Ускорение обработки сеансов

Обработка собственных сеансов в PHP в большинстве случаев прекрасна. Существует по крайней мере две возможные причины, по которым вы захотите изменить способ обработки сеансов:

* При использовании нескольких серверов необходимо иметь общее хранилище сеансов для обоих серверов.
* Сеансы PHP по умолчанию используют файлы, поэтому максимальная возможная производительность ограничена дисковым вводом-выводом.
* Сеансы PHP по умолчанию блокируют параллельные хранилища сеансов. В этом рецепте мы увидим, как использовать эффективное хранение для сеансов Yii.

Подготовка

Создайте новое yii2-app-basic приложение с помощью диспетчера пакетов Composer, как описано в официальном руководстве по адресу  
<http://www.yiiframework.com/doc-2.0/guide-start-installation.html>.   
По русски <http://yiiframework.domain-na.me/doc/guide/2.0/ru/start-installation>

и установить сервер Memcache и расширение MEMCACHE PHP.

Как это сделать...

Мы будем стресс-тестировать веб-сайт с помощью инструмента Apache ab. Он распространяется с двоичными файлами Apache, поэтому, если Вы используете Apache, вы найдете его в каталоге bin.

1. Выполните следующую команду, заменив ваш сайт фактическим именем хоста, который вы используете:

ab -n 1000 -c 5 http://yii-book.app/index.php?r=site/contact

This will send 1,000 requests, five at a time, and will output stats as follows:

This is ApacheBench, Version 2.3 <$Revision: 1528965 $>

Copyright 1996 Adam Twiss, Zeus Technology Ltd, <http://www.zeustech.net/>

Licensed to The Apache Software Foundation, <http://www.apache.org/>

Server Software: nginx

Server Hostname: yii-book.app

Server Port: 80

Document Path: /index.php?r=site/contact

Document Length: 14866 bytes

Concurrency Level: 5

Time taken for tests: 10.961 seconds

Complete requests: 1000

Failed requests: 0

Total transferred: 15442000 bytes

HTML transferred: 14866000 bytes

Requests per second: 91.24 [#/sec] (mean)

Time per request: 54.803 [ms] (mean)

Time per request: 10.961 [ms] (mean, across all concurrent requests)

Transfer rate: 1375.84 [Kbytes/sec] received

Connection Times (ms)

min mean[+/-sd] median max  
Connect: 0 0 0.0 0 0

Processing: 18 55 324.9 29 4702

Waiting: 15 41 255.1 24 4695

Total: 18 55 324.9 29 4702

Нас интересует показатель запросов в секунду. Это число означает, что веб-сайт может обрабатывать 91,24 запросов в секунду, если одновременно имеется пять запросов.

Замечание: Обратите внимание, что отладка не отключена, так как нас интересуют изменения скорости обработки сеанса.

1. Теперь добавьте следующее в /config/web.php раздел компоненты:

'session' => array(

'class' => 'yii\web\CacheSession',

'cache' => 'sessionCache',

),

'sessionCache' => array(

'class' => 'yii\caching\MemCache',

),

1. Запустите ab снова с теми же настройками. На этот раз, вы должны получить лучшие результаты. В моем случае это было 139.07 запросов в секунду. Это означает, что Memcache, как обработчик сеанса, выполнил 52% лучше, чем файловый обработчик сеанса по умолчанию.

Замечание:

Не полагайтесь на точные результаты, приведенные здесь. Все зависит от версий программного обеспечения, настроек и используемого оборудования. Всегда выполняйте все тесты самостоятельно в среде, в которой планируется развертывание приложения.

1. Вы можете получить значительный прирост производительности, выбрав правильный серверной обработки сеанса. Yii поддерживает больше кэширования серверных частей из коробки, включая wincache, xcache и кэш данных Zend, который поставляется с сервером Zend. Кроме того, вы можете реализовать собственную серверную часть кэша для использования быстрого хранения noSQL, например Redis.

Как это работает...

По умолчанию Yii использует собственные сеансы PHP; это означает, что файловая система используется в большинстве случаев. Файловая система не может эффективно работать с высоким параллелизмом.

Memcache или другие платформы отлично работают в следующей ситуации:

'session' => array(

'class' => 'yii\web\CacheSession',

'cache' => 'sessionCache',

),

'sessionCache' => array(

'class' => 'yii\caching\MemCache',

),

В предыдущем разделе config мы указываем Yii использовать CacheSession в качестве обработчика сеанса. С помощью этого компонента мы можем делегировать обработку сеанса компоненту кэша, указанному в кэше. На этот раз мы используем MemCache.

При использовании серверной части memcached следует учитывать тот факт, что при использовании этих решений пользователь приложения может потерять сеанс при достижении максимальной емкости кэша.

Замечание:

Обратите внимание, что при использовании серверной части кэша для сеанса Вы не можете полагаться на сеанс как на временное хранилище данных, так как тогда не будет памяти для хранения большего количества данных в memcached. В таком случае, это будет просто удалить все данные или удалить некоторые из них.

Если вы используете несколько серверов, Вы не можете использовать файловое хранилище. Не существует способа совместного использования данных сеанса между серверами. В случае memcached, это легко, потому что к нему можно легко получить доступ от стольких серверов, как вы хотите.

Кроме того, для обмена данных можно использовать DbSession:

return [

// ...

'components' => [

'session' => [

'class' => 'yii\web\DbSession',

],

],

];

Теперь создайте новую таблицу в базе данных:

CREATE TABLE session (

id CHAR(40) NOT NULL PRIMARY KEY,  
expire INTEGER,  
data BLOB

)

Кое что еще...

Рекомендуется как можно скорее закрыть сессию. Если Вы не собираетесь хранить что-либо в сеансе во время текущего запроса, вы можете даже закрыть его в самом начале действия контроллера. Таким образом, даже при использовании файлов в качестве хранилища ваше приложение должно быть в порядке.

Используйте следующую команду:

Yii:$app->session->close();

Смотрите так же

Дополнительные сведения о производительности и кэшировании см. по следующим URL-адресам:

* [http://www.yiiframework.com/doc-2.0/guide-tutorial-performance-tnning.html](http://www.yiiframework.com/doc-2.0/guide-tutorial-performance-tuning.html)
  1. по русски https://www.yiiframework.com/doc/guide/2.0/ru/tutorial-performance-tuning
* <http://www.yiiframework.com/doc-2.0/guide-caching-overview.html>
  1. по русски https://www.yiiframework.com/doc/guide/2.0/ru/caching-overview