Системное программное обеспечение локальных компьютерных сетей

Стек протоколов TCP/IP (Продолжение)

Автор: Денис Пынькин
Читает: Алексей Демидчук

2014 – 2015

e-mail: denis.pynkin@bsuir.by

http://goo.gl/32cTB

СЧАСТЬЕ ДЛЯ ВСЕХ, ДАРОМ, И ПУСТЬ НИКТО НЕ УЙДЕТ ОБИЖЕННЫЙ!

ARP

ARP – Address Resolution Protocol
протокол определения адреса

использующийся в компьютерных сетях протокол низкого уровня, предназначенный для определения адреса канального уровня по известному адресу сетевого уровня.

ARP: протокол определения адреса

Сетевой интерфейс имеет аппаратный адрес (48-битное значение для Ethernet или Token ring). Фреймы, которыми обмениваются на аппаратном уровне, должны адресоваться к корректному интерфейсу. Однако TCP/IP использует собственную схему адрессации: 32-битные IP адреса. Знание IP адреса хоста не позволяет ядру послать датаграмму этому хосту.

Драйвер Ethernet должен знать аппаратный адрес пункта назначения, чтобы послать туда данные. В задачу ARP входит обеспечение динамического соответствия между 32-битными IP адресами и аппаратными адресами, используемыми различными сетевыми технологиями.

Принцип работы

1. Узел, которому нужно выполнить отображение IP-адреса на локальный адрес, формирует ARP запрос, вкладывает его в кадр протокола канального уровня, указывая в нем известный IP-адрес, и рассылает запрос широковещательно.
2. Все узлы локальной сети получают ARP запрос и сравнивают указанный там IP-адрес с собственным.
3. В случае их совпадения узел формирует ARP-ответ, в котором указывает свой IP-адрес и свой локальный адрес и отправляет его уже направленно, так как в ARP запросе отправитель указывает свой локальный адрес.

ARP кэш

Эффективность функционирования ARP во многом зависит от ARP кэша (ARP cache), который присутствует на каждом хосте.
В кэше содержатся Internet адреса и соответствующие им аппаратные адреса.
Стандартное время жизни каждой записи в кэше составляет 2 минуты с момента создания записи.

Пример просмотра ARP-кэша

```
# arp -a
host1 (10.0.0.113) at 00:0f:fe:d4:e6:bb [ether] on eth0
host2 (10.0.0.15) at 78:e3:ff:94:63:58 [ether] on eth0
host3 (10.0.0.1) at 00:2f:29:35:03:7c [ether] on eth0
host4 (10.0.0.8) at 00:15:45:0e:2f:82 [ether] on eth0
```

Порт

Порт протокола транспортного уровня

идентифицируемый номером системный ресурс, выделяемый приложению, выполняемому на некотором сетевом хосте, для связи с приложениями, выполняемыми на других сетевых хостах (в том числе с другими приложениями на этом же хосте).

Для чего нужны порты?

Порты необходимы по двум причинам:

1. используются как идентификатор для разделения траспортных сессий между одинаковыми оконечными точками
2. используются для идентификации протокола прикладного уровня и ассоциированного с ним сервиса

1 порт = 16 бит

Для каждого из протоколов TCP и UDP стандарт определяет возможность одновременного выделения на хосте до 65536 уникальных портов, идентифицирующихся номерами от 0 до 65535.

1 порт = 16 бит

Для каждого из протоколов TCP и UDP стандарт определяет возможность одновременного выделения на хосте до 65536 уникальных портов, идентифицирующихся номерами от 0 до 65535.

Порты TCP не пересекаются с портами UDP. То есть, порт 1234 протокола TCP не будет мешать обмену по UDP через порт 1234!!!

Диапазоны портов

http://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xml:

- Системные (0-1023)
- Пользовательские (1024-49151)
- Динамические и/или частные (49152-65535)

Для большинства ОС системный диапазон доступен только с привилегиями администратора!

UDP – User Datagram Protocol

Протокол UDP (RFC-768) является одним из основных протоколов, расположенных непосредственно над IP. Он предоставляет прикладным процессам транспортные услуги, немногим отличающиеся от услуг протокола IP.

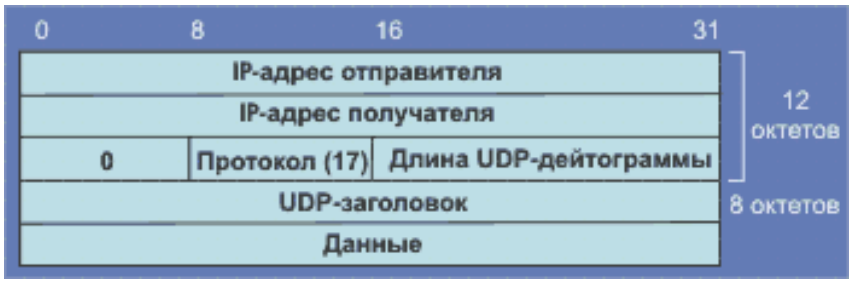
Протокол UDP обеспечивает доставку дейтограмм, но не требует подтверждения их получения. Протокол UDP не требует соединения с удаленным модулем UDP ("бессвязный"протокол)

Формат дейтаграммы UDP

Длина включает в себя заголовок.
Контрольная сумма рассчитывается с использованием псевдозаголовка!

Контрольная сумма

Контрольная сумма с использованием псевдозаголовка необходима как защита от дейтаграмм, ошибочно направленных по другому адресу.



Контрольная сумма UDP не является обязательной. Если она не подсчитывается, ее значение равно 0 (настоящая нулевая контрольная сумма кодируется всеми единицами).

TCP – RFC-793

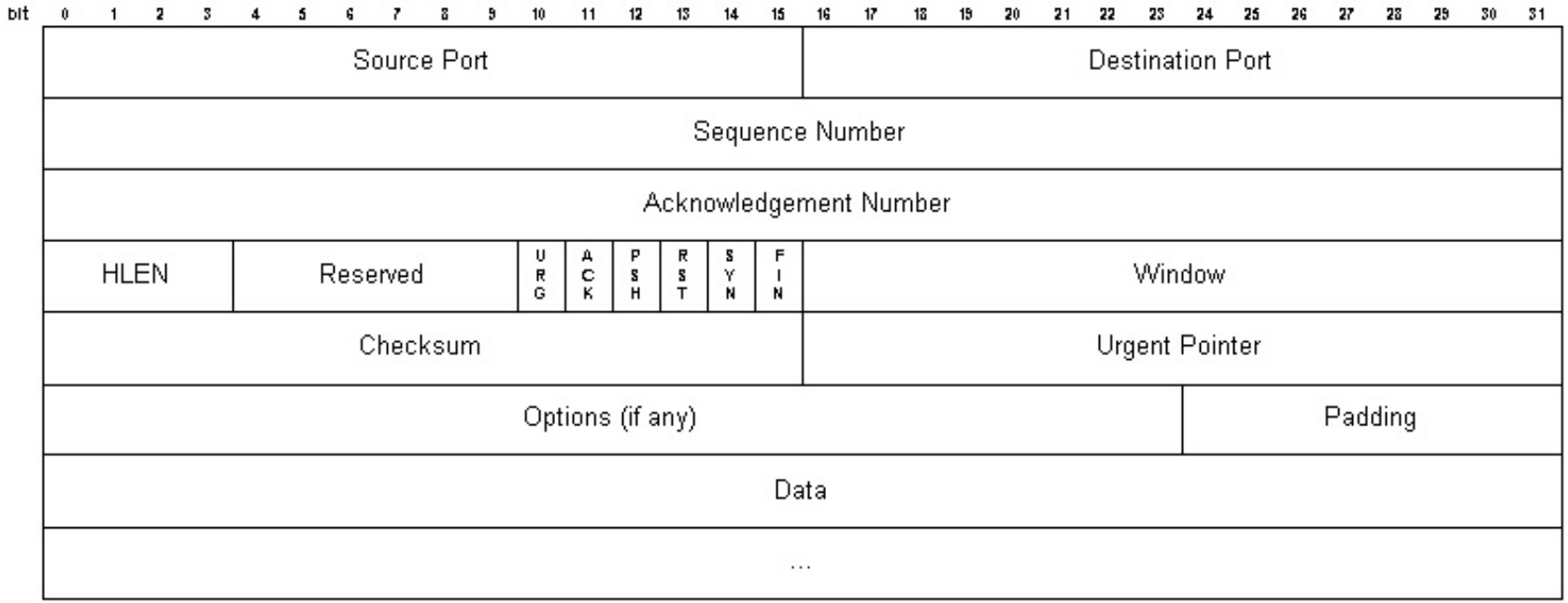
TCP – Transmission Control Protocol

Это *надежный* протокол *с установлением соединений*, позволяющий без ошибок доставлять *байтовый поток* с одной машины на любую другую машину объединенной сети. Он разбивает входной поток байтов на отдельные сообщения и передает их межсетевому уровню. В пункте назначения этот протокол собирает из полученных сообщений выходной поток. Кроме того, TCP осуществляет управление потоком.

Механизмы надежности TCP

- Контрольная сумма – позволяет определить повреждение данных и/или заголовка TCP-сегмента.
- Определение дублирующихся сегментов – "дубли"выкидываются.
- Повторная передача – используются подтверждения о доставке данных. Отсутствие подтверждений вместе с механизмом таймеров приводит к повторной передаче данных.
- Корректная последовательность – TCP собирает сегменты в правильном порядке.
- Таймеры – позволяют управлять повторной передачей сегментов.

Формат пакета TCP



HLEN – длина заголовка в 32-хбитных словах. В RFC обозначается, как "data offset"
window – указывает отправителю, сколько данных получатель готов принять

Urgent pointer – указывает смещение октета, следующего за срочными данными, относительно

первого октета

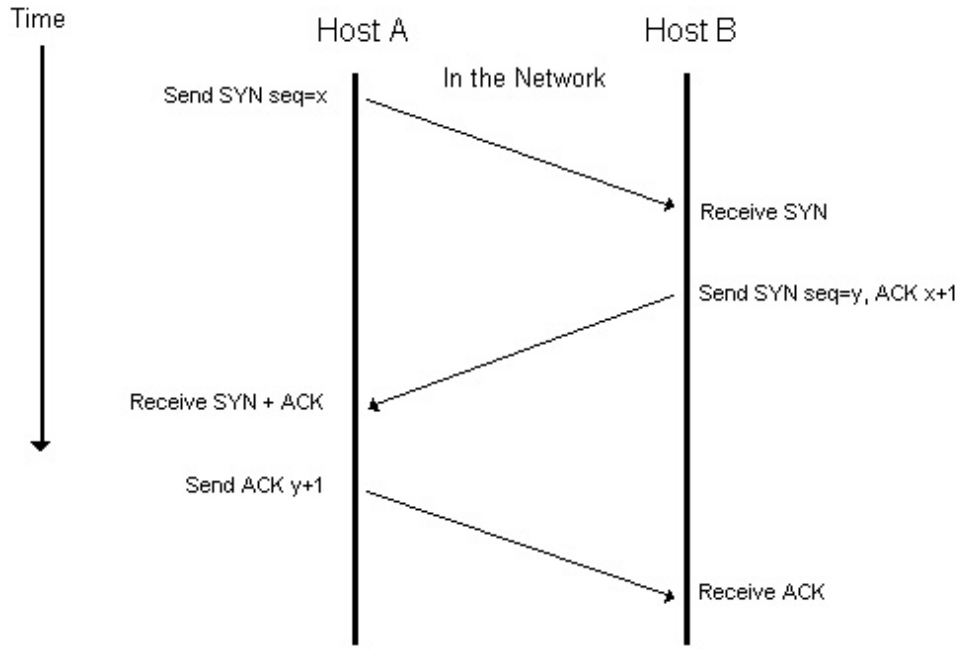
Поле Flags

- Urgent Pointer (URG)
- Acknowledgement (ACK)
- Push Function (PSH)
- Reset the Connection (RST)
- Synchronize (SYN)
- No More Data from Sender (FIN)

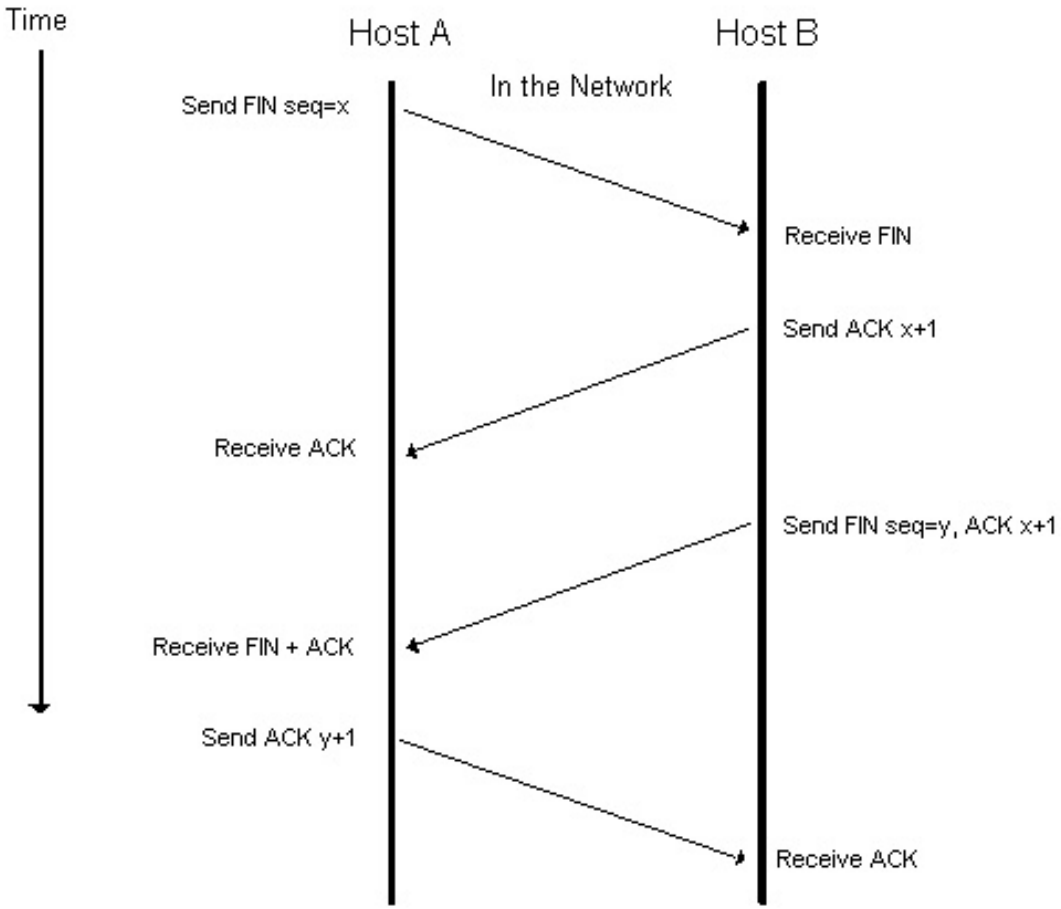
Поле Options

- End of Option List
- No-Operation
- Maximum Segment Size
- WSOPT - Window Scale
- SACK
- много дополнительных опций

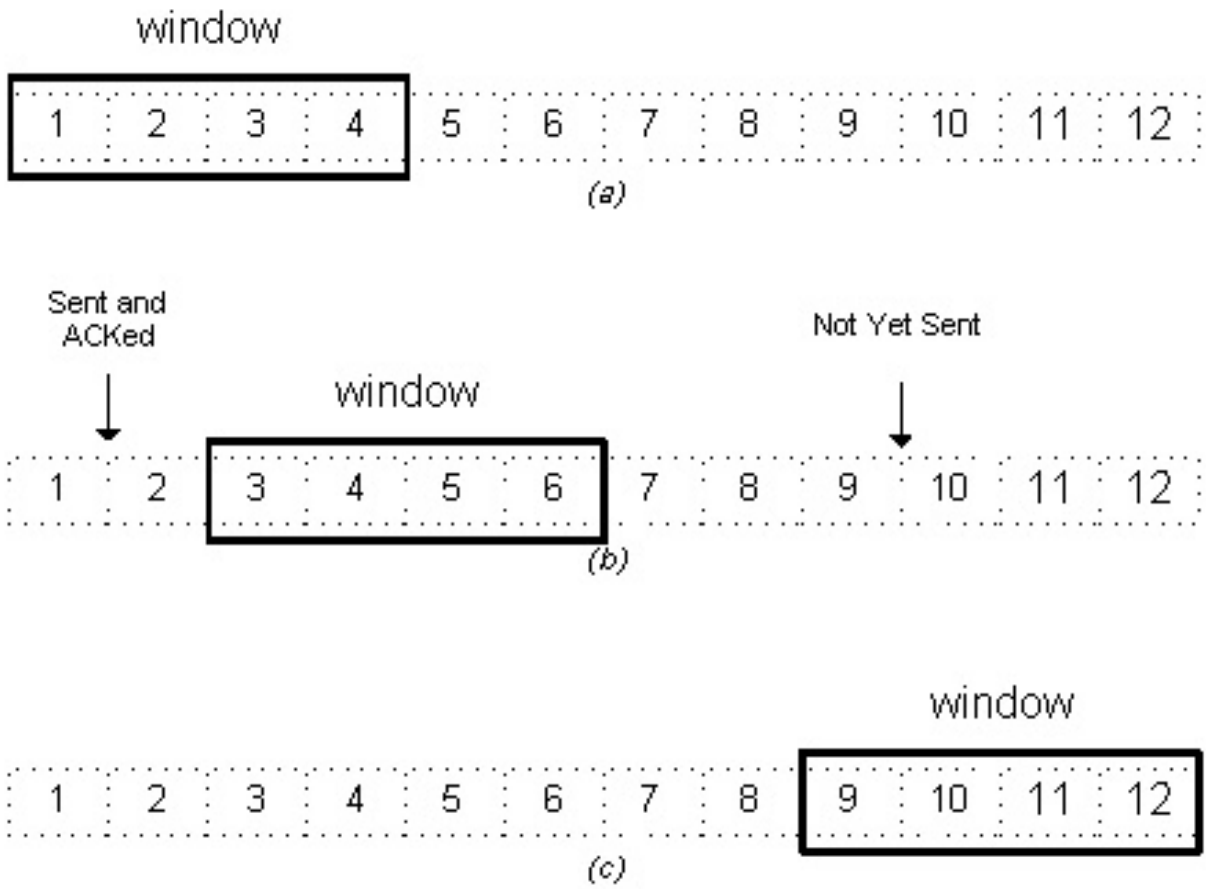
Установка соединения TCP



Разрыв соединения TCP



Управление потоком с помощью скользящего окна



Управление насыщением – механизм медленного старта

Congestion Control:

- Slow Start – с каждой успешной передачей размер буфера удваивается до половины точки насыщения– далее по 1 сегменту. Вплоть до размера окна.
- Congestion Avoidance – противовес медленному старту. Уменьшает размер окна в 2 раза, но не менее 2 сегментов. В случае таймаута – уменьшается до 1 сегмента и запускается механизм медленного старта.

Алгоритм Нэгла(Nagle)

Борется с неэффективной передачей.

```
if there is new data to send
  if the window size >= MSS and available data is >= MSS
      send complete MSS segment now
  else
    if there is unconfirmed data still in the pipe
      enqueue data in the buffer until an acknowledge is rec
    else
      send data immediately
    end if
  end if
end if
```

Синдром узкого окна

Такого рода проблема возникает в том случае, когда данные поступают отправителю крупными блоками, а интерактивное приложение адресата считывает информацию побайтно.

Синдром узкого окна

Такого рода проблема возникает в том случае, когда данные поступают отправителю крупными блоками, а интерактивное приложение адресата считывает информацию побайтно.

Кларк предложил не посылать уведомление о ненулевом значении ширины окна при считывании одного байта, а лишь после освобождения достаточно большого пространства в буфере. Например, когда адресат готов принять MSS байтов или когда буфер наполовину пуст.

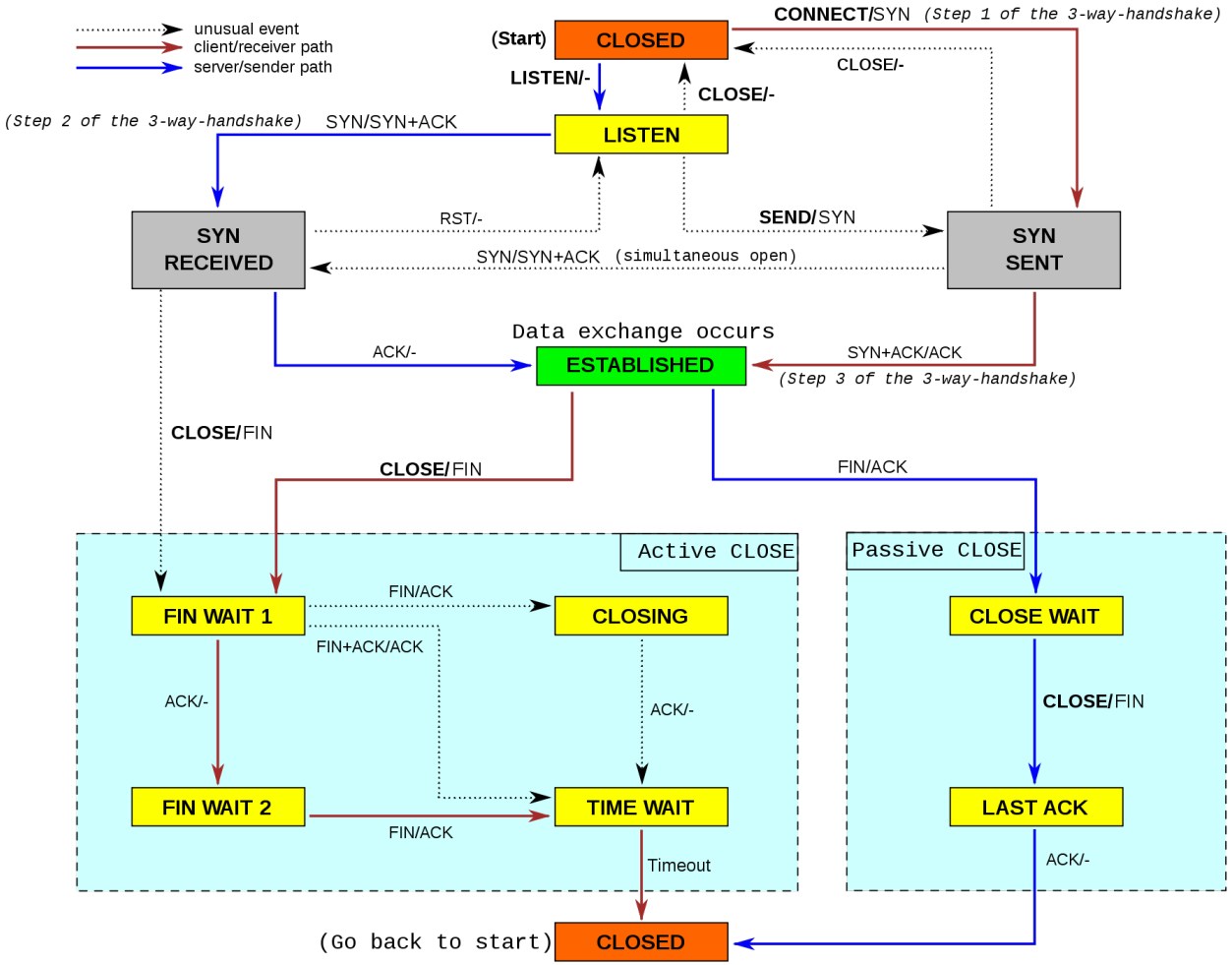
Таймеры TCP

- Таймер повторных передач (retransmission; RTO) – для сегмента
- Таймер запросов (persist timer) – для 0-го окна
- Таймер контроля работоспособности (keepalive)
- 2MSL-таймер (Maximum Segment Lifetime) – время в состоянии TIME_WAIT

Пример статистики TCP соединений

```
# netstat -antp
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address       Foreign Address             State       PID/Program name
tcp        0      0 0.0.0.0:45803       0.0.0.0:*                   LISTEN      6184/rpc.mountd
tcp        0      0 127.0.0.1:631       0.0.0.0:*                   LISTEN      6048/cupsd
tcp        0      0 192.168.0.1:56734   74.125.77.189:443           ESTABLISHED 23679/xulrunner
tcp        0      0 192.168.0.1:56724   74.125.77.189:443           TIME_WAIT   -
tcp        0      0 192.168.0.1:48176   69.171.241.10:5222          ESTABLISHED 10931/telepathy-gab
tcp        0      0 ::1:631             :::*                        LISTEN      6048/cupsd
```

Диаграмма состояний TCP



Спасибо за внимание!
Вопросы?