Clientgen — это инструмент с открытым исходным кодом для генерации трафика PTP-клиентов, который использует библиотеку PF\_RING и пример simpleclient. Он имитирует большое количество PTP-клиентов, взаимодействующих с PTP-сервером (например, Grandmaster). Вот ключевые аспекты:

**Основные функции**

1. Запросы грантов  
   Клиенты запрашивают у сервера три типа унидирекциональных грантов (разрешений) на определённый срок:
   * Sync (синхронизация времени)
   * Announce (объявление параметров)
   * Delay Response (ответ на запрос задержки).
2. Автоматическая повторная отправка  
   Если сервер не отвечает на запрос гранта, клиент повторяет его через заданный интервал (**ClientRetranTimeWhenNoResponseSec**).
3. Перезапуск клиентов  
   После истечения всех грантов клиент может автоматически перезапуститься через время, указанное в **TimeAfterDurationBeforeRestartSec**.
4. Периодические запросы  
   Клиенты отправляют запросы DelayReq с частотой, заданной в **TimeBetweenDelayReqSec**.

**Конфигурация**

Настройка выполняется через JSON-файл (**clientgen\_config.json**), где указываются:

* Сетевые параметры :
  + Интерфейс (**Iface**, например, **ens1f0**).
  + MAC-адрес сервера (**ServerMAC**).
  + IP-адрес сервера (**ServerAddress**).
  + Диапазон IP-адресов клиентов (**ClientIPStart**, **ClientIPEnd**, **ClientIPStep**).
* Поведение клиентов :
  + Продолжительность грантов (**DurationSec**).
  + Лимит скорости запуска клиентов (**SoftStartRate**).
  + Таймаут работы утилиты (**TimeoutSec**).
* Производительность :
  + Количество потоков для обработки TX/RX (**NumTXWorkers**, **NumRXWorkers**).
  + Настройки для обработки временных меток (timestamping).

**Обработка пакетов**

* Входящие пакеты :
  + Читаются через PF\_RING (эффективное распределение пакетов между потоками).
  + Парсятся с помощью библиотеки **gopacket**.
  + Обрабатываются в зависимости от типа (ARP, ICMPv6, UDP).
* Исходящие пакеты:
  + Отправляются через сокеты с поддержкой временных меток.
  + Для каждого отправленного пакета фиксируется TX-метка (время отправки).

**Архитектура**

* Рабочие потоки (goroutines) :
  + **RX Workers** — чтение входящих пакетов.
  + **TX Workers** — отправка пакетов.
  + **Packet Processors** — анализ и генерация ответов.
  + **Retransmit Workers** — управление повторными запросами.
  + **Restart Workers** — перезапуск клиентов.
* Оптимизация :  
  PF\_RING позволяет обрабатывать пакеты на уровне CPU, минуя ограничения сетевого адаптера.

**Пример вывода**

При запуске утилита выводит статистику:

* Количество активных клиентов (**TotalClients**).
* Задержки ответов сервера (например, **Announce Grant Latency**).
* Счётчики отправленных/полученных пакетов.
* Производительность потоков (загрузка CPU).

**Установка**

1. Установите PF\_RING:

bash

Копировать

1

2

3

4

git clone https://github.com/ntop/PF\_RING.git

cd PF\_RING/kernel

make && sudo make install

insmod ./pf\_ring.ko

1. Соберите Clientgen:

bash

Копировать

1

2

3

4

git clone https://github.com/opencomputeproject/Time-Appliance-Project

cd Software/Experimental/clientgen/

export LD\_LIBRARY\_PATH=/usr/local/lib/

go build

**Использование**

Запуск:

bash

Копировать

1

./clientgen -config clientgen\_config.json

Clientgen полезен для нагрузочного тестирования PTP-серверов, имитируя тысячи клиентов с контролем временных меток и автоматической обработкой ошибок.