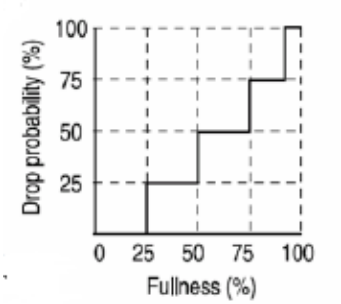
Визуализация DROP профилей трафика на Juniper MX

Наиболее наглядным отображением влияния механизмов COS на трафик разных классов и разметок является пропуск нескольких потоков трафика через интерфейсы ограниченной пропускной способности и построение графиков нагрузки от времени. График динамического процесса может дать больше понимания, чем страница текста, описывающего этот процесс.

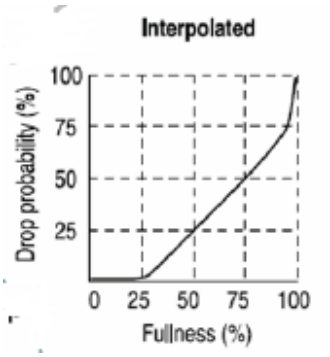
В теме COS есть один вопрос, который сложнее поддаётся визуализации – это drop-профили. Задача drop-профилей – не допустить переполнения буфера очереди, отбрасывая пакеты с вероятностью, зависящей от наполненности этого буфера. Долгое хранение пакетов в полностью наполненном буфере очереди может быть нежелательным для real-time траффика, а переполнение буфера приводит к tail-drop, что вызывает синхронизацию TCP-сессий трафика и пилообразную нагрузку интерфейсов.

Drop профили могут быть сегментные и интерполированные.



Сегментные задают «лесенку» и вероятность отбрасывания пакетов меняется скачкообразно.

```
drop-profiles {
    segmented {
        fill-level 25 drop-probability 25;
        fill-level 50 drop-probability 50;
        fill-level 75 drop-probability 75;
        fill-level 95 drop-probability 100;
    }
}
```



Интерполированные создают 64 точки между значениями (0,0) и (100, 100), включая настроенные точки.

```
drop-profiles {
    drop-inter-5-75 {
        interpolate {
            fill-level [ 25 95 ];
            drop-probability [ 5 75 ];
        }
    }
}
```

Картинка «в представлении художника» - это одно, а реальные замеры – совсем другое.

Непосредственное влияние drop-профиля на трафик можно увидеть на графике лабы №1.

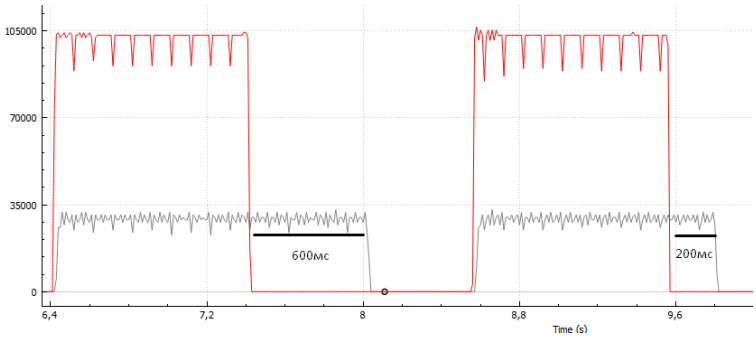
Лаба №1 представляет собой импульсы высотой примерно 10 Мбит/сек и длительностью 1 секунда. Интервал между импульсами - тоже 1 секунда. Красная линия – это трафик с выхода генератора, черная линия – трафик после прохождения маршрутизатора. Первый импульс обрабатывался следующей конфигурацией.

```
set schedulers sch-be transmit-rate remainder
set schedulers sch-bc transmit-rate percent 20
set schedulers sch-bc transmit-rate rate-limit
set schedulers sch-bc buffer-size percent 50
```

Виден «хвост» трафика на протяжении порядка 600мс. В момент прохождения хвоста генератор уже не шлёт трафик и мы видим пакеты, хранившиеся в буфере.

Второй импульс обрабатывается конфигурацией с агрессивным drop-профилем, отбрасывающим пакеты при не полностью заполненном буфере. «Хвост» трафика значительно укорачивается.

```
set drop-profiles aggessive fill-level 30 drop-probability 90
set schedulers sch-bc drop-profile-map loss-priority any protocol any drop-profile aggressive
```



Влияние drop-профиля на трафик очевидно, но сам вид drop-профиля при этом скрыт от нас. Все пакеты лабы №1 одинаковы и безымянны и мы видим только их объём и время прибытия на приёмник. Определить вероятность отбрасывания конкретного отправленного пакета от наполненности буфера таким образом нельзя. Поэтому перейдём к лабе №2.

Лаба №2 представляет собой импульс высотой 80 Мбит/сек (скорость по IP вместе с заголовком) и длительностью 3 секунды. Каждую миллисекунду импульс передаёт 100 IP-пакетов размером 100 байт. Каждый пакет в поле данных несёт информацию о времени отправки. Скорость в шедулере меньше скорости для рабочего трафика лабы, что будет приводить к постепенному заполнению буфера и применению разных ступеней drop-профиля.

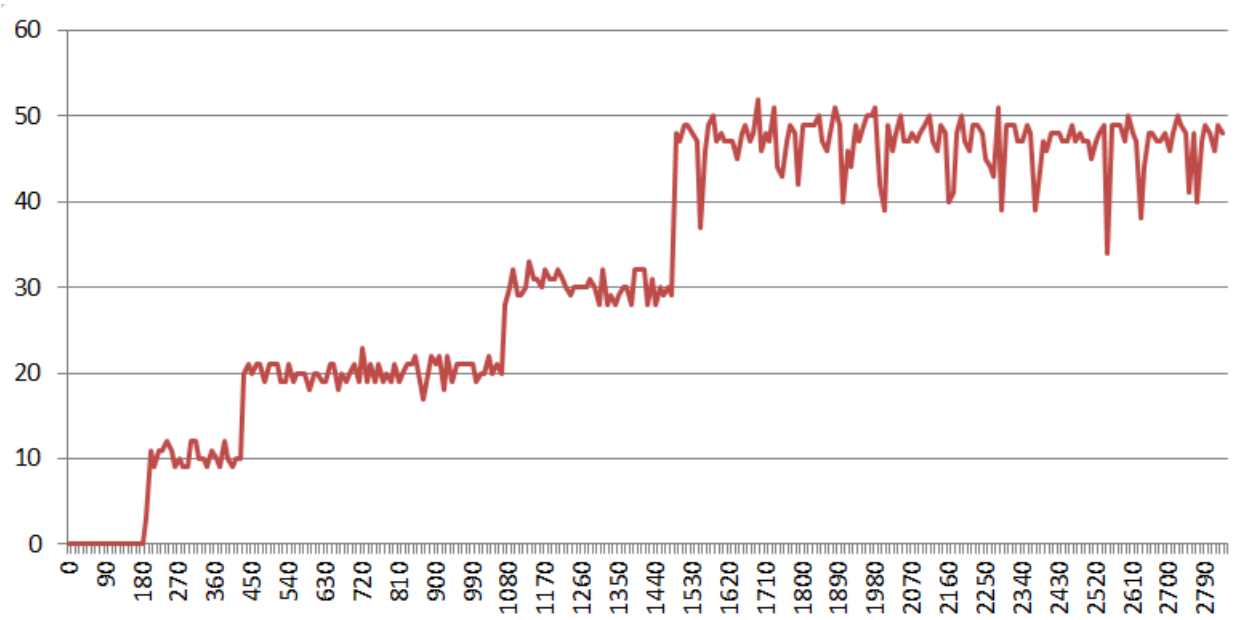
```
sch-be {
    transmit-rate {
        60m;
        exact;
    }
    buffer-size percent 40;
    drop-profile-map loss-priority any protocol any drop-profile drop-10-20-30;
}
```

После прохождения маршрутизатора трафик собирается, дамп конвертируется в txt формат и анализируется. Информация об интервале отправки извлекается из пакетов, интервалы отправки группируются по 10 мс.

Применяем сегментный drop-профиль следующего содержания.

```
drop-10-20-30 {
    fill-level 20 drop-probability 10;
    fill-level 40 drop-probability 20;
    fill-level 80 drop-probability 30;
}
```

По оси абсцисс время в миллисекундах, по оси ординат – вероятность потери пакета, посланного в данный интервал.



Следует учесть, что нам нельзя получить реальную наполненность буфера в момент прихода на маршрутизатор конкретных пакетов для построения полноценного графика drop-профиля. Остаётся только опираться на тот факт, что наполненность буфера будет расти, если входящий трафик превышает ограничение отправки. При включении очередной ступени drop-профиля приходящие пакеты начнут отбрасываться и скорость наполнения буфера снизится, что должно вызывать «растягивание» графика по оси абсцисс. При 100% наполненности буфера вероятность отбрасывания пакетов не может превышать отношение скоростей приходящего трафика и настроенной скорости отправки пакетов.

Точности графика мешает неравномерность отправки трафика программой tcpreplay в начальный период времени (Python + scapy даёт ещё большую неравномерность).

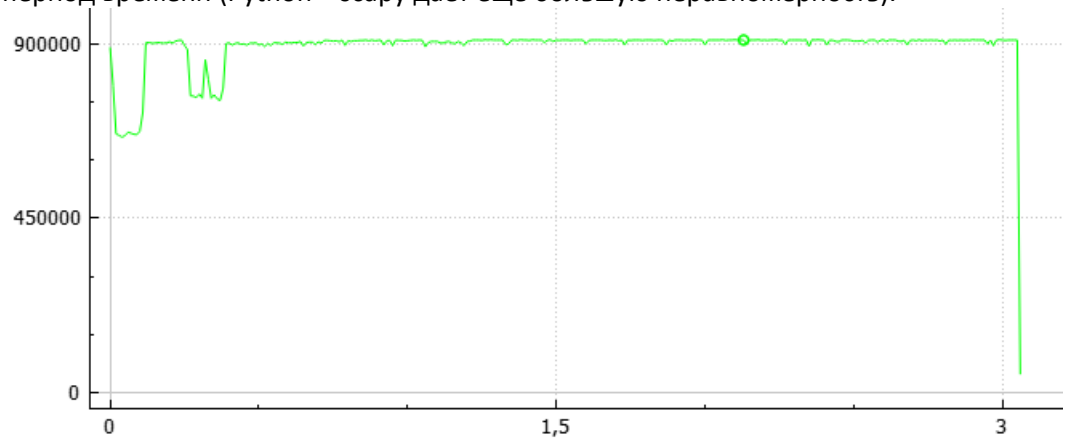
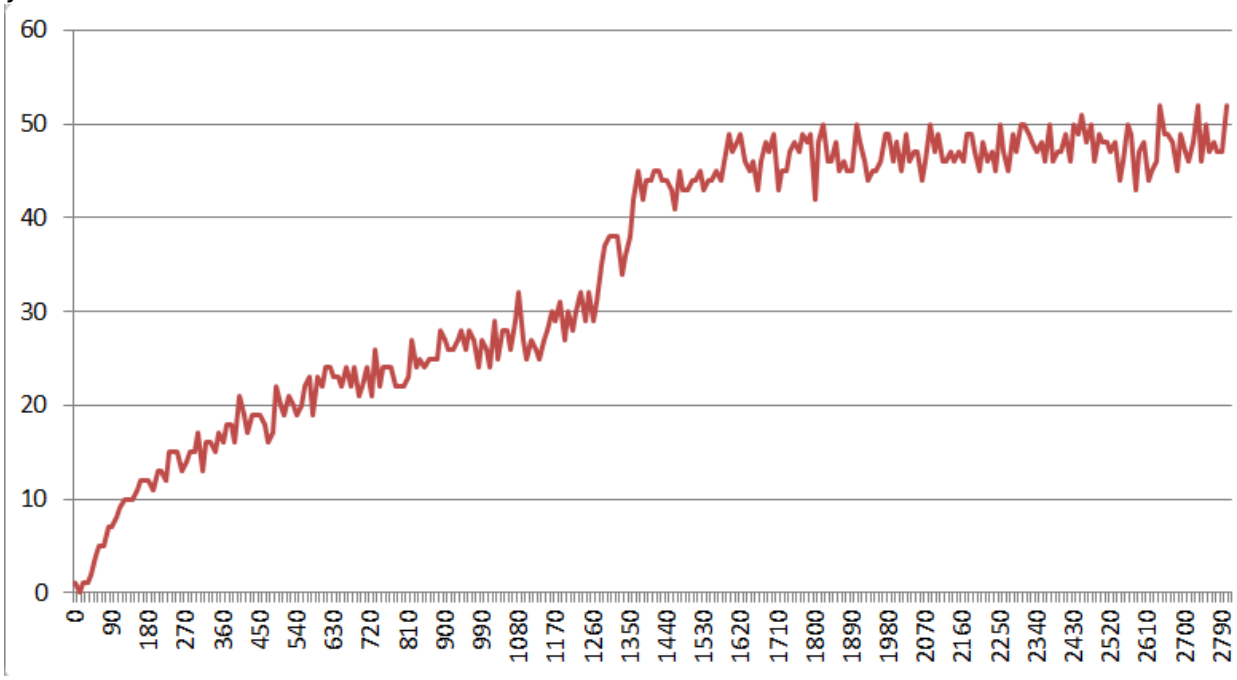


График отправки пакетов командой
tcpreplay -i ens256 -p 100000 --preload-pcap dump_80mbit_3sec_1ms_interval.pcapng

Лаба №3 повторяет №2. Применяем интерполированный drop-профиль.

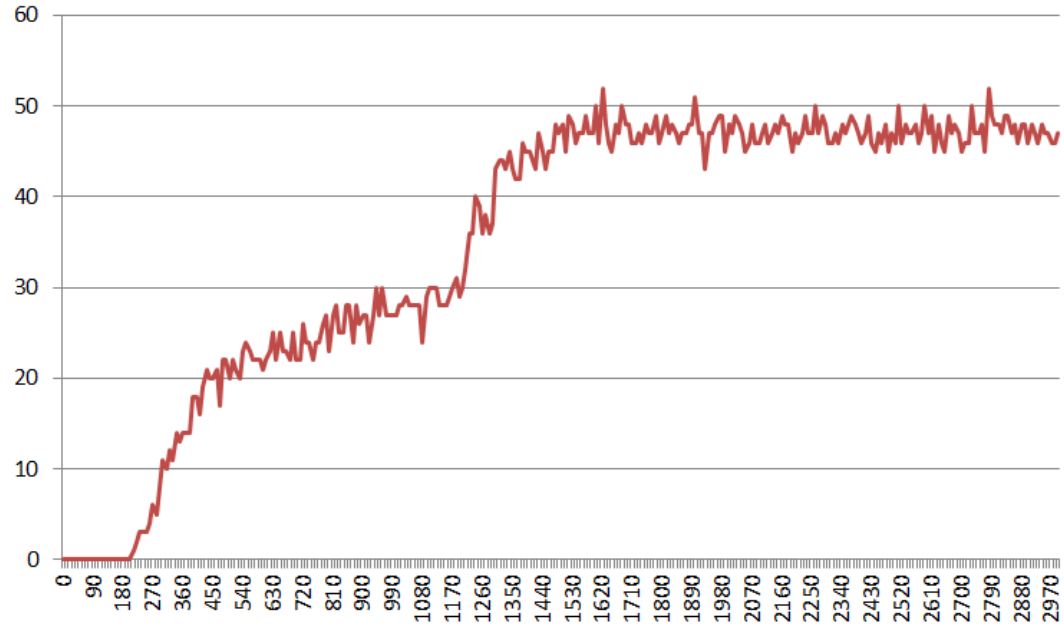
```
drop-inter-10-20-30 {
    interpolate {
        fill-level [ 20 40 80 ];
        drop-probability [ 10 20 30 ];
    }
}
```



Первая три точки интерполированного профиля (0-0, 20-10, 40-20) составляют прямую линию, но на графике вероятности отбрасывания пакетов мы видим ожидаемое «растягивание» графика по оси абсцисс. Пакеты начинают отбрасываться практически сразу, т.к. при 2% наполнении буфера такой drop-профиль должен отбрасывать 1% пакетов.

Лаба №4

```
drop-inter-1-20-30 {
    interpolate {
        fill-level [ 20 40 80 ];
        drop-probability [ 1 20 30 ];
    }
}
```



Лаба №5

Попробуем сделать пологий график.

```
drop-inter-1-5-10-30 {
    interpolate {
        fill-level [ 20 40 60 80 ];
        drop-probability [ 1 5 10 30 ];
    }
}
```

