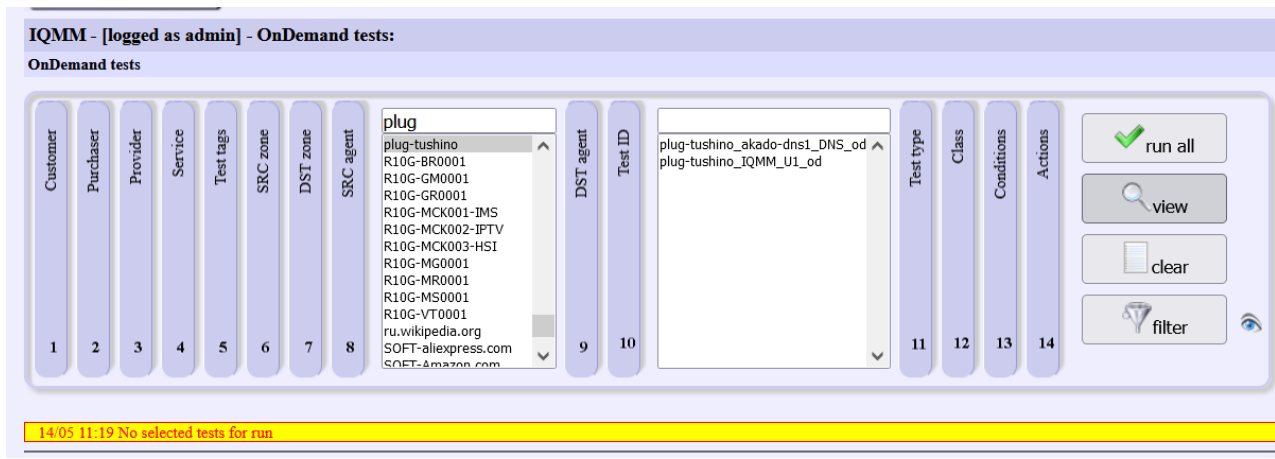


Рисунок 21. Нажатие View без выбора тестов. Выдана ошибка ниже фильтра.

Если же выбор сделан, будет отображение настроек для каждого из выбранных тестов в панели ниже фильтра, как показано на [рисунке 22](#).

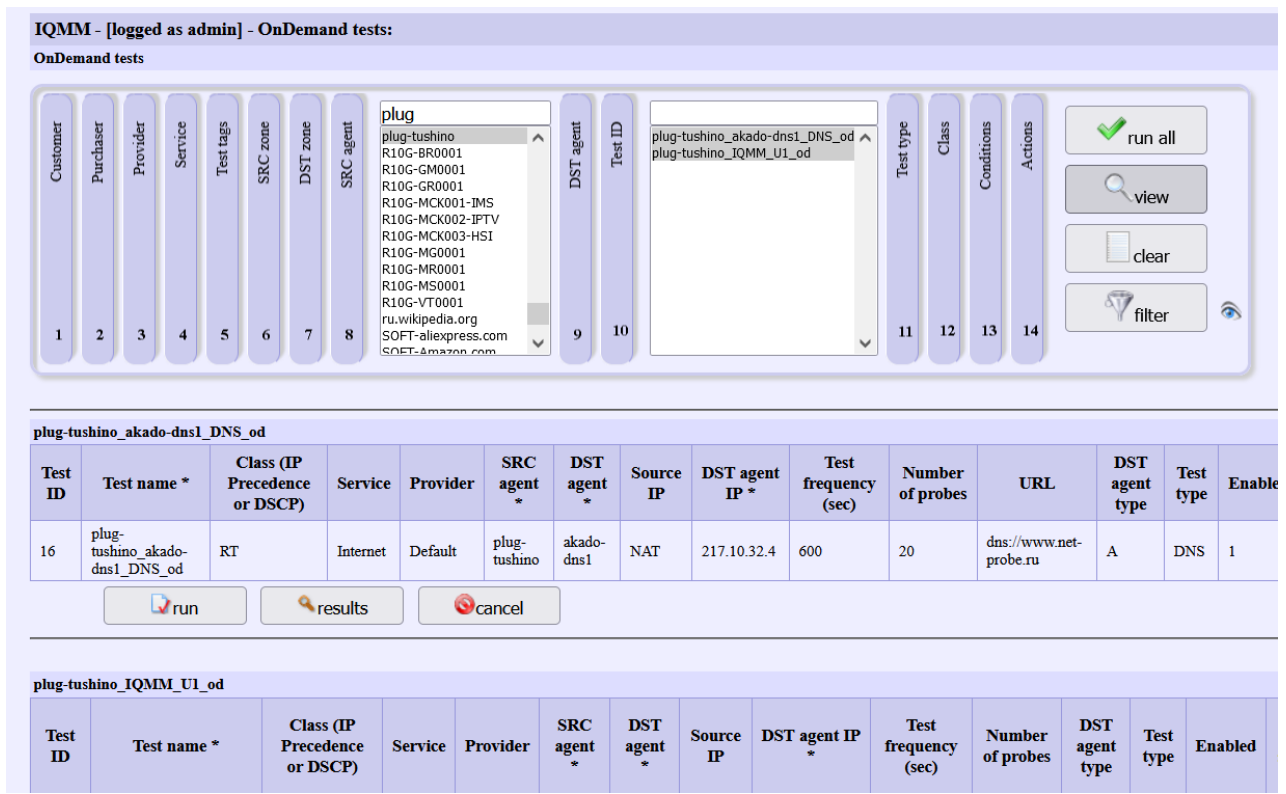


Рисунок 22. Нажатие View после выбора тестов DNS и U1.

Помимо настроек в табличной форме в панели ниже фильтра отображаются кнопки запуска, запроса результатов и принудительной остановки для каждого из тестов. Запускаем тест DNS, как показано на [рисунке 23](#).

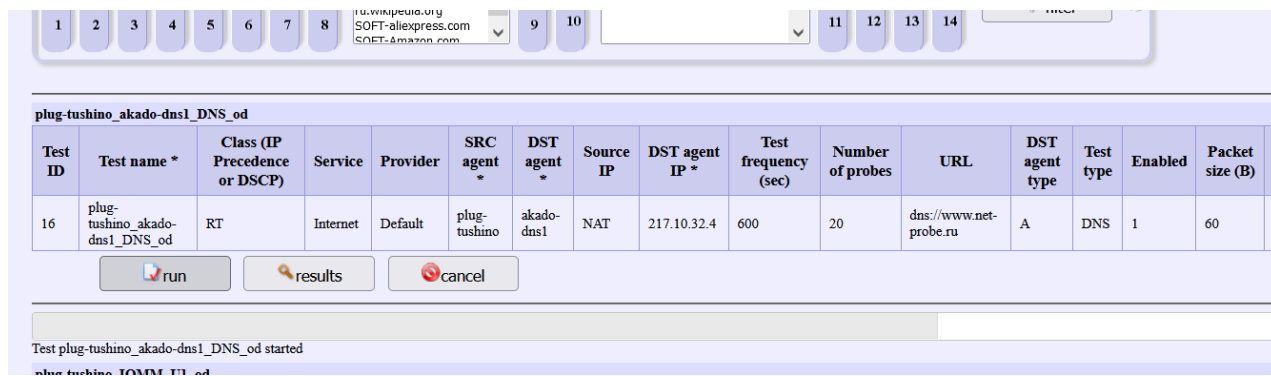


Рисунок 23. Нажата кнопка запуска теста по требованию DNS-типа. Виден «бегунок», отображающий примерный объём исполнения теста.

Когда бегунок дойдёт до конца, можно будет запрашивать результат. В принципе, это можно делать и в другой момент, но мы бы рекомендовали дождаться окончания теста. Агент хранит результаты по тесту некоторое время, так что запрашивать можно и позже. А если забыть запросить, они после исчерпания места во внутреннем хранилище удалятся. Размер хранилища достаточно велик, так что при текущей работе исчезновение результатов скорее исключение, чем правило. Итак запрашиваем результат кнопкой Results. Его видно на рисунках 24, 25, 26.

SID	DID	TestType	TID	TStart	TEnd	ServiceCode	ProtoServiceCode	SDBytes	DSBytes	SDLost	SDLostPercent	DSLost	DSLostPercent
plug-tushino	akado-dns1	DNS	plug-tushino_akado-dns1_DNS_od	2024-05-14 11:25:13.745281	2024-05-14 11:25:16.998107	4	0	880	0	10	50.0000000000	0	0.0000000000

Рисунок 24. Начало результатов теста по требованию типа DNS.

Видно, что потери при небольшом числе запросов (рисунок 17) составляют целых 50%. Конечно, можно сослаться на то, что оператор связи совсем не обязан каждому клиенту предоставлять полосу пропускания 64 килобита до своего DNS-сервера и содержать соответствующие серверные ресурсы. Тем более, что системные резолверы ОС общего назначения обычно более толерантны к качественным запросам DNS-протокола. Но всё-таки отметим, что потери — это ненормально.

SDBW	DSBW	SDBWPercent	DSBWPercent	MinRtt	AvgRtt	MaxRtt	SDMinDelay	SDAvgDelay	SDRMSDelay	SDMaxDelay	DSMinDelay	DSAvgDelay	DSRMSDelay	DSMaxDelay
37312	0	44.999758792	0.0000000000	11.258116000	13.808749000	17.975814000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000	0.0000000000

Рисунок 25. Продолжение результатов теста по требованию типа DNS.

В этом разделе интерес представляет только RTT, да и то приблизительный. Нормировать эти параметры не имеет смысла. Скорости же для DNS-ответов хотя и измеряются, не имеют значимости. Односторонние задержки, естественно, недоступны.

SDRemarked	SDRemarkedPercent	DSRemarked	DSRemarkedPercent	SDOOS	SDOOSPercent
10	100.000000000	0	0.000000000	1	10.000000000

Рисунок 26. Окончание результатов теста по требованию типа DNS.

Что касается оставшихся параметров, то из них интерес для DNS-тестирования представляет процент перекрашенного трафика (см. [рисунок 17](#), мы специально поместили трафик RT-классом!). В данном случае это корректное поведение, так как закупка на этой тушинской площадке предполагает только сервис самого низкого уровня (BE или «общий канал»), поэтому оператор связи абсолютно законно перекрашивает входящий с площадки трафик даже на собственный DNS-сервер. Процент же пакетов переданных не в том порядке представляет скорее спортивный интерес, так как штатный DNS-сервер любого типа обычно работает в многопоточном, либо многопроцессном (не путать с многопроцессорным!) режиме (подробности в классической сетевой литературе APUE или UNP). И каждый запрос обрабатывается отдельно. В связи с чем порядок следования ответов может быть произвольным.

Теперь запустим основной тест U1. Результаты запуска видны на [рисунках 27 и 28](#).

Test ID	Test name *	Class (IP Precedence or DSCP)	Service	Provider	SRC agent *	DST agent *	Source IP	DST agent IP *	Test frequency (sec)	Number of probes	DST agent type	Test type	Enabled	Packet size (B)	On demand test	Bandwidth for UDP tests (Kb/s)	Reverse bandwidth for UDP tests (Kb/s)	Creation date	Modify date
29374	plug-tushino_IQMM_U1_od	RT	Internet	Default	plug-tushino	IQMM	NAT	212	600	2000	A	U1	1	256	1	128	1024	2024-05-13 12:35:46.062607	2024-05-13 12:35:46.062607

run results cancel

Test plug-tushino_IQMM_U1_od started

Рисунок 27. Нажата кнопка запуска теста по требованию U1-типа. Виден «бегунок», отображающий примерный объём исполнения теста.

Бегунок для этого теста движется медленнее, так как число пакетов больше. Мы специально приводим два рисунка, чтобы это было наглядно видно.

SID	DID	TestType	TID	TStart	TEnd	ServiceCode	ProtoServiceCode	SDBytes	DSBytes	SDLost	SDLostPercent	DSLost	DSLostPercent	SDBW	DSBW	SDBWPercent	DSBWPercent	MinRtt
plug-tushino	akado-dns1	DNS	plug-tushino_akado-dns1_DNS_od	2024-05-14 11:25:13.745281	2024-05-14 11:25:16.998107	4	0	880	0	10	50.0000000000	0	0.0000000000	37312	0	44.999758792	0.0000000000	11.258116000

1 rows

Test ID	Test name *	Class (IP Precedence or DSCP)	Service	Provider	SRC agent *	DST agent *	Source IP	DST agent IP *	Test frequency (sec)	Number of probes	DST agent type	Test type	Enabled	Packet size (B)	On demand test	Bandwidth for UDP tests (Kb/s)	Reverse bandwidth for UDP tests (Kb/s)	Creation date	Modify date
29374	plug-tushino_IQMM_U1_od	RT	Internet	Default	plug-tushino	IQMM	NAT	212	600	2000	A	U1	1	256	1	128	1024	2024-05-13 12:35:46.062607	2024-05-13 12:35:46.062607

run results cancel

Test plug-tushino_IQMM_U1_od started

Рисунок 28. Бегунок продолжает движение вправо.

По окончании теста (точнее, по завершении движения бегунка) запрашиваем результаты U1-тестирования кнопкой Results. Результаты видны на [рисунках 29, 30, 31, 32, 33, 34](#).

SID	DID	TestType	TID	TStart	TEnd	ServiceCode	ProtoServiceCode	SDBytes	DSBytes	SDLost	SDLostPercent	DSLost	DSLostPercent	SDBW	DSBW	SDBWPercent	DSBWPercent
plug-tushino	IQMM	U1	plug-tushino_IQMM_U1_od	2024-05-14 11:30:15.178200	2024-05-14 11:30:53.438882	0	0	567148	568000	3	0.150000000	0	0.000000000	130887	1066095	99.858856201	101.670742035

1 rows

Copyright (C) 2008-2024, NetProbe, LLC. All rights reserved.

Рисунок 29. Начало результатов теста по требованию типа U1.

Тестирование заняло 38 секунд, число потерь от инициатора к сопряжённому составил 0,15%, в обратную сторону 0%. Скорости близки к запрошенным (99.85% и 101.67%). Увеличение скорости скорее всего связано с буферизацией трафика на промежуточных хостах, точно установить это место затруднительно.

MinRtt	AvgRtt	MaxRtt	SDMinDelay	SDAvgDelay	SDRMSDelay	SDMaxDelay	DSMinDelay	DSAvgDelay	DSRMSDelay	DSMaxDelay
12.661602000	15.219564000	26.021435000	7.541329000	9.732050000	0.000000000	16.760251000	5.120273000	5.487514000	0.000000000	9.261184000

Рисунок 30. Результаты задержек для теста по требованию типа U1.

RTT – от 12 миллисекунд до 26, среднее 15. Односторонние задержки в одну сторону от 7 до 16, в обратную от 5 до 9 миллисекунд. Среднее квадратичное предполагается к расчёту, но пока на этом протоколе (равно как и на U0) не рассчитывается. Поскольку этот параметр факультативен (нет стандартов, где он бы минимально регламентировался), результатами можно пользоваться.

plug-tushino_IQMM_U1_od (2024-05-14 11:33:43.646460)

SDJitter	DSJitter	SDMinIPDV	SDAvgIPDV	SDRMSIPDV	SDMaxIPDV	DSMinIPDV	DSAvgIPDV	DSRMSIPDV	DSMaxIPDV
0.615529000	0.113800000	0.100633000	2.191818000	2.462015000	9.218922000	0.004647000	0.367425000	0.545187000	4.140911000

Рисунок 31. Дрожания до RFC-3550 и Y.1540.

Дрожание по RFC-3550 от 0.6 до 0.1. По Y.1540 средние 2.1 и 0.3 миллисекунды, всплесков мало, так как среднее квадратичное близко к среднему арифметическому. В принципе, разницы от инициатора к сопряжённому и обратно могут быть связаны с тем, что у пробника plug-tushino аппаратная платформа на простом ARM, а у IQMM – на виртуальной машине. Минимальные значения по Y.1540 практического интереса не представляют ввиду использования в стандарте как базиса минимальной задержки.

SDMinMAPDV2	SDAvgMAPDV2	SDRMSMAPDV2	SDMaxMAPDV2	DSMinMAPDV2	DSAvgMAPDV2	DSRMSMAPDV2	DSMaxMAPDV2
0.000968000	0.905359000	0.932425000	2.292948000	0.058412000	0.159621000	0.244982000	1.713336000

Рисунок 32. Дрожания по G.1020.

Дрожания по G.1020 показывают аналогичное поведение. Выбор правильного алгоритма мы оставляем пользователю. В стандартах нет точного указания, по какому алгоритму следует рассчитывать дрожание.









 SDRemarked	 SDRemarkedPercent	 DSRemarked	 DSRemarkedPercent	 SDOOS	 SDOOSPercent	 DSOOS	 DSOOSPercent
1997	100.000000000	0	0.000000000	0	0.000000000	0	0.000000000

Рисунок 33. Перекраска трафика и изменение порядка.

Мы помним, что в настройках был указан RT-класс трафика. Видно, что оператор связи изменил класс для трафика для направления от plug-tushino до IQMM, но при этом не изменил тот же класс для трафика от IQMM до plug-tushino. Можно сделать вывод, что либо у оператора есть договорённости с точкой размещения IQMM, либо, что более вероятно, что на сети принимаемый от апплинков либо пиринговых точек трафик позволяет к пропуску по сети, используя раскраску, включая Real time. Это на наш взгляд, явная ошибка в конфигурации, возможно, обусловленная тем, что за апплинки отвечает обычно производительное оборудование и его настройщики могут образовать отдельный отдел в компании, который слабо связан с иными отделами.

Что же касается порядка следования пакетов, то здесь всё в порядке — проблем нет.









 SDMinTTL	 SDMaxTTL	 DSMinTTL	 DSMaxTTL	 SDLossBW	 SDLossBWPercent	 DSLossBW	 DSLossBWPercent
51	51	248	248	185	0.141143799	0	0.000000000

Рисунок 34. TTL и «потерянная» скорость.

Параметры минимального и максимального TTL нужны для расчёта т. н. «сдвига пути» (в текущем агенте в тестировании, поэтому не представлен), здесь они скорее для демонстрации возможностей, чем для реального использования.

«Потерянная» скорость — это обратная к скорости (BW) величина, рассчитываемая по принципу «чем меньше, тем лучше», в отличие от оригинала. Следует ли её использовать — решать пользователю системы.

Итак, мы видим, что с помощью технологии ССС-демона и агента с поддержкой модуля ССС и протокола U1, легко создавать выездное тестирование у клиента, равно как и постоянную установку на площадках, закрытых технологией трансляции адресов (NAT).

Содержание

1 Введение.....	2
2 Схемы включения IQM-агентов.....	2
3 Протокол U1.....	8
4 Схема включения ССС-демона.....	9
5 Ключи управления ССС-демона.....	12
6 Примеры запуска тестов из-за NAT.....	15

Настоящим подтверждается, что все исключительные авторские права на данную документацию принадлежат ООО «НетПроб». Предоставление прав на данную документацию осуществляется по лицензионному договору присоединения, ссылки на юридический текст которого указаны в данном тексте. Неотчуждаемые личные неимущественные права на данную документацию принадлежат физическим лицам – авторам, перечисленным в документации. Настоящим подтверждается, что все права на использованные системные и стандартные модули программного обеспечения принадлежат их авторам и используются правомерно в соответствии с предоставленными авторами лицензионными договорами, в том числе, но не ограничиваясь, GNU General Public License, Artistic License и т.д.

Copyright © 2008-2024



ООО «Нетпроб»

Copyright © 2010-2024



Сергей Александрович Еременко