1. Титульный слайд

Представиться и назвать тему.

2. Цели и задачи

Моя работа посвящена в основном реализации в ядре Linux дисциплины обслуживания Class-Based WFQ, известной в русской литературе как взвешенный алгоритм честного обслуживания на основе классов. Далее он будет адресоваться как CBWFQ.

Этот алгоритм был представлен компанией Cisco и сейчас часто используется на маршрутизаторах компании. Целью же работы служит определение алгоритма по имеющимся о нём данным и реализация его в ядре Linux – открытой системе, которая часто используется как бесплатное сетевого ПО.

Задачи, которые я перед собой поставила, представлены на слайде. Очень важным шагом к реализации алгоритма служит исследование уже существующих дисциплин обслуживания. Выполнение этой задачи позволило бы не только проанализировать слабые и сильные стороны существующих решений, но и определить, на какие детали стоит обращать внимание, какие подходы используются и как реализованы те или иные вещи при реализации дисциплины. Остальные задачи носят технический характер.

3. Дисциплины обслуживания

Основополагающая сущность работы – дисциплина обслуживания.

Дисциплина обслуживания – это ящик, принимающий на себя поток пакетов и выпускающий каким-либо образом обработанный поток. По сути, это набор механизмов, определяющий, каким образом будет происходить организация очереди пакетов.

Важно понимать, что мы можем управлять лишь скоростью передачи отправляемых данных, и дисциплины обслуживания – это механизм управления.

На слайде представлена схема, определяющая место дисциплин обслуживания очередей в пути движения пакетов в системе Linux. Очереди могут быть входящие и выходящие. Дисциплины обслуживания регулируют выходящие очереди пакетов, входящая же очередь обрабатывается дисциплиной ingress.

4. Priority Queueing

Приоритетные очереди – это одна из самых простых классовых дисциплин обслуживания, реализованных в Linux. Суть её алгоритма достаточно прост: трафик классифицируется по классам согласно карте приоритетов (или иными дополнительными способами), классы обслуживаются в циклическом порядке (round robin) от самого высокого приоритета до самого низкого приоритета, а внутри классы обслуживаются в порядке FIFO. Описанная схема отображена на слайде.

Преимущества, обозначенные на слайде, являются следствием простоты алгоритма. Как, впрочем, и недостатки.

Дисциплина используется, чтобы приоритизировать трафик без использования только TOS-флагов, но и с использованием гибкости фильтров, а также понизить время отклика, когда нет нужды замедлять трафик.

5. Class Based Queueing

CBQ – это алгоритм, реализующий механизм иерархического разделения канала. Это довольно интересный механизм, суть которого заключается распределении пропускной способности класса, который её не использует, между соседями в иерархии классов. Пример такой иерархии представлен на слайде.

Непосредственная планировка пакетов CBQ работает на взаимодействии планировщика разделения канала и так называемого общего планировщика, которым может быть алгоритм от простого циклического до взвешенного.

Как видно на слайде, CBQ представляет данные в древовидной структуре. В листьях дерева, обозначенных зелёным, непосредственно находятся очереди; узлы, обозначенные розовым, выполняют "организационную"работу (к примеру, содержат в себе фильтры) и не участвуют в хранении пакетов. Очереди в листьях по умолчанию используют FIFO, однако есть возможность настраивать другие дисциплины.

Помимо планировки, CBQ позволяет шейпинг трафика. Без описания деталей, CBQ не пропускает пакеты, если значение среднего времени простоя достигает некоторого заданного порога.

CBQ – это первая дисциплина, в которой реализован механизм разделения канала. Те, кто близко сталкивался с CBQ, утверждают, что она не всегда точна и иногда допускает грубые просчёты при измерении времени, что фатально, поскольку CBQ на основе измерения времени между запросами рассчитывает среднюю загруженность канала. Разработчики отмечают, что она слишком сложна и слабо оптимизирована для большинства типичных ситуация.

CBQ имеет много параметров, её сложно конфигурировать, однако её возможно настроить так, чтобы она показывала неплохие результаты.

6. Hierarchhy Token Bucket

Проблемы, которые есть в CBQ, призвана решить дисциплина HTB, которая, подобно CBQ, реализует механизм разделения канала, но исправляет многие ошибки первой. HTB призвана было более понятной и интуитивной заменой CBQ в Linux.

Как и CBQ, HTB использует древовидную структуру данных. На левом рисунке слайде изображена иерархия классов. Первое число в узлах обозначают идентификатор родителя, второе – собственный идентификатор узла. Идентификаторы используется для дополнительной настройки классов (к примеру, для присоединения фильтра к классу). При добавлении пакета в очередь HTB обходит дерево с корня в поисках класса, который должен получить данные. Внутри листьев очереди обслуживаются конфигурируемой ДО (по умолчанию опять же FIFO). Это схема отображена на правом изображении. В листьях можно наблюдать аббревиатуру TBF, что означает, как многие знают, Token Bucket Filter. HTB её напрямую не использует, но её концепции непосредственно вшиты в HTB, что делает из дисциплины неплохой механизм для шейпинга.

Несмотря на то, что дисциплина HTB призвана улучшить механизмы CBQ, она всё ещё имеет много опций (но меньше, чем у CBQ) и сложна в конфигурации. Более того, она медленнее CBQ, что, однако, может компенсироваться простой вычислений.

7. HFSC (Эйч Эф Эс Си (:c))

Алгоритм иерархических честных кривых обслуживания – это дисциплина, основанная на математической модели честных кривых обслуживания. Кривая обслуживания – это

неубывающая функция от времени, которая служит нижней границей количества обслуживания, предоставляемого системой.

Алгоритм ставит перед собой цели: гарантировать точное выделение ПС для всех листовых классов (критерий реального времени), честно выделять избыточную пропускную способность соответственно классовой иерархии и минимизировать разницу между реальной и идеальной кривых обслуживания.

TODO: Придумать, что сказать про convex и concave кривые. На слайде представлен пример линейных кривых, определяемых параметрыми наклона m1 и m2 и точкой пересечения прямых. Кривые могут быть выпуклыми и вогнутыми. Выпуклые кривые служат результатом низкой средней и худшей задержек. TODO: Как используются вогнутые кривые?

Благодаря тому, что HFSC основан на математической модели с доказанными нижними границами, алгоритм даёт результаты лучше, чем аналогичные по классу алгоритмы CBQ и HTB.

8. Weighted Fair Queueing

WFQ – это механизм планирования пакетных потоков данных с различными приоритетами. Он основан на математической модели идеального планировщика, которая позволяет максимально точно разделить пропускную способность между классами трафика в соответствии с весами. В силу того, что GPS предполагает, что за цикл обслуживается бесконечно малая часть пакета, в реализации минимальной единицей обслуживания был выбран пакет.

Общий алгоритм WFQ довольно прост. Планировщик обслуживает N потоков, каждому из которых назначается вес; на основе всех весов и пропускной способности канала вычисляется полоса для потока. Каждый пакет имеет виртуальное время отправления (вычисляемое в предположении, что планировщик идеальный); пакет с наименьшим временем выбирается для отправки.

9. WFQ Drop Policy

WFQ использует механизмы раннего отбрасывания пакетов: агрессивное отбрасывание и раннее отбрасывание. Общий алгоритм приведён на слайде. Раннее отбрасывание пакетов срабатывает, когда достигается порог количества пакетов перед отбрасыванием из самой длиной очереди (обозначен как CDT на слайде). Агрессивное отбрасывание начинается, когда достигается максимальное количество пакетов, которое может находиться во всех очередях (обозначен как HQO).

10. Flow-based weighted fair queueing

Есть много вариаций алгоритма WFQ, один из которых – WFQ на основе потоков, который часто непосредственно называют WFQ.

Данный алгоритм реализует всё то, что было сказано ранее о WFQ, за тем исключением, что для увеличения производительности вместо времени отравления используется размер пакета, так как он пропорционален времени, а веса вычисляются на основе IP-приоритета.

WFQ не допускает пользовательскую конфигурацию классов, что, как мы увидели ранее, часто используется классовых дисциплинах, чем по сути является WFQ. Более

того, алгоритм не предоставляет задание фиксированной пропускной способности, что так же важно.

11. Class-based Weighted Fair Queueing

Эти недостатки призвана исправить алгоритм WFQ на основе классов, представляющий гибкую конфигурацию классов и фиксированную пропускную способность.

CBWFQ предоставляет возможность задать классы трафика на основе критериев соответствия (условных фильтров). Пакеты, удовлетворяющие критериям, составляют трафик класса. После определения класса ему назначаются характеристики, которые определяют политику очереди: пропускная способность, максимальная длина очереди и так далее. CBWFQ позволяет явно указать требуемую минимальную полосу пропускания для каждого трафика. Пакеты, которые не попали ни в один класс, отправляются в класс по умолчанию. Очереди в классах управляются алгоритмами FIFO или WFQ. В качестве политики отбрасывания CBWFQ позволяет использовать как отбрасывание конца очереди, так и алгоритм взвешенного раннего обнаружения, что конфигурируется для каждого класса.

Каким именно образом организовано хранение данных внутри алгоритма, какой непосредственно алгоритм выборке – всё это скрыто внутри. Реализация CBWFQ, в отличие от почти всех описанных выше дисциплин, закрытая, что задаёт задачу воссоздания алгоритма.

12. Сравнительная таблица

Для подведения итогов по обзору дисциплин обслуживания на слайде приведена сравнительная таблица, демонстрирующая особенности каждой по типам характеристик.

Особыми важными здесь я считаю метод планирования и честность. Честность – это довольно спорное понятие, по которому нет какого-то общего соглашения. В случае алгоритмов WFQ семейства честность означает равномерное распределение ресурсов, пропорциональную весу. В случае HFSC честность означает распределение избыточных ресурсов в соответствии с обозначенными политиками и отсутствия наказания для сессий, которые использовали слишком много ресурсов.

Несмотря на то, что понятие честности слабо определено, многие исследователи и сетевые инженеры соглашаются, что наличие этой характеристики довольно важно для планировщиков, поскольку этот механизм предоставляет возможность принимать решения на основе более серьёзных и точных характеристик, чем, к примеру, взвешенное циклическое обслуживание, которое может не учитывать некоторые аспекты.

TODO: Разумно ли то, что я здесь говорю?

13. Тестирование модуля

Для тестирования модуля CBWFQ была собрана система их трёх виртуальных машин, представленная на слайде: источник трафика, узел с CBWFQ, через который проходит трафик, и целевой узел, на который идёт трафик. Для возможности взаимодействия с дисциплиной из пользовательского пространства был разработан модуль для системы управления трафиком, выраженной в утилите tc. Настройки CBWFQ можно видеть на слайде: на узле выделяется два класса с разными выделенными пропускными способностями, каждый класс имеет фильтр, который определяет, какие пакеты попадут в класс.

Измерение проводилось с помощью утилиты ipref, которая позволяет измерять количество пропускной способности, который получил сгенерированный утилитой поток.

14. Тестирование модуля

На слайде можно наблюдать график зависимости пропускной способности, выделенной классам, от времени.

Важно помнить, что CBWFQ всегда имеет класс по умолчанию, в который попадает неклассифицированный трафик, который на графике представлен жёлтой кривой.

Из графика видно, что два класса трафика, обозначенные синим и красным цветом, получают примерно количество пропускной способности, пропорциональное сконфигурированному; жёлтый – оставшуюся ПС. Так же на графике видны сильные отклонения от предполагаемых значений. Это отражает состояние модуля: у него страдает точность. Он в данный момент находится на стадии отладки и тестирования.

15. Вывод

TODO: Что можно сказать в завершении?