ИНДУСТРИЯ 4.0

X МЕЖДУНАРОДНАЯ КОНФЕРЕНЦИЯ РАЗРАБОТЧИКОВ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

SECON'2018

# Забег по типичным граблям TCP приложений в Linux


Александр Попов

SDN Switch Team Lead,
Brain4Net, Москва



2018.SECON.RU

# О себе

с 2010 GPGPU, OpenCL, OpenGL

с 2012 JRuby, Ruby, Backend

Сетевые приложения

с 2015 по сейчас - Brain4Net, C++ и Software-Defined Networking

Работа с сетью непосредственно

# О чем говорим и зачем

Накопленный опыт работы с TCP

Абстракции над сетью текут, нужно знать что внизу чтобы нормально писать

Обобщенно для всех сетевых приложений на Linux, не привязано к языку программирования

**О чем не говорим**

HighLoad

Другие ОС(Windows, iOS), хотя должно быть похоже

Если вы писали nginx или high load сервер, то вам будет неинтересно

# Антипаттерн: “Подкрутим в ОС”

Попытка решение какой-нибудь проблемы тюнингом параметров ОС без вникания в суть

Чаще всего идет от отделов эксплуатации, вынужденных поддерживать ваше приложение

Часто проблема в приложении и ее можно решить без этого

Тюнинг ОС требует полного понимания, что нужен именно он

**Решение**

Поищите проблему в приложении, используйте инструменты

# Антипаттерн: Вера в надежность TCP. Или как потерять данные в TCP

Отправляем данные, с той стороны тихо умирает хост

**Вопрос:** Узнаем ли мы сколько данных до него дошло?

**Ответ:**

# Антипаттерн: Вера в надежность TCP. Или как потерять данные в TCP

Отправляем данные, с той стороны тихо умирает хост

**Вопрос:** Узнаем ли мы сколько данных до него дошло?

**Ответ:** Нет

**Выводы**

1. Из приложения нельзя узнать дошли ли данные
2. Все ACK они про ОС, не про приложение
3. Если хотите узнать, что приложение(не ОС!) получило данные, отправляйте собственное подтверждение на уровне протокола

**Почитать**

https://blog.netherlabs.nl/articles/2009/01/18/the-ultimate-so_linger-page-or-why-is-my-tcp-not-reliable

# Таймауты: Подключение

Пытаемся подключиться к серверу, сервер недоступен.

**Вопрос:** Через сколько ОС скажет, что все, не смогла?

**Ответ:**

# Таймауты: Подключение

Пытаемся подключиться к серверу, сервер недоступен

**Вопрос:** Через сколько ОС скажет, что все, не смогла?

**Ответ:** 127 секунд, т.е. около 2 минут

Что обычно слишком долго

**Решение**

1. Non-blocking socket + connect() + select() w/ timeout
2. Ищите параметр timeout в вашем фреймворке или библиотеке

# Таймауты: Сколько будет ждать read()?

Ждем чтения из сокета, с той стороны тихо умер сервер или сеть

**Вопрос:** Через сколько приложение поймет, что ждать уже бесполезно и можно закрыть сокет?

**Ответ:**

# Таймауты: Сколько будет ждать read()?

Ждем чтения из сокета, с той стороны тихо умер сервер или сеть

**Вопрос:** Через сколько приложение поймет, что ждать уже бесполезно и можно закрыть сокет?

**Ответ:** Никогда не поймет, так и будет висеть вечно

Лично видел сокеты, висящие по 8 часов

Это нормально, так спроектирован TCP

Вечное ожидание может быть скрыто

**Решение**

Используйте таймауты

# Таймауты: Через сколько закроется сокет?

Отправили данные на хост, который уже не жив

**Вопрос:** Через сколько ОС решит, что сокет уже не живой и вернет ошибку?

**Ответ:**

# Таймауты: Через сколько закроется сокет?

Отправили данные на хост, который уже не жив

**Вопрос:** Через сколько ОС решит, что сокет уже не живой и вернет ошибку?

**Ответ:** 13-30 минут в Linux, tcp_retries2

**Выводы**

Таймаут большой, используйте свой по необходимости

# Таймауты: TCP Keepalive

Одно из первых, что приходит после гугления проблемы проверки жизни соединения

По RFC ждет 2 часа!

И это глобальная настройка на хост

Но в Linux можно крутить настройки для каждого соединения

# Проверка жизни соединения(heartbeats)

Сначала кажется простой задачей, на деле куча нюансов

Наивная реализация имеет проблемы

“Застревание хертбитов”

В общем виде задача сложная, но решение в ослаблении условий

**Читаем**

Martin Sústrik, creator of ZeroMQ, nanomsg, libdill, Cartesian:250bpm:TCP and heartbeats http://250bpm.com/blog:22

Re: Protocol for TCP heartbeats?
https://www.ietf.org/mail-archive/web/ietf/current/msg62500.html

# Антипаттерн: Тестирование через kill -9

Хотим проверить как ведет себя приложение при отказе сети

Поступает предложение убить “жестко” приложение на другой стороне через kill -9

Что не так?

ОС во время закрытия приложения корректно закрывает сокет

**Выводы**

TCP-коннект - ресурс ОС, не приложения и за его жизнь отвечает ОС

Для полного тестирования нужно жестче рубить сеть через iptables или специальные шаталки соединений

# Антипаттерн: Считывание в бесконечный буфер

Если активно вычитывать, то соединение разгоняется и никакого буфера не хватит

Каждое TCP соединение - это еще и встроенный flow control, реализованный в ОС

**Решение**

Нужно протянуть через все приложение этот flow control

# Инструменты: Классика

- netstat, ss, lsof
- tcpdump/wireshark
- strace НЕОБХОДИМО ЗНАТЬ
  1. Позволяет смотреть взаимодействие в ОС
  2. Независим от языка, можно дебажить даже неизвестную платформу, бывает даже удобнее чем из языка
  3. Умеет тестировать
  4. Стоит посмотреть два доклада Дмитрия Левина (https://youtu.be/2K1M9ggC8Gk https://youtu.be/bD4WC3-soA8 )
  5. Стоит минимум один раз целиком прочитать man strace

# Инструменты: strace workflow

# Ищем PID процесс

ps -eF | grep myapp

# Цепляемся к процессу

sudo strace -f -yy -e trace=network -p 1691

ppoll([{fd=27, events=POLLIN}], 1, NULL, NULL, 8

man ppoll

int ppoll(struct pollfd *fds, nfds_t nfds, const struct timespec *timeout_ts, const sigset_t *sigmask);

If timeout_ts is specified as NULL, then ppoll() can block indefinitely.

# Инструменты: Новое время(ebpf)

В новых(4.X) ядрах Linux есть возможность заглянуть в ядро

И это закрывает gap между “смотреть трафик” и приложением, т.е. позволяет мониторить слой(ОС) между приложением и сетью

bcc:connect, accept, tcptop и другие

https://github.com/iovisor/bcc

Я написал свой https://github.com/alexgpg/deeptcptracer

Ловит интересные события: смена TCP state, retransmission, получение нулевого окна, получение RST, получение FIN

# Почитать

man read, recv, send, write, connect

man tcp

https://www.kernel.org/doc/Documentation/networking/ip-sysctl.txt

High Performance Browser Networking by Ilya Grigorik https://hpbn.co/

Однако крайне рекомендую и практический подход tcpdump+strace

# Выводы

TCP не тот, чем кажется

TCP-коннект это ресурс ОС, не приложения

Управление этим ресурсом весьма опосредованное

Думайте о таймаутах т.к. дефолтные таймауты огромные

Тестируйте приложение жестче

Умейте пользоваться инструментами ОС для отладки

Держите под рукой схему состояний
http://www.cs.northwestern.edu/~agupta/cs340/project2/TCPIP_State_Transition_Diagram.pdf

# Поддержка этого доклада

https://github.com/alexgpg/secon2018_tcp_talk

Слайды, ссылки, видео(если будет), фиксы, комментарии


Александр Попов

alexgpg@gmail.com
@alexgpg