

Пора ли отправлять С на свалку истории?

Пишем демонов на PHP с использованием расширения libevent



<http://www.devconf.ru>

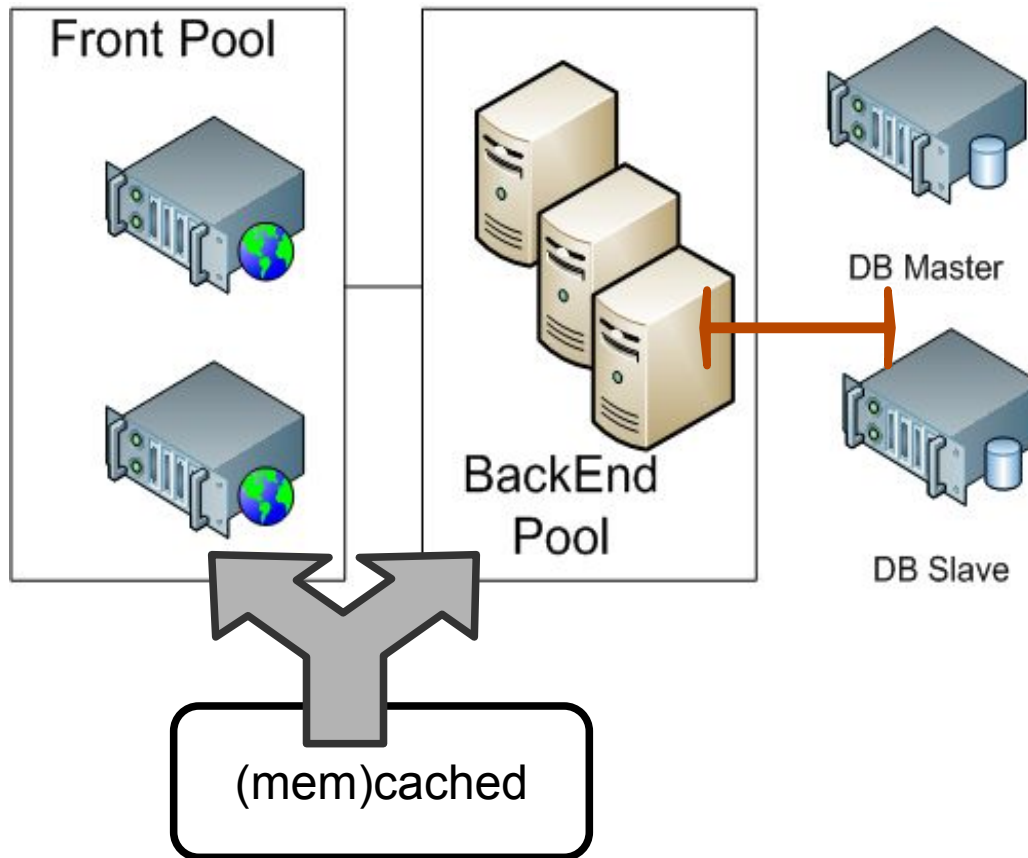
Кто мы такие?

- Вадим Крючков [Long], руководитель группы разработки
- Андрей Голубев [440hz], ведущий разработчик
- Евгений Прудников, ведущий разработчик



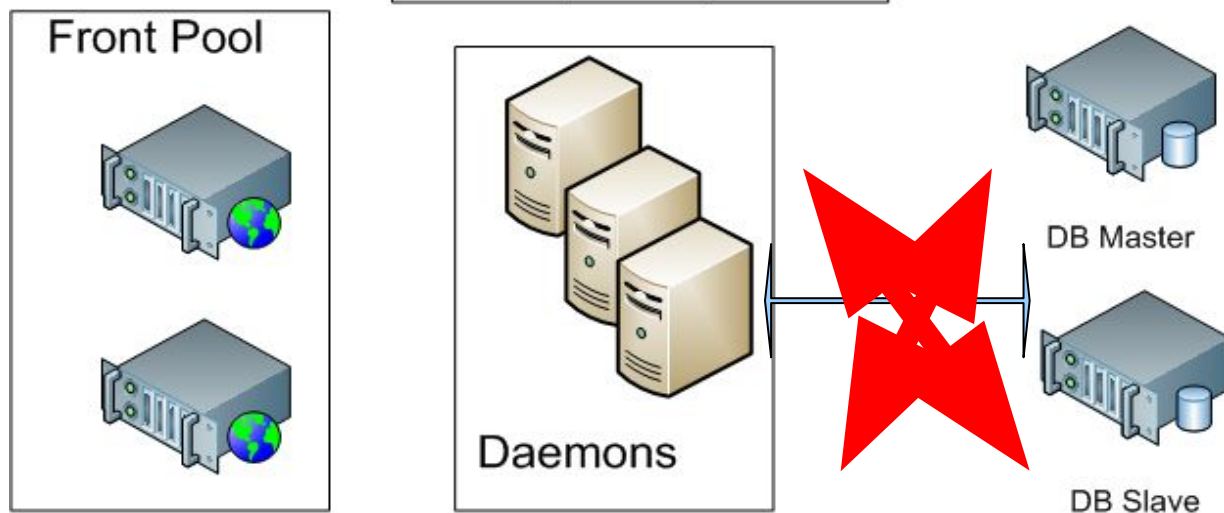
КОРПОРАЦИЯ ИГР

Обычная архитектура



Наша архитектура – включаем демоны

- | |
|---------------------|
| • Данные в памяти |
| • Готовые структуры |
| • Нет затрат на |
| • компиляцию |
| • инициализацию |



Демонизация. Что есть такое libevent?



- Предоставляет простой механизм для запуска callback функций, при наступлении определенного события на дескрипторе:
 - READ
 - WRITE
 - TIMEOUT
 - SIGNAL
- <http://www.monkey.org/~provos/libevent/>
- <http://ru.php.net/manual/en/intro.libevent.php>



Пишем демона, работающего с сокетом

```
// Создаем сокет - event вешается на дескриптор
```

```
$rSocket = stream_socket_server (
```

```
    'tcp://127.0.0.1:666',
```

```
    $errno, $errstr,
```

```
    STREAM_SERVER_BIND | STREAM_SERVER_LISTEN );
```

```
// далее его не блокирующим, что бы позволить принимать другие  
    КОННЕКТЫ
```

```
stream_set_blocking ( $rSocket, 0 );
```

Пишем демона – подключаем libevent

```
// создаем событийную базу
$rBaseEvent = event_base_new ( );

// создаем новое событие для сокета
$rSocketEvent = event_new ( );

/**
 * ловим события "чтение" и после операции чтения возвращаем
 * событие в базу
 * EV_READ - чтение
 * EV_PERSIST - вернуть событие в базу после выполнения
 */

event_set ( $rSocketEvent, $rSocket, EV_READ | EV_PERSIST,
'onAcceptEvent' );

// устанавливаем событие в базу событий
event_base_set ( $rSocketEvent, $rBaseEvent );

// запускаем отслеживание
```


Метод обработки

```
function onAcceptEvent ( $rSocket, $rEvent, $args ) {  
    global $rBaseEvent; // удобнее сделать через объект ;)  
  
    static $iConnect = 0; // идентификатор конекта  
    $iConnect++;  
    // Примем коннект  
    $rConnection = stream_socket_accept ( $rSocket );  
    // далее коннект не блокирующим, что бы позволить принимать  
    // еще коннекты  
    stream_set_blocking ( $rConnection, 0 );  
    // создадим буфер обмена данными  
    $buf = event_buffer_new ( $iConnect, 'onReadEvent', 'onWriteEvent',  
        'onFailureEvent', $iConnect);  
    // подключаем буфер к базе событий  
    event_buffer_base_set ( $buf, $rBaseEvent );  
}
```

<http://www.devconf.ru>

Метод чтения

```
$iBufferReadLenght = 1024; // размер буфера чтения
function onReadEvent($rStream, $args) {
    global $iBufferReadLenght;
    $tmp = "";
    do {
        $tmp .= event_buffer_read ( $hBuffer, $this->iBufferReadLenght );

        if( $iBufferReadLenght > strlen($tmp) ) {
            break;
        }
    } while ( true );

    return $tmp;
}
```

Превращаем демона в ...



или не документированные возможности

Таймеры (thnx 440hz)



- Стандартный таймер libevent'a не работает :(
- Выход есть!
 - событие можно повесить на «любой» дескриптор
 - `event_add (resource $event, int $timeout)`

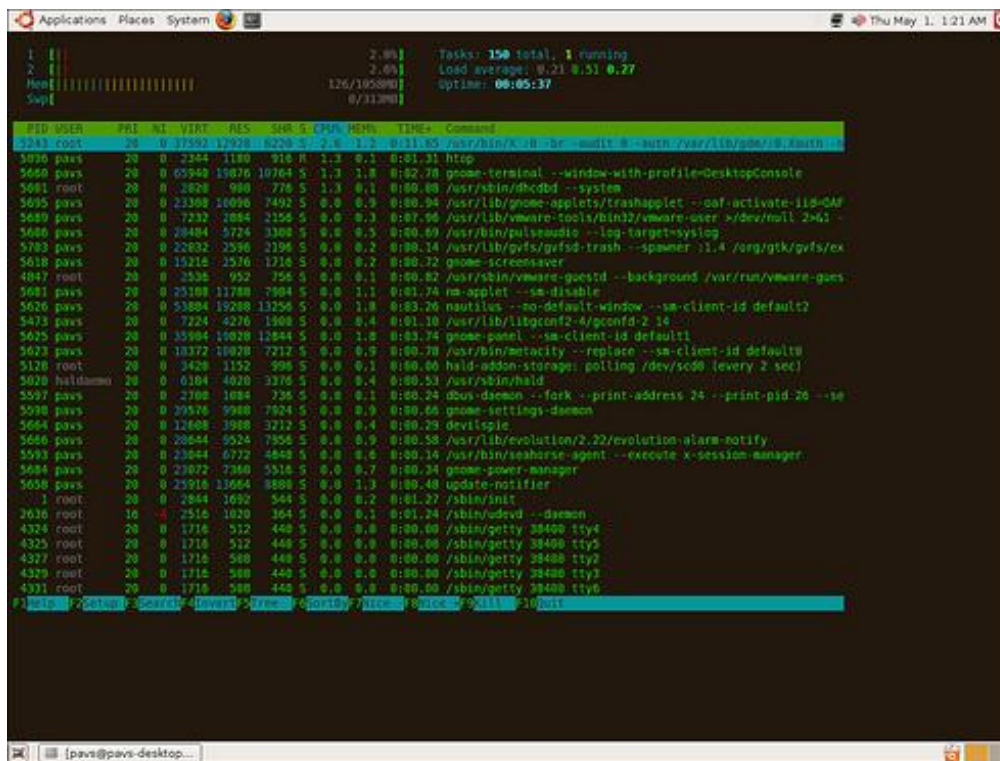
Таймеры - решение



- `tmpfile()` - открываем новый временный файл
- «вешаем» на этот дескриптор отложенное событие

```
event_set(
    $rTimers,
    $rTmpFile,
    0, |—————
    'onTimer',
);
```

Демонстрация



```

Applications Places System Thu May 11 1:21 AM
1 | | 2.0% Tasks: 150 total, 1 running
2 | | 2.0% Load average: 0.21 0.51 0.27
Mem| | 126/1958800 | Optime: 00:05:37
Swap| | 0/113800 |

PID USER PRI NI VIRT RES SHR S CPU% MEM% TIME+ Command
5243 root 20 0 17592 17920 6220 S 2.0 1.2 0:01.85 /usr/bin/xsh -br audit -s -suid /var/lib/udev/lib/ksmt
5036 pavv 20 0 2344 1180 916 R 1.3 0.1 0:01.33 http
5660 pavv 20 0 65940 19076 10764 S 1.3 1.8 0:02.78 gnome-terminal --window-with-profile=DesktopConsole
5681 root 20 0 2828 980 776 S 1.3 0.1 0:00.88 /usr/sbin/dhcdbd --system
5695 pavv 20 0 23308 10090 7492 S 0.9 0.9 0:00.94 /usr/lib/gnome-applets/trashapplet --caf activate-ild=0dF
5689 pavv 20 0 7232 2084 2156 S 0.9 0.3 0:07.90 /usr/lib/vmware-tools/bin32/vmware-user >/dev/null 2>&1
5680 pavv 20 0 20404 5724 3308 S 0.8 0.5 0:00.49 /usr/bin/polkitd --log-target=syslog
5783 pavv 20 0 22832 2596 2196 S 0.8 0.2 0:00.14 /usr/lib/gvfs/gvfsd-trash --spawner /1.4/org.gtk/gvfs/ex
5808 pavv 20 0 15216 2520 1716 S 0.8 0.2 0:00.72 gnome-screensaver
4847 root 20 0 2536 952 756 S 0.8 0.1 0:00.82 /usr/sbin/vmware-guestd --background /var/run/vmware-gues
5681 pavv 20 0 25108 11788 7904 S 0.8 1.1 0:01.74 nm-applet --sm-disable
5626 pavv 20 0 53804 19288 13256 S 0.8 1.8 0:03.26 nautilus --no-default-window --sm-client-id default2
5473 pavv 20 0 7224 4220 1908 S 0.8 0.4 0:01.10 /usr/lib/libgconf2-4/gconfd-2.14
5625 pavv 20 0 35904 19020 12844 S 0.8 1.8 0:03.74 gnome-panel --sm-client-id default1
5623 pavv 20 0 10372 10020 7212 S 0.8 0.9 0:00.78 /usr/bin/metacity --replace --sm-client-id default0
5128 root 20 0 3428 1152 996 S 0.8 0.1 0:00.00 hald-addon-storage: polling /dev/scd0 (every 2 sec)
5828 hald@dm 20 0 6104 4020 3376 S 0.8 0.4 0:00.53 /usr/sbin/hald
5597 pavv 20 0 2108 1084 736 S 0.8 0.1 0:00.24 dbus-daemon --fork --print-address 24 --print-pid 20 --se
5698 pavv 20 0 29576 9980 7924 S 0.8 0.9 0:00.66 gnome-settings-daemon
5664 pavv 20 0 12608 3988 3712 S 0.8 0.4 0:00.29 evilpic
5680 pavv 20 0 20844 5524 7956 S 0.8 0.9 0:00.58 /usr/lib/evolution/2.22/evolution-alarm-notify
5593 pavv 20 0 23844 6772 4840 S 0.8 0.6 0:00.16 /usr/bin/seahorse-agent --execute x-session-manager
5684 pavv 20 0 23872 7860 5516 S 0.8 0.7 0:00.24 gnome-power-manager
5698 pavv 20 0 25916 13664 8808 S 0.8 1.3 0:00.48 update-notifier
1 root 20 0 2844 1692 544 S 0.8 0.2 0:01.27 /sbin/init
2636 root 16 4 2516 1020 364 S 0.8 0.1 0:01.24 /sbin/udevd --daemon
4324 root 20 0 1716 512 440 S 0.8 0.0 0:00.00 /sbin/getty 38400 tty4
4325 root 20 0 1716 512 440 S 0.8 0.0 0:00.00 /sbin/getty 38400 tty5
4327 root 20 0 1716 500 440 S 0.8 0.0 0:00.00 /sbin/getty 38400 tty2
4329 root 20 0 1716 500 440 S 0.8 0.0 0:00.00 /sbin/getty 38400 tty1
4331 root 20 0 1716 500 440 S 0.8 0.0 0:00.00 /sbin/getty 38400 tty6

```

<http://cyberdot.ru/src/socket.php>

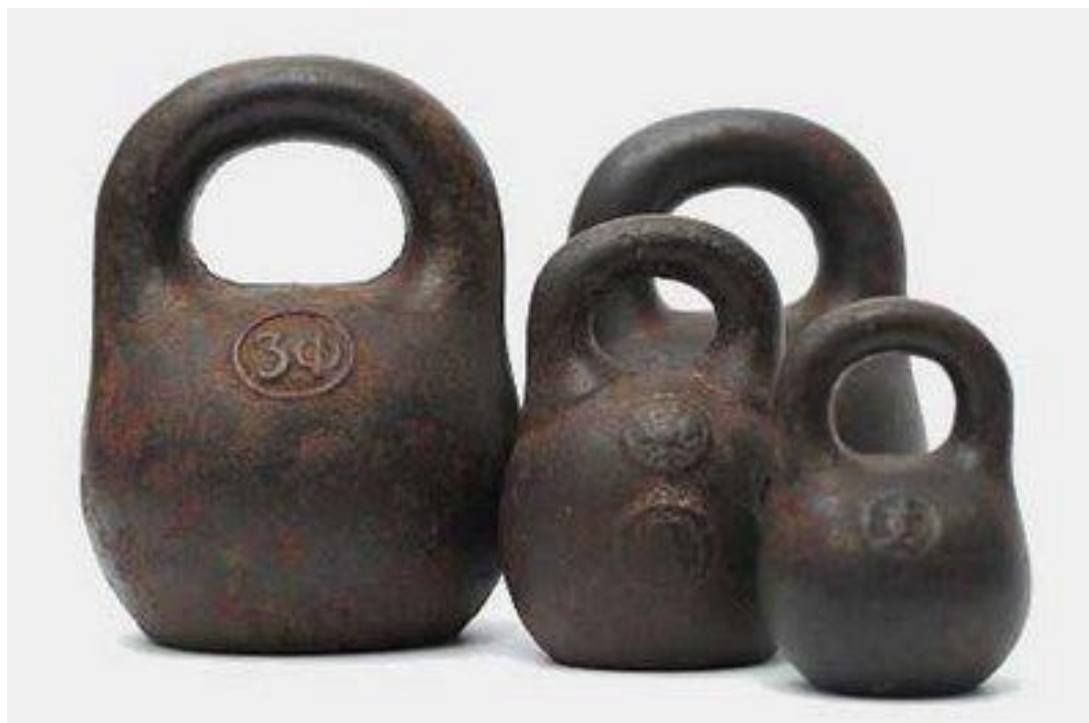
S

Подводные камни



- Очень мало информации и примеров
- Следить за ресурсами, не забываем их освобождать
- Хитрости при чтении данных, превышающих размер буфера
- Входных данных — много и они бывают «чужие» :)
- Проблемы с отслеживанием сигналов (EV_SIGNAL)

Даем нагрузку



Тестирование ботами



Имитируем ... пользователей в on-line:

- Воспользовались API
- Написали приложение, генерирующее ботов

Результаты

Сервер Xeon 8x2.66GHz, RAM 8Gb:

- Около 2.5 тысяч запросов в секунду (не Hello, World)
- На 1 пользователя в online расходуется около 1Мб памяти
- Приложение (пока) не подвергалось жесткой оптимизации

Результаты

Сервер Xeon 8x2.66GHz, RAM 8Gb:

- Около 2.5 тысяч запросов в секунду (не Hello, World)
- На 1 пользователя в online расходуется около 1Мб памяти
- Приложение (пока) не подвергалось жесткой оптимизации
- Память не «течет» (1 месяц публичного бета-теста не выявили)

Советы

- **Научитесь «мыслить параллельно»**
 - Процесс не завершается
 - Чужие данные
- **Читайте исходники — в них много полезного**
- **Если демон будет не один — напишите простенький фреймворк**
- **Документируйте код + протокол взаимодействия**
- **Напишите хороший логгер - без него отлаживать приложение будет сложно**
 - Сделайте несколько уровней логгирования



Выводы

Выводы пока делать рано

:)

Выводы (серьезно)

- Можно рекомендовать к использованию на продакшене
- Позволяет держать хорошие нагрузки (при этом оставляя LA в разумных пределах)
- Реальные тесты – придется подождать :(



Вопросы?

