Реализация управления перегрузкой на практике

(базовые алгоритмы)

Кафедра АСВК

Три основных вопроса

1. *Когда следует посылать новые данные?*
2. *Когда следует посылать данные повторно?*
3. *Когда надо отправлять подтверждения?*

TCP Pre-Tahoe

Кафедра АСВК

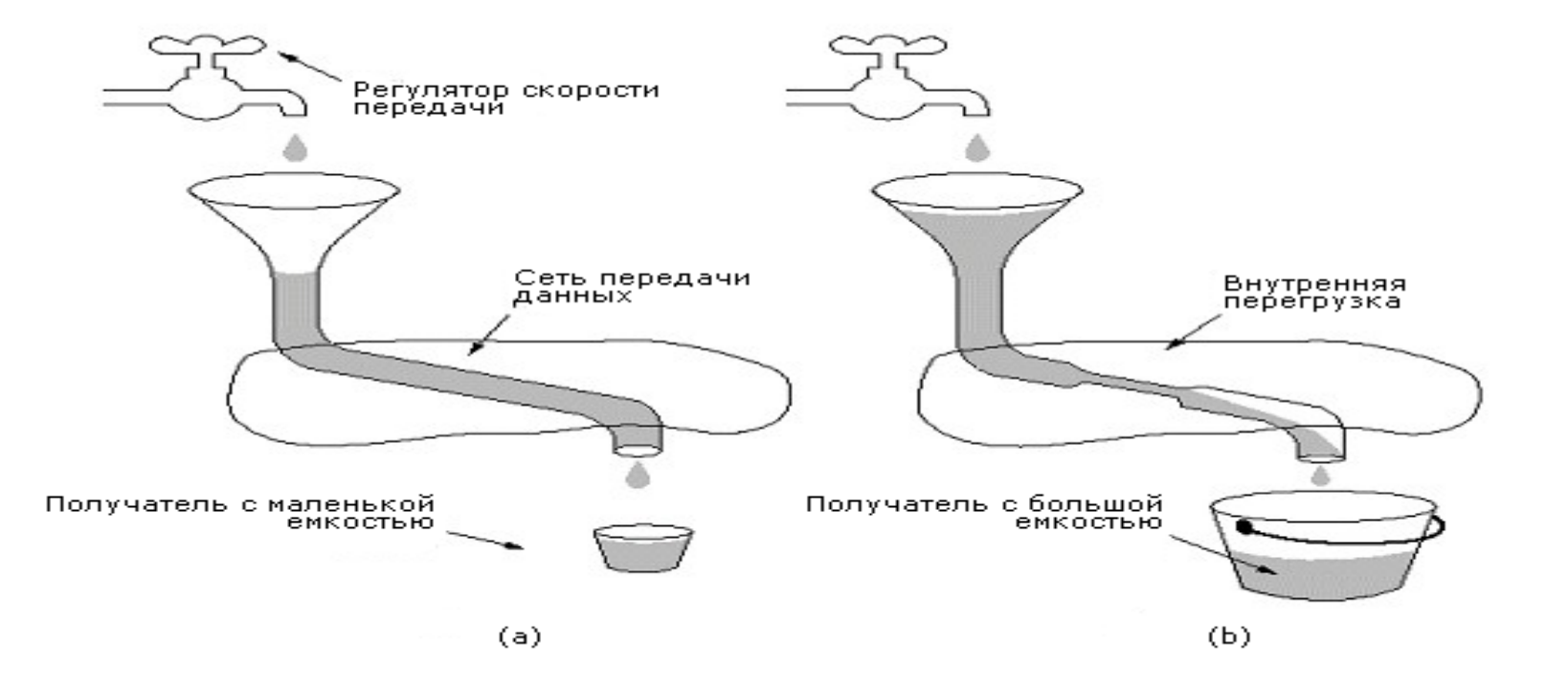
Немного истории

- *1983 – ARPA переходит на TCP/IP*
- *1986 - Интернет страдает от перегрузок*
- *1987 - Ван Якобсон предлагает TCP Tahoe*
- *1990 – Добавляются режимы быстрого восстановления и быстрой повторной передачи (Reno)*

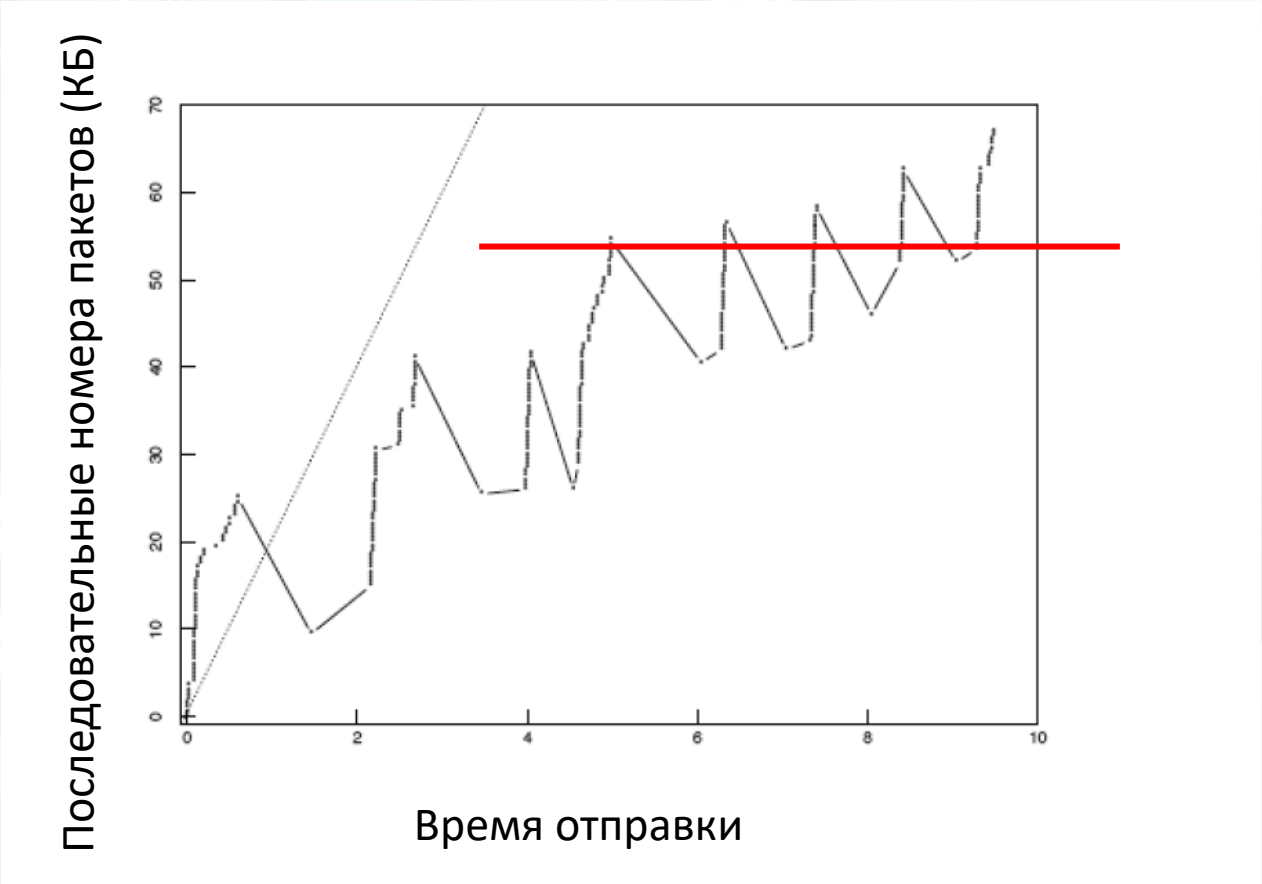
TCP Pre-Tahoe

- *Получатель устанавливает размер окна управления потоком (размер скользящего окна)*
- *Отправитель шлет пакеты, число которых полностью соответствует размеру скользящего окна*
- *На каждый пакет устанавливается таймер*
- *Проблема: что будет если размер окна превышает пропускную способность сети?*

Управление потоком и управление перегрузкой



TCP образца 1986



Переотправка одного пакета 4 раза!

Рисунок из статья ван Якобсона и Карела

TCP Tahoe

Кафедра АСВК

Три усовершенствования TCP

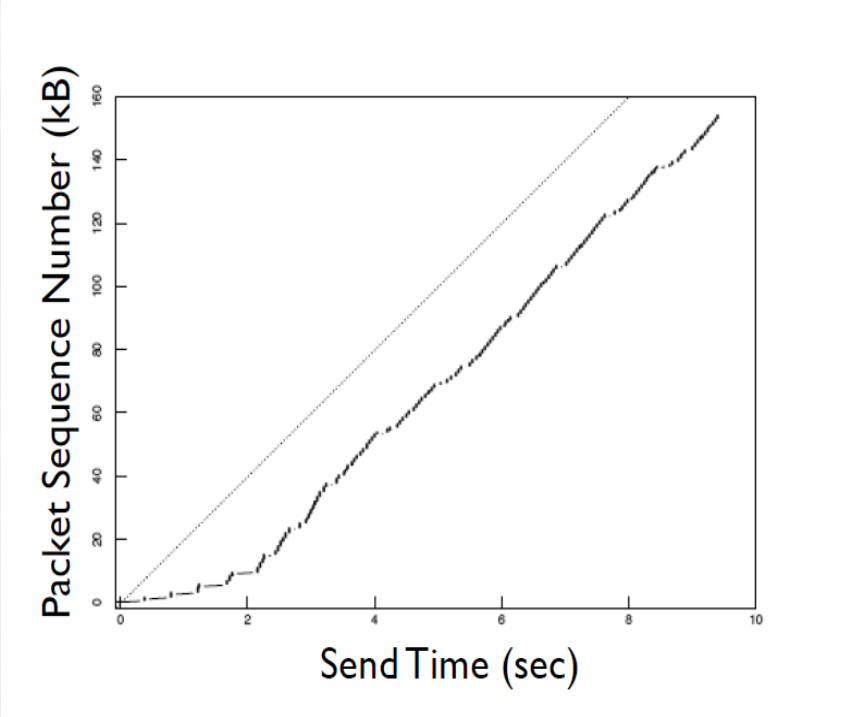
- *Окно перегрузки (CWND)*
- *Оценка Time_out*
- *Автосинхронизация (Self-clocking)*

Окно перегрузки (TCP Tahoe)

- *Отправитель*
  - *узнает от получателя FCWND (Flow Control WND)*
  - *оценивает размер CWND*
- *Окно отправителя = min (FCWND,CWND)*
- *Разделение фазы управления перегрузкой на две*
  - *Медленный старт*
  - *Предотвращение перегрузки - стабилизация*

Медленный старт

- *Медленный старт*
  - *CWND = MSS*
  - *На каждый ACK увеличиваем окно на MSS*
- *Экспоненциально увеличиваем (удваиваем) размер окна перегрузки, прощупывая возможность сети*
- *«медленный» по сравнению с изначальным алгоритмом*

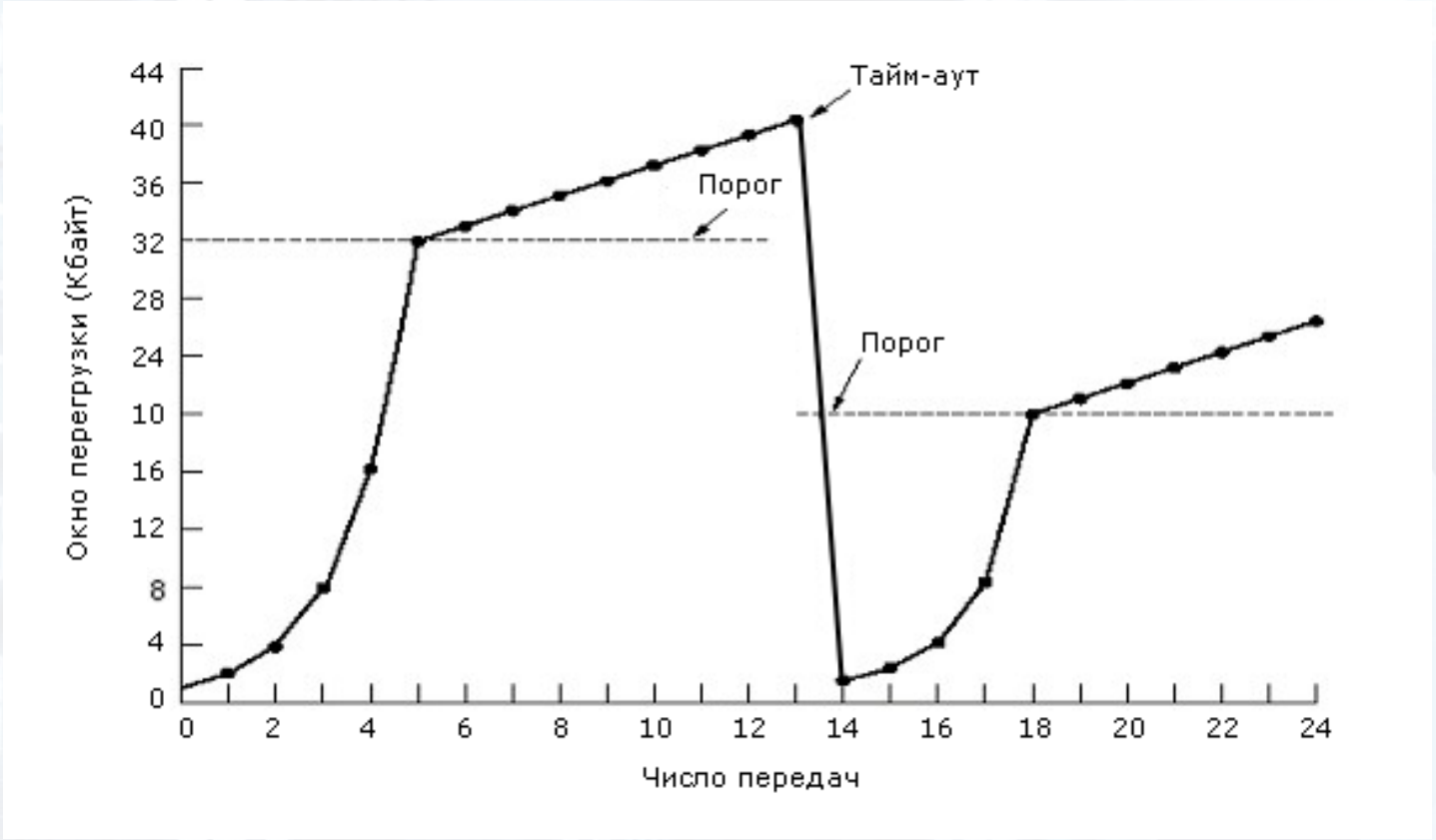


Предотвращение перегрузки

- *Медленный старт (slow start)*
  - *Увеличиваем CWND на MSS на каждое подтверждение*
  - *Экспоненциальный рост (за RTT удваиваем cwnd) пока не достигнем порога (cwnd/2 в предыдущей фазе предотвращения перегрузки) или не обнаружим перегрузку*

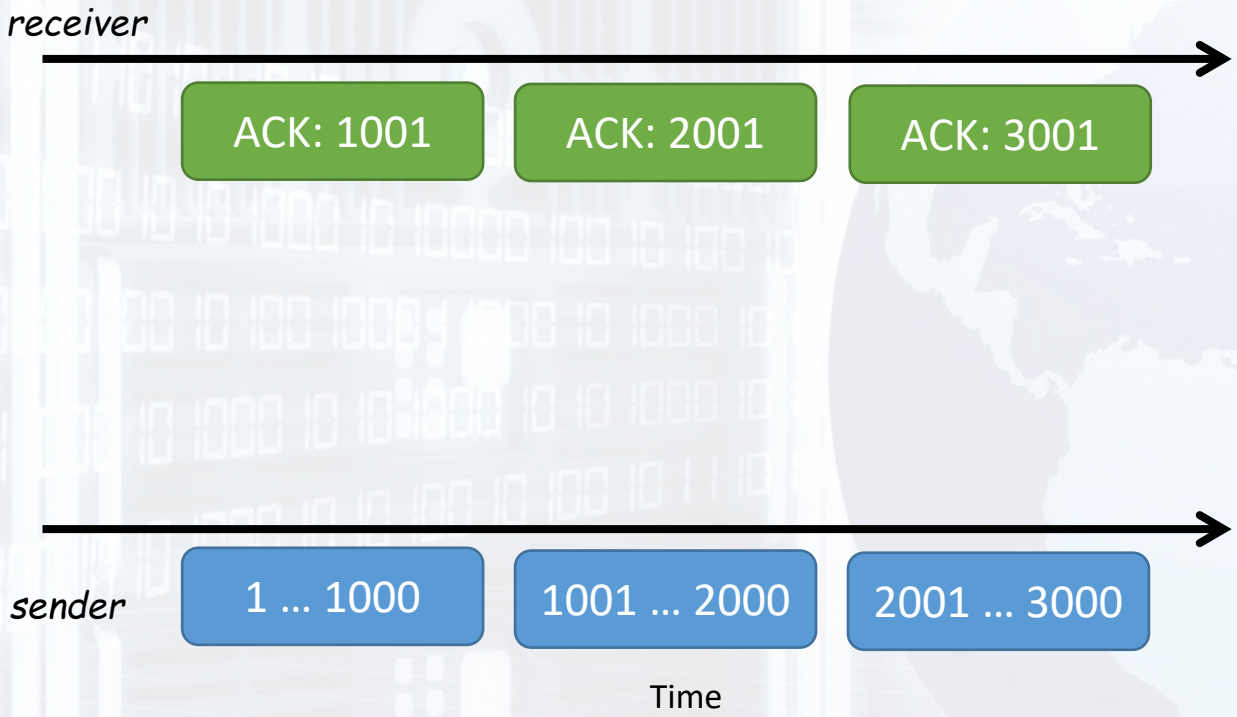
- *Предотвращение перегрузки (congestion avoidance)*
  - *Увеличиваем окно перегрузки только на MSS^2/CWND при каждом подтверждении*
  - *За каждый RTT (cwnd/MSS) увеличиваем cwnd на MSS*
  - *Линейный рост*

Пример управления перегрузками



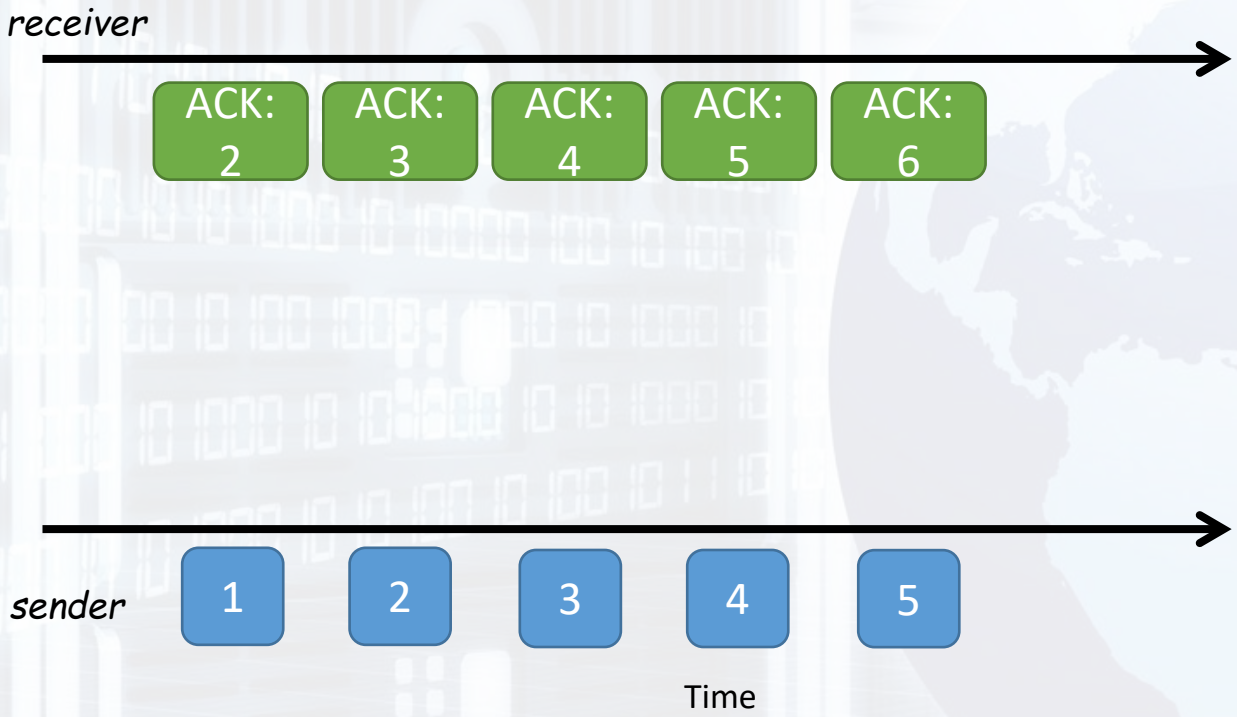
Механизм подтверждений

- *В подтверждении содержится номер следующего ожидаемого байта*



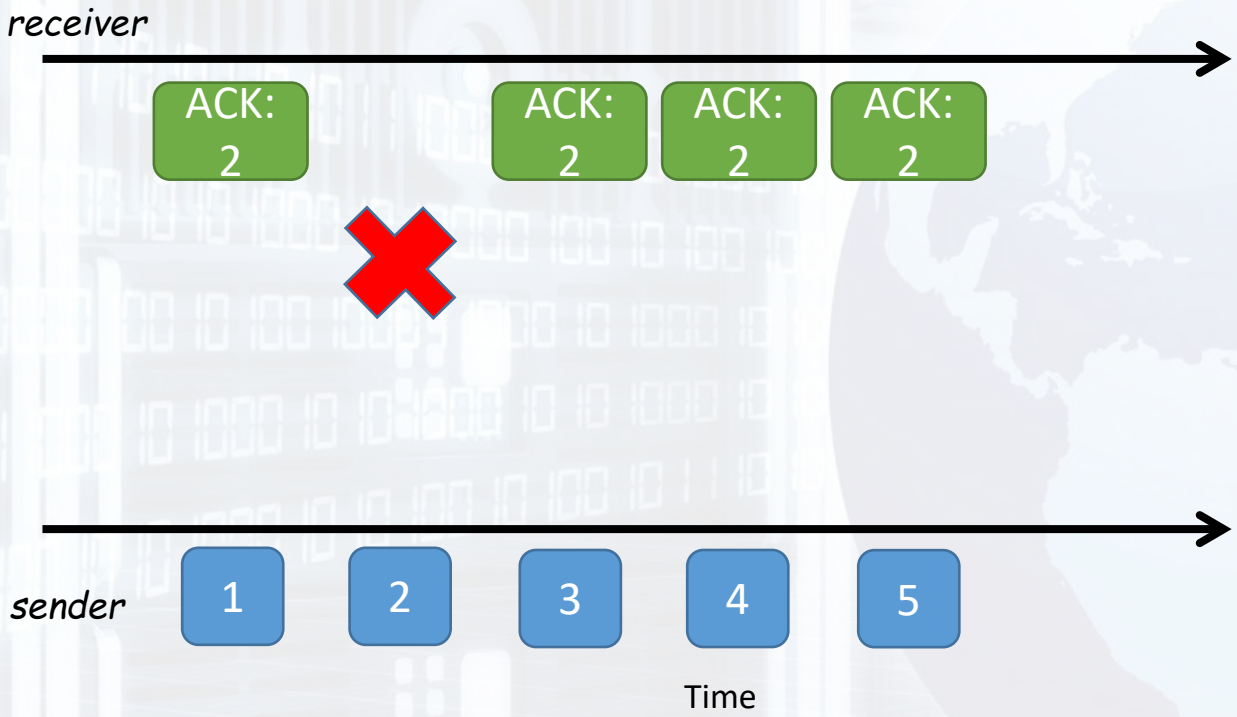
Механизм подтверждений

- *Для упрощения будем говорить не о байтах, а о пакетах.*



Механизм подтверждений

- *При потере пакета появляются повторные подтверждения (duplicate ACK)*



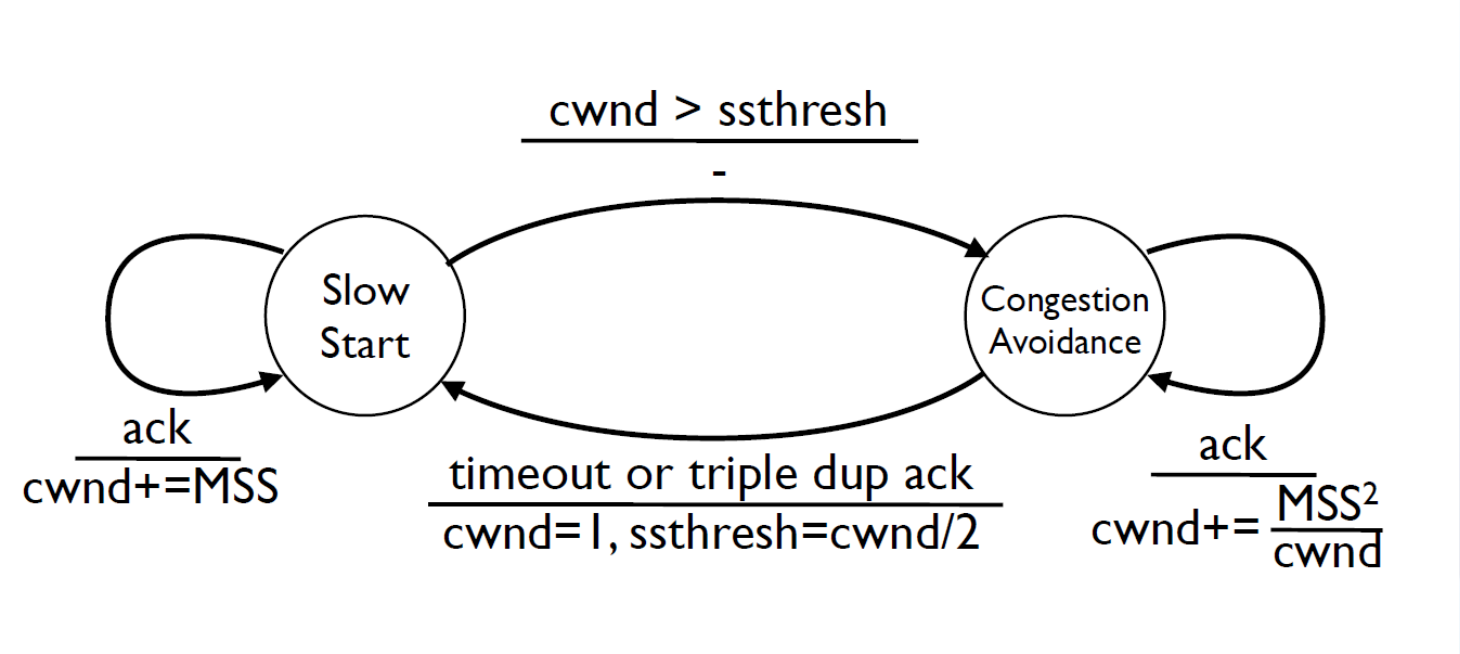
Определение перегрузки

- Потеря пакета
  - по таймеру;
  - 3 повторных подтверждения (всего 4 подтверждения с одинаковым номером).
- …

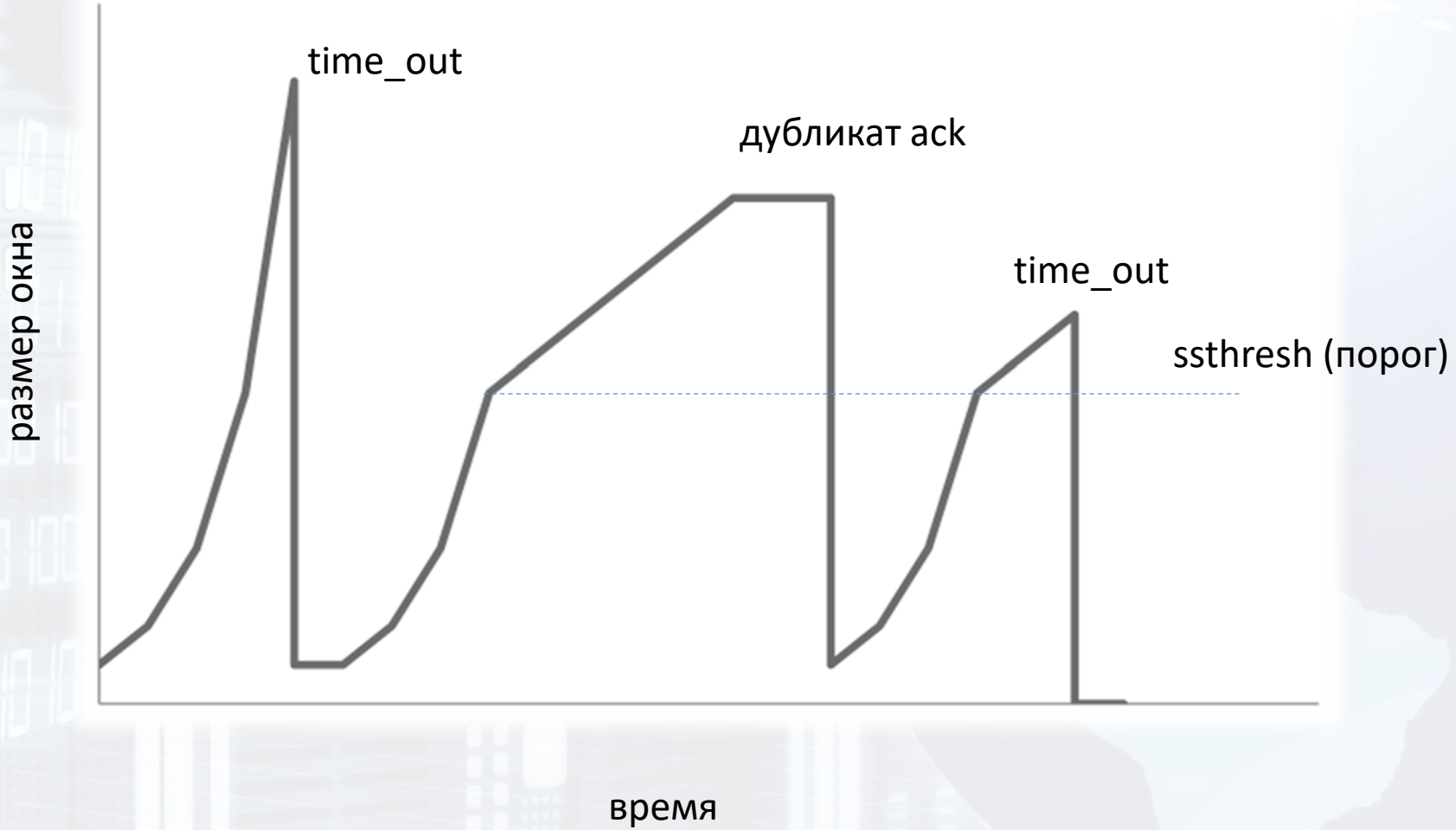
Стратегия Tahoe

- *Стратегия:*
  - *Используя медленный старт, быстро нащупать доступную пропускную способность сети*
  - *Приблизившись к насыщению, перейти в режим предотвращения перегрузки, очень осторожно пробируя возможность роста*
- *Три сигнала:*
  - *Рост номеров уведомлений – передача данных идет хорошо*
  - *Повторные уведомления – где-то произошла задержка/потеря данных, прекратить увеличение окна перегрузки*
  - *Если наступил Time_out или три раза получили dup ACK с одним и тем же номером, то устанавливаем порог = cwnd/2, cwnd = 1 и переходим в фазу медленного страта*
  - *Если пришло ACK с нужным номером, то продолжаем в фазе предотвращения перегрузки*

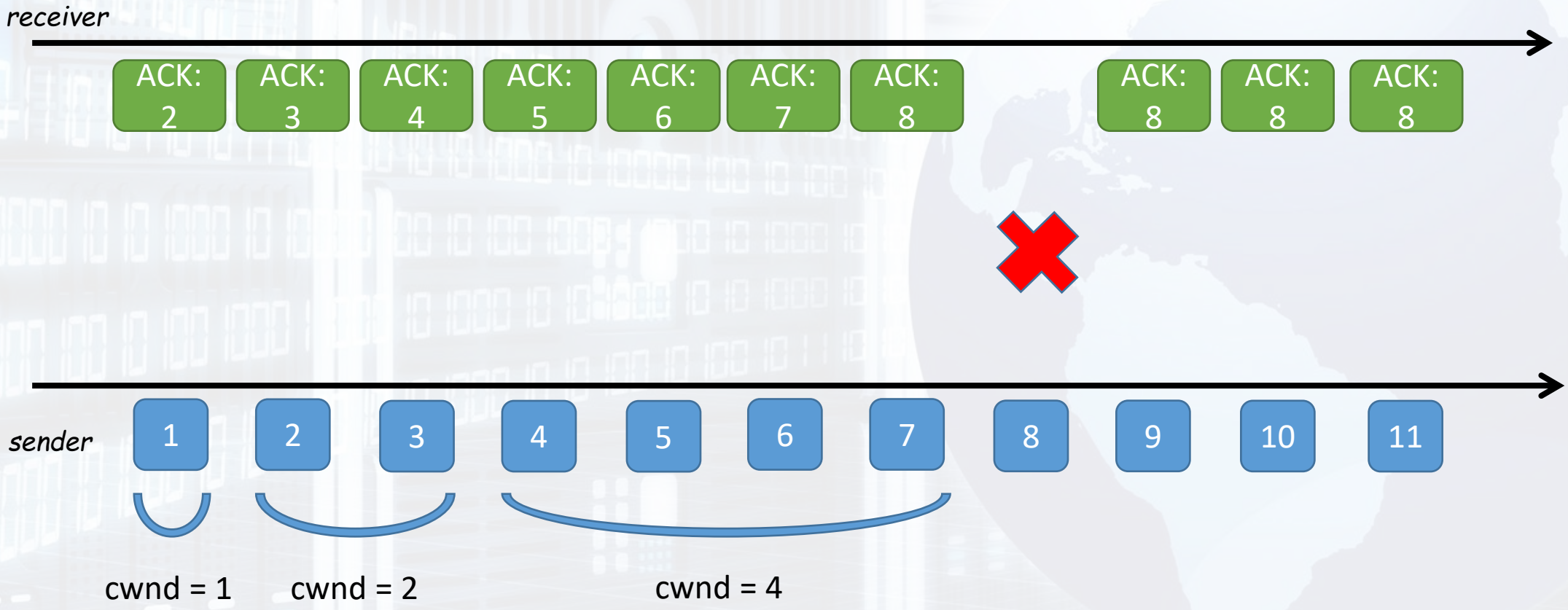
Диаграмма состояний TCP Tahoe



Динамика TCP Tahoe



Пример работы TCP Tahoe



Оценка time-out

- *RTT измерение критично для оценки time-out*
  - *Если слишком коротко – впустую тратим ресурсы сети на повторные передачи, «ломаем» медленный старт*
  - *Если слишком длинный – зря тратим ресурсы на ожидание*

- *Трудности –*
  - *RTT меняется очень динамично*
  - *RTT сильно зависит от загрузки (load) сети*

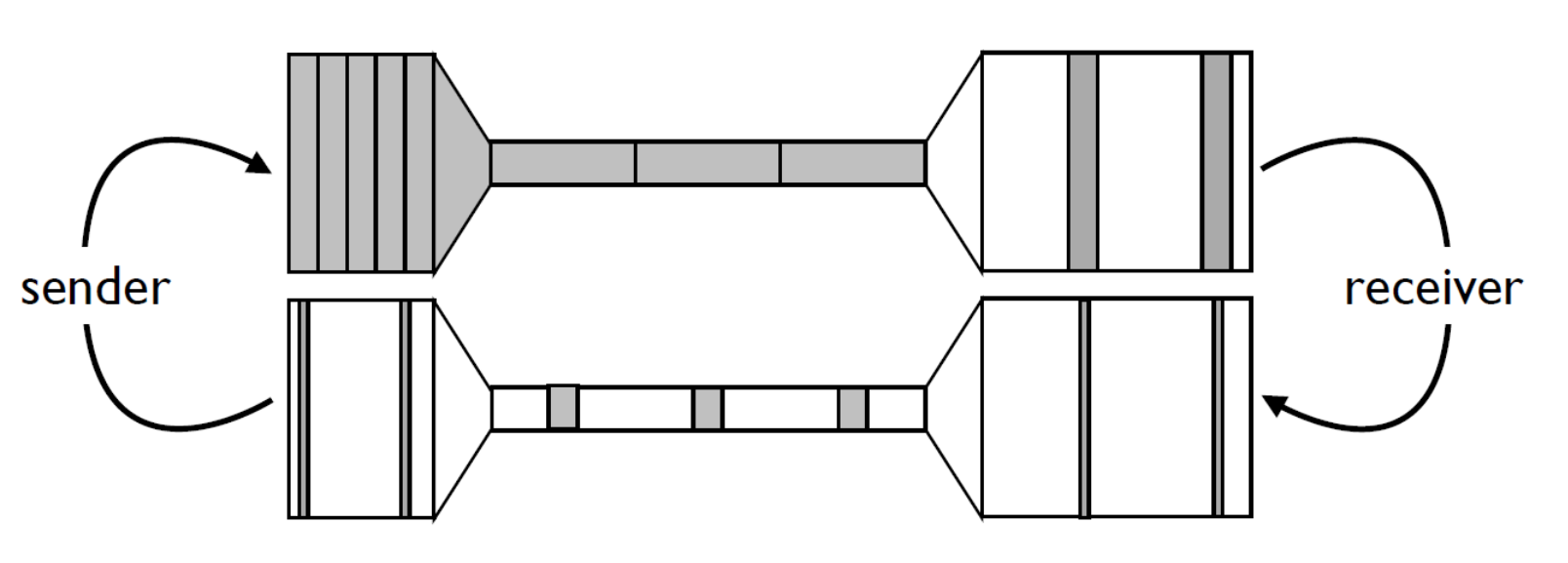
Pre Tahoe time_out

- *r – начальная оценка RTT из соображений здравого смысла*
- *m – измерение RTT для последнего подтвержденного пакета*
- *Вычисляем взвешенное среднее – $r=\alpha r + (1-\alpha)m$,*
  где $0< a <1$
- *Time_out= βr, где β = 2*
- *В чем проблема?*

TCP Tahoe time-out (т.2 стр.132-134)

- *r – начальная оценка RTT из соображений здравого смысла*
- *m – измерение RTT для последнего подтвержденного пакета*
- *Ошибка – e=m-r, где m измерение для последнего аск.*
- *Вычисляем r= ar + (1-a)m, где 1-α~ 0.125*
- *Измеряем вариацию – v = av + (1-a)|e|*
- *time-out = r + в x v, где в=4.*
- *В случае повторной передачи RTO = в * time-out*

Самонастройка



Принципы самонастройки

- *Отправлять данные только после того, как предыдущие покинули сеть*
- *Посылать данные только при получении уведомления*
- *Отправлять уведомления как можно быстрее – это важно!*

Прочесть «Congestion Avoidance and Control» van Jacobson and Karels

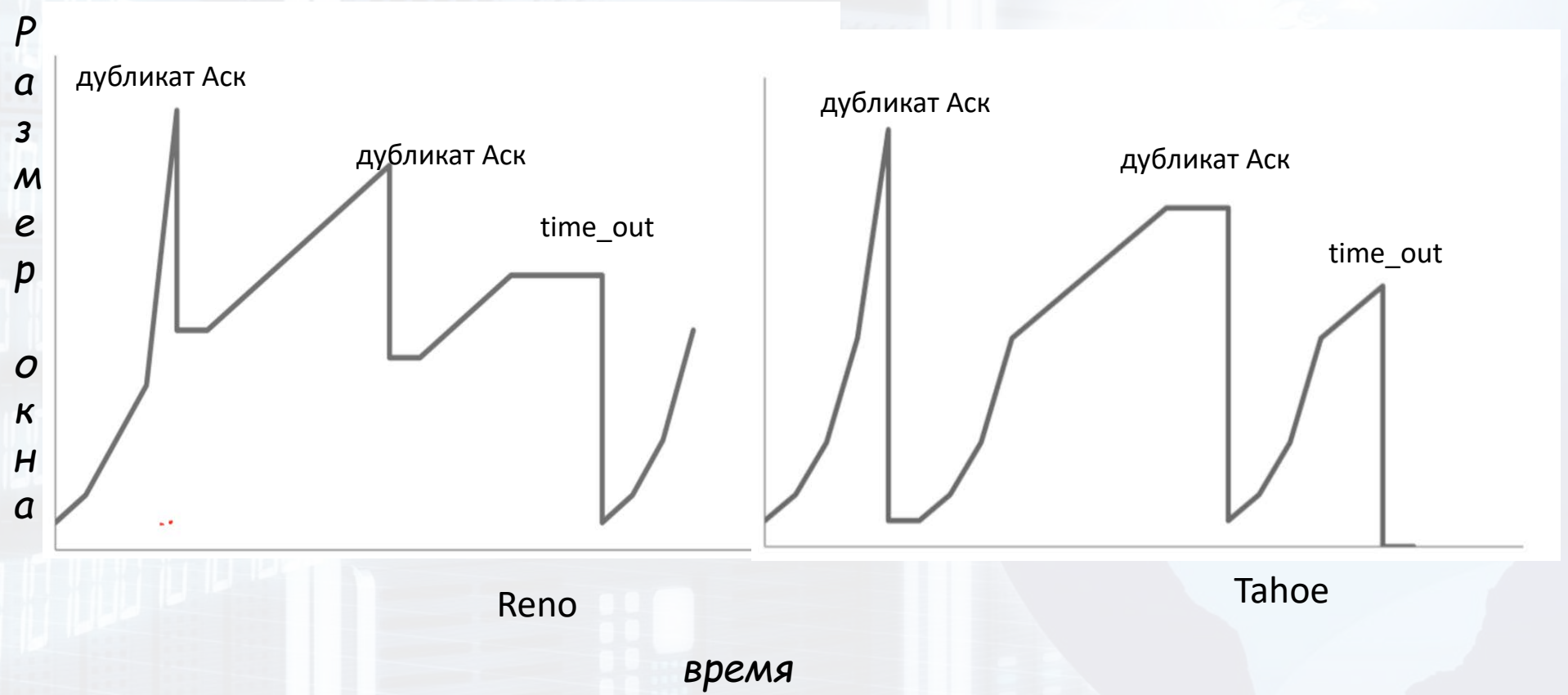
TCP Reno

Кафедра АСВК

TCP Tahoe vs TCP Reno

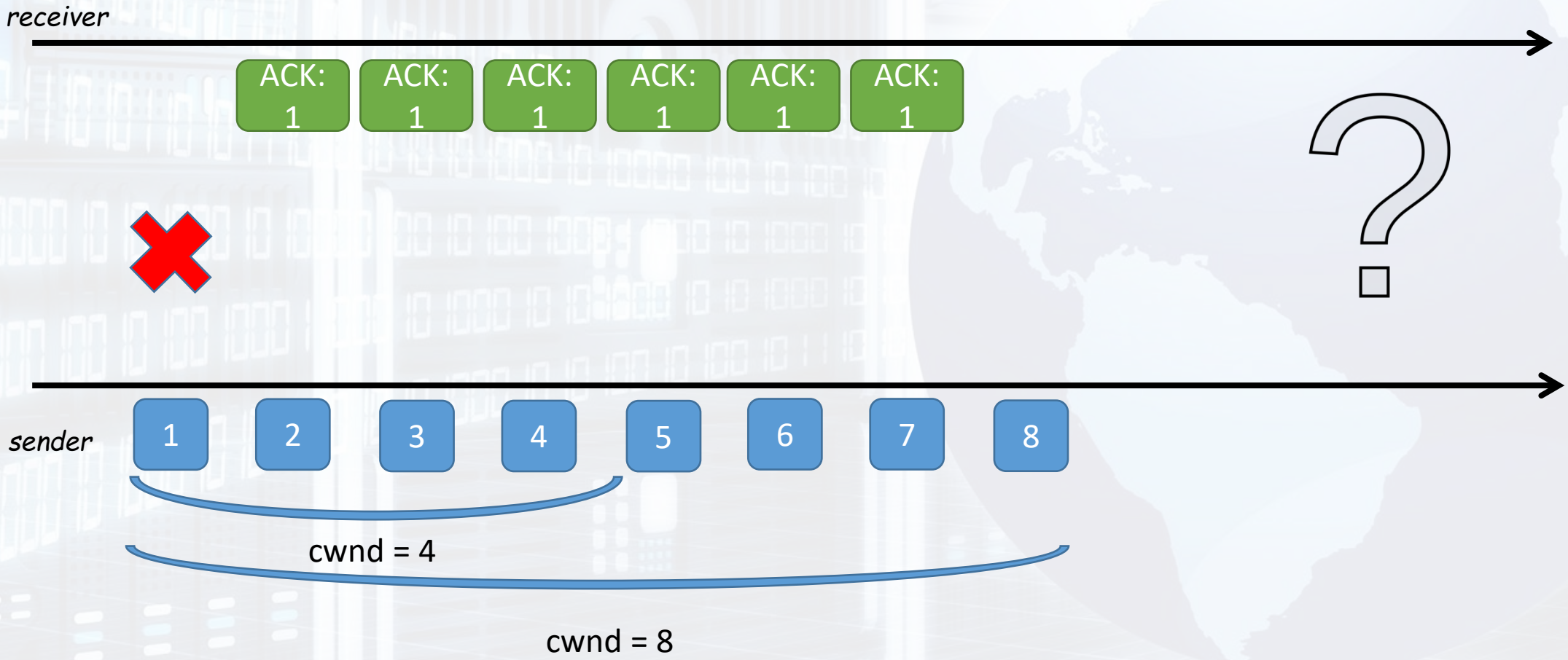
- ***TCP Tahoe*** *- Не ждать time-out, а уже по тройному уведомлению (подразумевается потеря пакета) начинать повторную пересылку и переход в фазу медленного старта*
  - *установить порог = CWDN/2*
  - *сбросить CWDN в 1*
  - *начать медленный старт*
- ***TCP Reno –*** *Добавлены фазы быстрой переотправки (fast retransmit) и быстрого восстановления (fast recovery):*
  - *по тройному ACK*
    - *cwnd = cwnd/2, а не 1*
    - *шлем запрашиваемый сегмент (fast retransmit)*
    - *Переходим в фазу быстрого восстановления (fast recovery)*
  - *По time-out схема Tahoe*

TCP Tahoe vs TCP Reno



Быстрое восстановление

Какая должна быть реакция на большое количество повторных подтверждений ?



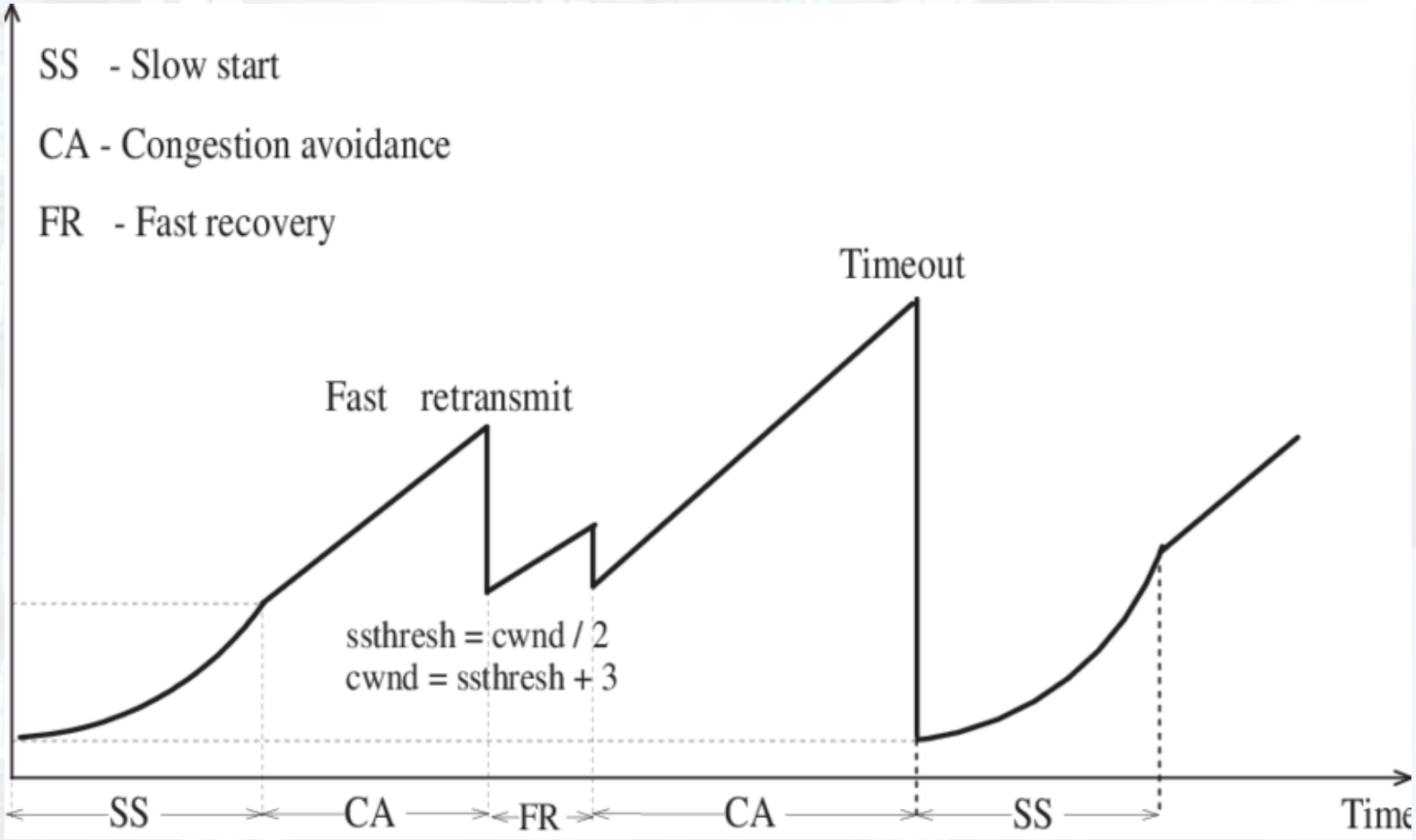
Быстрое восстановление

- *Окно перегрузки определяет количество пакетов, которое может одновременно находиться в сети*
- *Повторное подтверждение означает, что некоторый пакет был доставлен не по порядку -> в сети стало меньше пакетов, чем cwnd*
- *3 повторных подтверждения означают, что 3 пакета покинуло сеть*
- *Проблема: мы не можем сдвинуть sliding window вправо на 3 позиции из-за неподтвержденного пакета*
- *Решение: мы можем сдвигать правую границу sliding window (в данном случае congestion window), увеличивая значение sliding window*

Быстрое восстановление

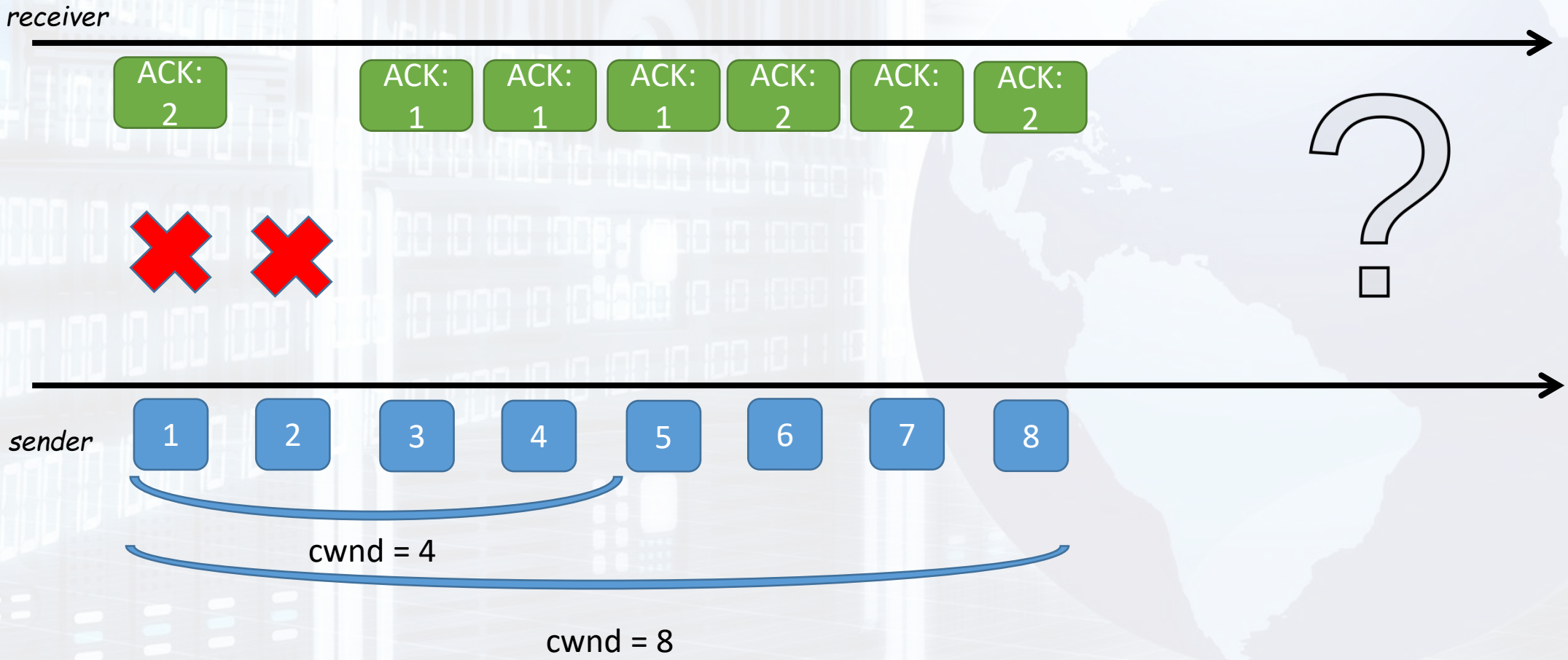
- В начале фазы быстрого восстановления устанавливаем окно в cwnd/2 + 3
- Каждое последующее повторное подтверждение означает выход еще одного пакета из сети, поэтому на каждый dup ack увеличиваем cwnd на 1: cwnd += 1
- При получении подтверждения на переотправленный пакет мы можем двигать левую границу sliding window, поэтому возвращаем значение cwnd в cwnd/2.

Быстрое восстановление



Быстрое восстановление: проблема нескольких потерь

Представим, что теряется несколько пакетов



TCP New Reno

- *Решает проблему нескольких потерянных пакетов*
- *По time-out поведение такое же как и у Tahoe/Reno*
- *В фазе быстрого восстановления:*
  - *при входе в эту фазу - запомнить последний не подтвержденный пакет*
  - *при каждом повторном уведомлении – увеличить CWND на 1*
  - *Начать отправку новых пакетов пока находимся в фазе быстрого восстановления*
  - *Когда последний пакет подтвержден:*
    - *вернуться в фазу предотвращения перегрузки*
    - *восстановить размер CWND до того размера, который оно имело до входа в фазу быстрого восстановления*

Определение перегрузки

- Потеря пакета
  - по таймеру;
  - 3 повторных подтверждения (всего 4 подтверждения с одинаковым номером).
- По изменению RTT (при перегрузке увеличивается задержка буферизации)
- По уведомлениям из сети (ECN)
- Гибридные методы
- ?

Управление перегрузками

- *Одна из сложнейших проблем в компьютерных сетях (особенно на неоднородных, протяженных, с ошибками соединениях)*
- *Основной подход: AIMD (additive increase, multiple decrease)*
- *Дома построить диаграмму состояний для TCP Reno*