افزایش سرعت و بازدهی در برنامه‌های

PHP

افزایش سرعت و بازدهی در برنامه‌های

PHP

(PHP Application Performance)

ترجمه و تألیف: مهندس محمدرضا کرامتی‌فر

انتشارات پندار پارس

سرشناسه :کرامتی‌فر، محمدرضا، ۱۳۶۲ -
عنوان و نام پدیدآور :افزایش سرعت و بازدهی در برنامه‌های PHP/ ترجمه و تالیف محمدرضا کرامتی‌فر، فرزانه مومنی‌زاده.
مشخصات نشر :تهران : پندار پارس، ۱۳۹۳.
مشخصات ظاهری :۲۴۰ص.: مصور، جدول.
شابک :۱۳۰۰۰۰ ریال: ۹-۵۰-۶۵۲۹-۶۰۰-۹۷۸
وضعیت فهرست نویسی :فیپا
موضوع :پی.اچ.پی. (زبان برنامه‌نویسی کامپیوتر)
موضوع :بازرگانی الکترونیکی -- برنامه‌های کامپیوتری
موضوع :وب-- سایت‌ها-- طراحی
شناسه افزوده :مومنی‌زاده، فرزانه، ۱۳۶۵ -
رده بندی کنگره :۸۶پ/ QA۷۶/۷۳ک۴ ۱۳۹۳
رده بندی دیویی :۰۰۵/۲۷۶۲
شماره کتابشناسی ملی :۳۴۲۲۴۰۰

انتشارات پندارپارس

دفتر فروش: انقلاب، ابتدای کارگرجنوبی، کوی رشتچی، شماره ۱٤، واحد ۱٦ **www.pendarepars.com**

تلفن: ٦٦٥٧٢٣٣٥ – تلفکس: ٦٦٩٢٦٥٧٨همراه: ۰۹۱۲۲٤٥۲۳٤۸ **info@pendarepars.com**

نام کتاب : **افزایش سرعت و بازدهی در برنامه‌های PHP**

ناشر : **انتشارات پندار پارس** (با همکاری جامعه‌ی برنامه‌نویس: http://barnamenevis.org)

ترجمه و تالیف : **محمد رضا کرامتی‌فر، فرزانه مؤمنی زاده**

چاپ نخست : **فروردین ۹۳**

شمارگان : **۱۰۰۰ نسخه**

لیتوگرافی : **ترام‌سنج**

چاپ، صحافی : **فرشیوه، خیام**

قیمت : **۱۳۰۰۰ تومان (به‌همراه کارت بوک‌کد)** **شابک : ۹-۵۰-٦٥۲۹-٦۰۰-۹۷۸**

فهرست فصل‌ها

فهرست

مقدمه‌ی مترجم

پیش از هر چیز، از اینکه این کتاب را برای خواندن انتخاب کردید سپاسگزارم.

یکی از مشکلاتی که همه‌ی برنامه‌نویسان، به‌ویژه در برنامه‌نویسی وب، با آن روبه‌رو هستند، مشکل پایین بودن سرعت و بازدهی است. طبق آمارهای بین‌المللی روانشناسی کاربر، شما بین ۱۰ الی ۱۳ ثانیه پس از ورود کاربر به سایت فرصت دارید که او را جذب و ترغیب به محتوای خود کنید. اینک بیاندیشید چه روی می‌دهد وقتی دست‌کم ٤٥ تا ٦٠ ثانیه زمان می‌برد تا سایت شما به‌صورت کامل لود شود! بنابراین باید فکری به حال سرعت کنیم.

بخشی از کتابی که پیش رو دارید ترجمه‌ی نسخه‌ی لاتین کتاب "PHP Application Performance" از انتشارات Apress می‌باشد که در دسامبر۲۰۱۰ منتشر گردیده است، و بخشی از آن تجربه‌ی شخصی مترجم در مورد برنامه‌نویسی و Perofrmance می‌باشد. بنابراین، خواندن این کتاب مناسب آن دسته از برنامه‌نویسان PHP است که قصد دارند با بهره‌گیری از متدها، ابزارها و تکنیک‌های معرفی شده در این کتاب ، کارایی و عملکرد برنامه‌های خود را ارتقا بخشند.

در بخش‌های ترجمه شده‌ی این کتاب تلاش کردیم تا از به‌کاربردن برابرهای فارسی عبارات رایج در برنامه‌نویسی جلوگیری کنیم و تا جای امکان، به زبان برنامه‌نویسان با خواننده ارتباط برقرار کنیم. اما با همه‌ی این تفاسیر، متن کتاب هنوز جای ویرایش دارد، بنابراین از خوانندگان محترم تقاضا می‌شود تا پیشنهادات و نظرات خود را به آدرس eng.keramati@gmail.com ارسال نمایند تا به خواست خداوند در نسخه‌های بعدی کتاب، بازنگری شود.

همچنین برای بالا بردن سطح معلومات خود در زمینه‌ی PHP Performance می‌توانید با مراجعه به وب‌سایت www.barnamenevis.info یا www.keramatifar.ir در دوره‌های حضوری که توسط نویسنده برگزار و تدریس می‌شود ثبت نام کرده و شرکت فرمایید.

در اینجا لازم می‌دانم از همکار محترم، سرکار خانم مهندس فرزانه مومنی زاده که در آماده‌سازی این اثر کمک‌های بسیاری به من کردند، قدردانی و سپاسگزاری نمایم.

پیش‌گفتار

چه یک مهندس PHP باشید که برای ایجاد برنامه‌های بزرگ سرش درد می‌کند یا کسی که تنها یک اپلیکیشن PHP با ترافیک بالا را پشتیبانی کرده است، این کتاب برای شما می‌باشد. شمایی که یک برنامه‌نویس PHP هستید و دید خوبی از آن دارید و خواهان درک "چراها"ی آن می‌باشید.

هدف این کتاب ایجاد یک تصویر کامل از همه‌ی ابزارهایی است که در زمان بهینه‌سازی برنامه‌ی PHP باید به‌کارگیری شوند.

از جاوا اسکریپت (JavaScript) گرفته تا نرم‌افزار وب‌سروری که برنامه را اجرا می‌کند، همه و همه در این کتاب گنجانده شده است. این کتاب به دو بخش کلی تقسیم می‌شود، front end[۱] و back end[۲] در یک برنامه‌ی تحت وب یا همان web application.

بخش نخست کتاب که در مورد front end است، به شما کمک می‌کند، تنگناهایی را که browser در حین عملکرد با آنها مواجه می‌شود، شناسایی کنید و آنها را از بین ببرید. این بخش همچنین شامل تمرین‌هایی برای استفاده از بهترین کدهای PHP و عمل caching با استفاده از ابزارهای موجود می‌باشد.

بخش دوم در مورد back end است که به شما در مورد انواع بی‌شمار نرم‌افزارهای وب‌سرور، چگونگی بهینه‌سازی نرم‌افزار و دیتابیس آموزش می‌دهد.

در اینجا نگاهی گذرا به محتویات فصل‌های کتاب می‌اندازیم:

فصل نخست: تکنیک‌های ارزیابی (benchmarking)

با ایجاد ابزار مورد نیاز برای اندازه‌گیری عملکرد برنامه، کار را آغاز می‌کنیم. ابزاری که برای نصب، خواندن نتایج و اجرا فرا خواهید گرفت، Apache Benchmark(ab) و Siege هستند که از معروف‌ترین ابزارهای ارزیابی در صنعت وب می‌باشند.

فصل دوم: بهبود دانلود کاربر و رندر کردن کارایی[۳]

عملکرد برنامه‌ی شما تنها به کد phpتان وابسته نیست. در این فصل تمرکز ما بر این است که بفهمیم مرورگرها چگونه محتوا را برمی‌گردانند و ابزارهای موجود برای تست (benchmark) جاوا

---

[۱] بخشی از برنامه است که کاربر با آن سرو کار دارد و از طریق browser قابل مشاهده است.

[۲] قسمتی از برنامه است که تنها برای ادمین در دسترس است.

[3] improving client download and rendering performance

اسکریپت، اندازه‌گیری مقدار داده‌ای را که مرورگر سعی در لود کردن آن دارد و اینکه یک مرورگر چقدر در هنگام لود کردن محتوا کارآمد است، خواهید آموخت. این کارها را با نصب و استفاده از ابزارهایی مانند firebug، Page Speed و Yahoo!'s YSlow انجام خواهید داد.

با استفاده از این ابزار، یک صفحه‌ی وب ساده را با تشخیص بهبود عملکرد آن برای JavaScript بهینه‌سازی می‌کنیم.

فصل سوم: بهینه‌سازی کد PHP

در این فصل، ایجاد یک حلقه‌ی for با اجرای سریع‌تر، چگونگی include کردن فایل‌ها با استفاده از functionهای بهینه شده‌ی PHP و از همه مهم‌تر چگونگی استفاده و نصب VLD،strace و Xdebug را خواهید آموخت.

هنگامی که VLD و strace نصب شدند، Opcode را همانند پروسه‌های سطح C Apache را که PHP script برای اجرا به آنها نیاز دارد آنالیز خواهید کرد. از سمت دیگر با استفاده از Xdebug، می‌توان تنگناهای موجود در کد PHP را شناسایی کرد.

فصل چهارم: Opcode Caching

دانستن چرخه‌ی زندگیPHP برای بهینه‌سازی مهم است. در این فصل مراحلی که PHP طی می‌کند، از زمانی که یک کاربر درخواست ارسال می‌کند، تا شناسایی نواحی را که ما می‌توانیم بهینه‌سازی را با استفاده از opcodecacherها انجام دهیم یاد خواهیم گرفت. همچنین نصب و کانفیگ کردن OpcodeCacherهایی چون APC، Xcache و eAccelarator را فرا خواهیم گرفت.

فصل پنجم: کش کردن‌های گوناگون

در این فصل با ابزارهای گوناگون کش کردن مثل memcached و همچنین استفاده از APC برای ذخیره‌ی اطلاعات آشنا خواهید شد. نصب، کانفیگ و اجرای یک مثال ساده را برای آشنایی با نرم‌افزار در کنار آموزش یک مثال واقعی که از مجموعه نتایج یک دیتابیس استفاده می‌کند، خواهید آموخت.

فصل ششم: انتخاب وب‌سرور مناسب

اگرچه به تازگی وب‌سرورهای جدیدتر و مهیج‌تر از Apache آمده‌اند اما در این فصل با جزئیات وب‌سرور Apache آشنا خواهیم شد و آن را با وب‌سرورهای جدیدتری چون LightTPD و Nginx مقایسه خواهیم کرد.

فصل هفتم: بهینه‌سازی وب‌سرور Apache

Apache یک وب‌سرور بسیار کارآمد و حرفه‌ایست که با کمی tunning و تعدادی تکنیک می‌توان عملکرد و ماندگاری آن را افزایش داد. در این فصل همچنین نگاهی به رمز و رازهای scaling out که ترافیک بالاتر را پشتیبانی می‌کند می‌اندازیم.

فصل هشتم: بهینه‌سازی دیتابیس

در بیشتر برنامه‌های تحت وب، سرور دیتابیس نقش عمده‌ای ایفا می‌کند. در این فصل با بهینه‌سازی سرور دیتابیس mysql و پیاده‌سازی متدها و ابزارها آشنا می‌شویم.

فصل ۱

تکنیک‌های Benchmarking

تلفن زنگ می‌خورد و صدایی از آن طرف فریاد می‌زند: چرا این برنامه، ۲۰۰ کاربر همزمان را پشتیبانی نمی‌کند؟

شما نفس عمیقی می‌کشید و من‌من کنان می‌گویید: من نگاهی می‌اندازم و یک راه‌حل برای این مشکل پیدا می‌کنم.

چندی پیش، باید یک برنامه‌ی PHP دیتابیس‌محور می‌ساختید. به‌عنوان یک برنامه‌نویس PHP با تجربه، شروع به نوشتن کد، ایجاد لایه‌های معماری آغازین، PHP back end، CSS و JavaScript می‌کنید و از آنجا که در فتوشاپ مهارت دارید، یک قالب گرافیکی نیز خلق می‌کنید و برنامه‌ی تولید شده را وارد بازار می‌نمایید.

هنگامی که برنامه‌ی شما معروف شد و بازدید کننده‌های بیشتری از سایت شما دیدن کردند، آن وقت است که حجم وسیعی از شکایت‌ها از راه می‌رسند. همه‌ی پیام‌ها حاوی یک معضل هستند: سایت یا اصلا جواب نمی‌دهد یا خیلی کند است.

در این هنگام نگاهی به کدتان می‌اندازید و به این فکر می‌کنید که چگونه این مشکل را با بیرون کشیدن کدهایی که باعث کند شدن برنامه شده‌اند برطرف سازید، و در آخر به این پرسش بر می‌خورید که چگونه یک برنامه‌ی تحت وب می‌تواند عملکرد خوبی داشته باشد در زمانی که ۵۰، ۱۰۰، ۲۰۰ یا حتی ۳۰۰ کاربر به‌طور همزمان به هاست وب‌سایت شما درخواست ارسال می‌کنند؟ و یا اینکه چگونه می‌توان برنامه را زیر همچنین بار ترافیکی تست کرد؟

در این فصل ما دو ابزار اوپن سورس برای تست کارایی (benchmarking) را بررسی می‌کنیم که نه تنها به پرسش‌های این چنینی پاسخ می‌دهند، بلکه کمک می‌کنند تغییرات در عملکرد برنامه را هنگام اعمال کارایی بسنجیم. ابزارهایی که استفاده می‌کنیم Apache Benchmark(ab) و Siege می‌باشند.

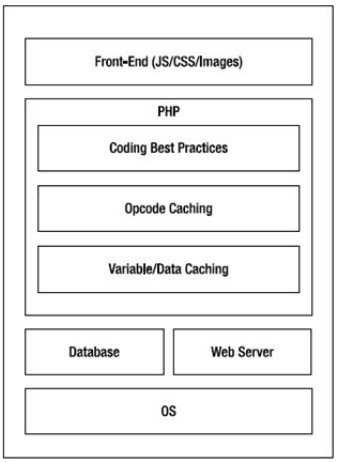
خواهید آموخت که چگونه این ابزار را نصب کنید، چگونه نتایج آنها را بخوانید و از آنها استفاده کنید تا انواع گوناگون محتوا، از HTML ساده گرفته تا یک عکس بزرگ را درخواست کنید.

در آخر به‌صورت پایه‌ای با چگونگی فرایند درخواست و پاسخ HTML (request/response) آشنا می‌شویم و اینکه یک درخواست، در پشت برنامه چه عملیاتی انجام می‌دهد.

اما نخست بیایید پشته یا (stack) برنامه‌ی PHP را کشف کنیم.

پشته‌ی برنامه‌ی PHP

هر برنامه‌ی PHP یک stack دارد، که هنگامی که مورد مشاهده قرار می‌گیرد همانند شکل ۱-۱ می‌باشد:



شکل ۱-۱: پشته‌ی برنامه‌ی PHP

بیشتر برنامه‌های PHP از طریق مرورگر و با استفاده از کد front-end در قالب JavaScript(JS)، CSS[1]، فلش (flash)، عکس و تکنولوژی‌های دیگر front-end به کاربر نمایش داده می‌شوند. front end نشان داده شده در بالاترین بلوک پشته‌ی یک برنامه‌ی PHP، به کاربران کمک می‌کند تا به سمت برنامه هدایت شوند، و لایه‌ی PHP را فعال کنند.

لایه‌ی PHP شامل منطقه‌ی خاصی از برنامه است که معمولا یا با دیتابیس یا با یک وب‌سرویس در تعامل است، تا اگر از یک سیستم ذخیره‌سازی خارجی استفاده می‌کند داده‌های پویا را ذخیره سازد. در آخر، همه‌ی برنامه‌های تحت وب PHP در یک چیز مشترک‌اند: باید روی یک وب‌سرور همچون Apache یا Nginx که روی یک سیستم‌عامل نصب شده‌اند، نصب شوند.

---

[1] Cascading Style Sheet

هر لایه‌ی درون برنامه‌ی PHP می‌تواند بهینه شود. از front end گرفته تا وب‌سرور. این کتاب همه‌ی لایه‌های نشان داده شده در شکل را در بر می‌گیرد، اما ما به ابزاری نیاز داریم که کارایی برنامه را پیش و پس از اعمال افزایش‌دهنده‌ی کارایی اندازه‌گیری کند که Apache Benchmark و Seige این کار را انجام می‌دهند.

نرم‌افزارهای تست کارایی

Ab وSeige ابزاری هستند که چگونگی پاسخ یک وب‌سرور را در مقابل درخواست‌های شبیه‌سازی شده‌ی گوناگون از سمت کاربر اندازه می‌گیرند. آنها به ما اجازه می‌دهند که هر تعداد دلخواهی از کاربران را که بر روی وب‌سرور درخواست ارسال می‌کنند و از آن مهم‌تر بازدیدهایی که به‌صورت همزمان از سوی کاربران انجام می‌شود، شبیه‌سازی کنیم.

برای نمونه، هر یک از این ابزارها، اطلاعاتی همچون موارد زیر را ارائه می‌کند:

- مدت زمانی که طول می‌کشد تا یک درخواست، پاسخ دهد (دریافت کند).
- میزان پاسخ دریافت شده از سمت سرور.
- تعداد درخواست‌هایی که یک وب‌سرور در هر ثانیه می‌تواند مدیریت کند.

کار این ابزار تست عملکرد نیست، بلکه تست درخواست‌های اجرا شده‌ی روی یک وب‌سرور خاص می‌باشد.

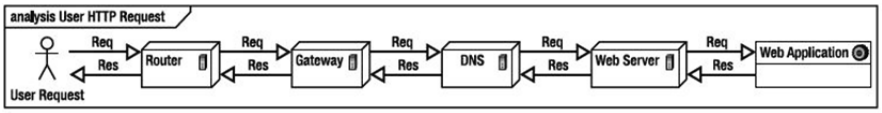
به‌دلایل زیر ما از ab و seige استفاده می‌کنیم:

- **سهولت در استفاده**: هر دو ابزار ab و seige تنها یک خط برای تایپ کردن با تعداد کمی آپشن برای استفاده دارند، که این یعنی یادگیری این ابزار از پایه، بسیار آسان و سریع است.
- **نصب آسان**: هر دوی این ابزارها بسیار ساده و ظرف مدت زمان کمی نصب می‌شوند.
- **مبتنی بر خط فرمان** (**command-line based**): بیشتر برنامه‌نویسان و توسعه‌دهندگان وب، چه در سرور لینوکس و چه در ویندوز، از command line استفاده می‌کنند.

تعریف چرخه‌ی درخواست/پاسخ (request/response)

پیش از هر چیز باید بفهمیم که یک درخواست HTTP چیست و چه کار می‌کند، زیرا این ابزار برای اندازه‌گیری کارایی برنامه، از چرخه‌ی درخواست استفاده می‌کند.

درخواست‌های HTTP، عملیاتی است که از سمت کاربر برای دریافت محتویات یک وب‌سایت به‌سمت وب‌سرور ارسال می‌شود. این درخواست می‌تواند با تایپ آدرس سایت در مرورگر یا کلیک بر روی یک لینک اتفاق بیافتد. یک درخواست HTTP معمولی شامل اطلاعاتی در مورد هاستی که قصد دسترسی به آن را داریم، مرورگر و دیگر اطلاعات مفید برای وب‌سرور می‌باشد. شکل ۲-۱ فرایند یک درخواست/پاسخ HTTP را از سمت کامپیوتر شخصی کاربر نشان می‌دهد.



شکل ۲-۱: چرخه‌ی درخواست HTTP

این شکل، مراحل یک درخواست HTTP ساده را از وب‌سرور نشان می‌دهد. درخواست از کامپیوتر کاربر آغاز شده و پس از برخورد به روتر کاربر، گیت‌وی ISP و DNS که IP مرتبط با اسم دامین درخواست شده را جست‌وجو می‌کند، به وب‌سرور با IP تعیین شده می‌رسد و در آخر از برنامه (web application) درخواست می‌کند که محتویات خاصی را تولید کند.

قسمت دوم چرخه، پاسخ HTTP است. هنگامی که درخواست به وب‌سرور می‌رسد، وب‌سرور داده‌ای را که کاربر درخواست کرده، واکشی (fetch) و فرمت می‌کند. سپس بسته‌های داده را از همان مسیری که درخواست صورت گرفته بود برای کاربر ارسال می‌کند. اگر داده به اندازه‌ی کافی بزرگ باشد در بسته‌های کوچک‌تر فرستاده می‌شود و در هنگام ارسال، خطاهای (error) آن چک می‌شود و پیش از اینکه مرورگر فرآیند رندر کردن (rendering) را آغاز کند، این بسته‌های داده توسط مرورگر بازسازی می‌شوند. همه‌ی این مراحل باید پیش از رندر شدن صفحه‌ی وب توسط مرورگر انجام شود. هریک از این مراحل باعث کند شدن اجرای صفحه‌ی وب می‌شوند. ابزاری که در ادامه خواهیم آموخت، به ما امکان تست و اندازه‌گیری مدت زمان پاسخ‌گویی به درخواست کاربر از سمت برنامه را می‌دهد.

تست کارایی توسط Apache - Apache Benchmark

ab که یکی از ابزارهای شناخته شده برای تست کارایی است، به‌صورت پیش‌فرض روی Apache نصب می‌شود و قابلیت تست (load-test) وب‌سرور را از طریق ارسال یک‌سری درخواست‌های شبیه‌سازی شده، به‌سمت یک URL خاص، فراهم می‌کند.

استفاده از ab اطلاعات زیر را برمی‌گرداند:

- حجم داده‌ی منتقل شده (بایت).

- تعداد درخواست‌هایی که یک وب‌سرور در هر ثانیه می‌تواند پاسخ‌گو باشد.
- بیشترین زمان کامل شدن یک درخواست (میلی‌ثانیه).
- کمترین زمان کامل شدن یک درخواست (میلی‌ثانیه).

ab همچنین به‌شما اجازه می‌دهد که loadingهای شبیه‌سازی شده‌ی گوناگونی را اجرا کنید؛ مانند:

- درخواست‌های همزمان به یک محتوای خاص از یک وب‌سایت.
- درخواست‌های در یک مدت زمان خاص.
- درخواست‌های با Keep-Alive (برقرار نگه داشتن ارتباط) فعال.

از همه مهم‌تر، ab مستقل از وب‌سرور آپاچی کار می‌کند و به شما اجازه می‌دهد حتی زمانی که apache غیرفعال است، بتوانید ab را اجرا کنید.

نصب ab

در ادامه، با نصب فایل‌های مورد نیاز برای اجرای ab هم بر روی ویندوز و هم بر روی لینوکس آشنا خواهیم شد.

نصب بر روی Mac و Unix

اگر بر روی سیستم‌عامل یونیکس[1] هستید، گزینه‌های زیادی برای نصب Apache خواهید داشت. می‌توانید آن را از ports، yum، و apt-get نصب کنید یا سورس آن را دانلود کنید.

فهرست کامل دستورات نصب، در جدول ۱-۱ نشان داده شده است.

Repository	Command
YUM	Yum install apache2
Ports	sudo port install apache2
Apt-get	Apt-get install apache2

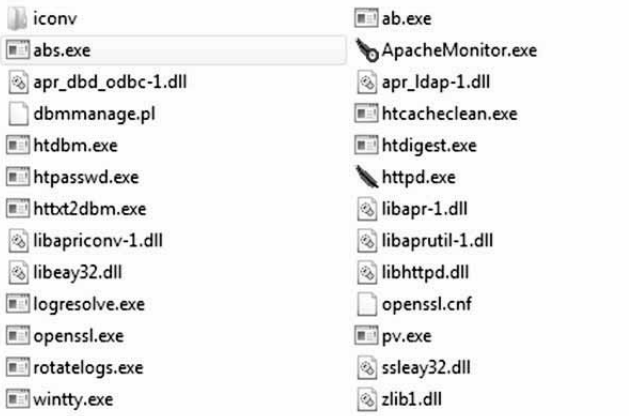
جدول ۱-۱. فهرست کامل دستورات نصب

کاربران Mac می‌توانند از MacPorts استفاده کنند و دستور ports نشان داده شده در جدول را از طریق ترمینال اجرا کنند.

[1] Unix OS

نصب بر روی ویندوز

کاربران ویندوز می‌توانند به وب‌سایت http://httpd.apache.org رفته، بر روی لینک دانلود From a Mirror در سمت چپ کلیک کرده و پکیج مناسب با نسخه‌ی ویندوز خود را دانلود کنند. پس از دانلود نرم‌افزار می‌توانید آن را هر جای سیستم که خواستید نصب کنید. محلی که برای نصب انتخاب می‌کنید APACHE_HOME شما خواهد بود و اگر دایرکتوری <APACHE_HOME>\Apache2.4\bin را باز کنید باید مجموعه‌ای از فایل‌ها و دایرکتوری‌هایی همانند شکل ۳-۱ را ببینید.



شکل ۳-۱: دایرکتوری bin آپاچی نصب شده در ویندوز

اجرای ab

نخستین تستی از benchmark که می‌خواهیم اجرا کنیم، یک تست ساده بر روی دامین www.example.com می‌باشد. هدف اصلی این تست، آشنا کردن شما با syntax ابزار، مرور همه‌ی گزینه‌های موجود و همچنین مرور یک پاسخ کامل است.

تمام دستورات ab از ساختار زیر پیروی می‌کنند:

```
ab [options] [full path to web document]
```

با استفاده از ab syntax یک درخواست ساده را شبیه‌سازی می‌کنیم.

اگر از ویندوز استفاده می‌کنید، کلیدهای Windows+R را بگیرید و در کادر Run، عبارت cmd را تایپ و OK را کلیک کنید تا محیط Command باز شود.

در لینوکس می‌توانید از ترکیب ALT+T برای باز کردن Shell Terminal استفاده کنید.

پس از باز شدن Command یا Terminal، وارد فولدر apache/bin در مسیر نصب apache شوید. برای نمونه، اگر از Xampp استفاده می‌کنید و آن را در درایو :c نصب کرده‌اید، برای ورود به فولدر bin دستور زیر را تایپ کنید:

```
Cd c:\xampp\apache\bin
```

اکنون برای اجرای ab، دستور زیر را تایپ کنید:

```
Ab –n 1 http://www.example.com/index.php
```

در این دستور تنها از یک گزینه استفاده شده و آن هم n است که به معنای تعداد درخواست‌هایی است که به‌سمت یک URL خاص ارسال می‌شود. n می‌تواند هر مقدار دلخواه کوچک‌تر از ۵۰۰۰۰ باشد و به‌صورت پیش‌فرض ۱ در نظر گرفته می‌شود.

بخش دوم دستور، بخش URL است که در دستور بالا http://keramatifar.i/index.php می‌باشد.

اگر بخواهیم صفحه‌ی خاصی همچون test.php را از این دامین تست کنیم، URL زیر را جایگزین می‌کنیم:

```
http://www.example.com/test.php
```

هنگامی که دستور ab را در ترمینال تایپ کردیم خروجی همانند شکل ٤-١ بر روی صفحه برگردانده می‌شود:

```
c:\xampp\apache\bin>ab -n 1 http://www.keramatifar.ir/index.php
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking www.keramatifar.ir (be patient).....done


Server Software:        nginx
Server Hostname:        www.keramatifar.ir
Server Port:            80

Document Path:          /index.php
Document Length:        20551 bytes

Concurrency Level:      1
Time taken for tests:   0.983 seconds
Complete requests:      1
Failed requests:        0
Write errors:           0
Total transferred:      20885 bytes
HTML transferred:       20551 bytes
Requests per second:    1.02 [#/sec] (mean)
Time per request:       982.802 [ms] (mean)
Time per request:       982.802 [ms] (mean, across all concurrent requests)
Transfer rate:          20.75 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:      281  281   0.0    281     281
Processing:   702  702   0.0    702     702
Waiting:      328  328   0.0    328     328
Total:        983  983   0.0    983     983

c:\xampp\apache\bin>
```

شکل ۴-۱: پاسخ ab برای response به http://example.com/index.php

هنگام تست سایت‌های دیگر، لطفا تعداد درخواست‌های خود را به وب‌سرور محدود کنید تا برای سرورهای از همه جا بی خبر، دردسر درست نکنید.

ایجاد یک response قابل قبول

پاسخی که در خروجی ترمینال نشان داده شد می‌تواند کمی خسته کننده باشد، به همین دلیل می‌خواهیم تنها به آیتم‌های مهمی که کیفیت بهینه‌سازی کد را نشان می‌دهند اشاره کنیم.

با توجه به شکل ٤-١، داده‌ها به چهار بخش عمده تقسیم می‌شوند که در ادامه، هر بخش را بررسی خواهیم کرد.

- اطلاعات سرور

```
Server Software:        nginx
Server Hostname:        www.keramatifar.ir
Server Port:            80
```

قسمت اطلاعات سرور شامل نرم‌افزاری است که وب‌سرور اجرا می‌کند. در مثال ما این نرم‌افزار nginx است.

به هر روی، این اطلاعات در نخستین بخش، که همان نرم‌افزار سرور (server software) است موجود می‌باشد. مقدار این فیلد بسته به نرم‌افزار وب‌سروری که وب‌سایت از آن استفاده می‌کند، می‌تواند تغییر کند. برای نمونه، در وب‌سرورهای دیگر ممکن است این نرم‌افزار apache باشد. همچنین ممکن است به‌دلیل استفاده از شیوه‌های امنیتی توسط مدیران وب‌سایت، مقدار این فیلد برای شما ناآشنا باشد.

دو فیلد بعدی، یعنی Server Hostname و Server Port، شامل نام هاستی که شبیه‌سازی را بر روی آن اجرا کرده‌ایم و پورتی که وب‌سرور روی آن در حال listening است، می‌باشند.

- اطلاعات اسکریپت (Script)

```
Document Path:          /index.php
Document Length:        20551 bytes
```

بخش دوم پاسخ ab شامل اطلاعاتی در مورد سند (document) وب‌سایتی است که شبیه‌سازی بر روی آن اجرا می‌شود .

Document Path شامل صفحه‌ی درخواست شده و Document Length در برگیرنده‌ی مجموع حجم همه‌ی HTMLها، عکس‌ها، جاوا اسکریپت‌ها (JS)، استایل شیت‌ها (CSS) و هر چیز دیگر موجود در پاسخ، بر حسب بایت می‌باشد.

- اطلاعات ارتباط (Connection Information)

```
Concurrency Level:      1
Time taken for tests:   0.983 seconds
Complete requests:      1
Failed requests:        0
Write errors:           0
Total transferred:      20885 bytes
HTML transferred:       20551 bytes
Requests per second:    1.02 [#/sec] (mean)
Time per request:       982.802 [ms] (mean)
Time per request:       982.802 [ms] (mean, across all concurrent requests)
Transfer rate:          20.75 [Kbytes/sec] received
```

این قسمت به پرسش‌هایی مانند زیر پاسخ می‌دهد:

- چقدر طول می‌کشد تا پاسخ یک درخواست داده شود؟
- چقدر داده برگردانده شد؟
- و از همه مهم‌تر: وب‌سرور هنگام پردازش سند چه تعداد کاربر را می‌تواند پشتیبانی کند؟

جدول ۲-۱ فهرست و توضیحات کاملی از اطلاعات موجود در این بخش را ارائه می‌کند.

جدول ۲-۱: توضیحات پاسخ ab

عنوان	توضیحات	مثالی از مقدار
Concurrency Level	تعداد کل Requestهای ارسال شده به‌صورت همزمان.	۱و۲و۳وN، یا هر عدد دلخواه دیگری که خودمان تعیین کردیم.
Time taken for tests	کل زمان اجرا	۹۸۳ ثانیه
Complete requests	تعداد کل Requestهای کامل شده از کل Requestهای ارسالی.	۱و۲و۳وN، یا هر عدد دلخواه دیگری که خودمان تعیین کردیم.
Failed requests	تعداد Requestهایی که fail شده‌اند، از تعداد کل requestها.	۱و۲و۳وN، یا هر عدد دلخواه دیگری که خودمان تعیین کردیم.
Write errors	تعداد کل خطاهایی که هنگام عملیات نوشتن روی داده است.	۱و۲و۳وN، یا هر عدد دلخواه دیگری که خودمان تعیین کردیم.
Non-2xx responses	تعداد Requestهایی که پاسخ موفقیت‌آمیز (۲۰۰) از HTTP دریافت نکرده‌اند.	۱و۲و۳وN، یا هر عدد دلخواه دیگری که خودمان تعیین کردیم.

عنوان	توضیحات	مثالی از مقدار
Total transferred	حجم کل اطلاعات پاسخ، **برای** کل عملیات شبیه‌سازی–حجم شامل داده‌های Header نیز می‌شود.	۲۰۸۸۵ بایت
HTML transferred	حجم کل محتوایی که برای عملیات شبیه‌سازی ارسال شده است	۲۰۵۵۱ بایت
Requests per second	تعداد کل Requestهای پشتیبانی شده بر ثانیه.	۱.۰۲ [#/sec] (mean)
Time per request	کل زمانی که هر یک از Requestها برای دریافت اطلاعات طی می‌کنند.	۹۸۲.۸۰۲ms (mean)
Time per request	کل زمانی که هر یک از Requestها برای دریافت اطلاعات طی می‌کنند، در مقابل زمان کل Requestها.	۹۸۲.۸۰۲ms (mean)
Transfer rate	حجم کل اطلاعات دریافت شده بر ثانیه، بر حسب کیلوبایت.	۲۰.۷۵ KB/Sec

HTML transferred، Requests per second و Time per request فیلدهای کلیدی برای ما محسوب می‌شوند. این فیلدها مقدار داده‌ای که وب‌سرور در پاسخ به یک درخواست خاص برگردانده است، تعداد درخواست‌هایی که وب‌سرور در هر ثانیه می‌تواند handle کند و کل زمان سپری شده از لحظه‌ی ارسال درخواست و دریافت پاسخ از سوی وب‌سرور را مشخص می‌کنند.

هدف ما این است که HTML transferred را کم کنیم، request per second (تعداد درخواست در هر ثانیه) را زیاد کنیم و Time per request (مدت زمان پاسخ برای هر درخواست) را کاهش دهیم.

- متریک‌های خرابی ارتباط

آخرین بخش نیز شامل جدولی با فیلدهای waiting، processing، connect و total می‌باشد. این فیلدها مدت زمان پاسخ‌گویی به یک درخواست را، در حین اجرای هر یک از این فرایندها به ما می‌دهد. ما بیشتر با فیلد total و ستون‌های min و max آن سرو کار داریم که این ستون‌ها، کمترین و بیشترین مدت زمانی را که طول می‌کشد تا به یک درخواست پاسخ داده شود، نشان می‌دهند.

Flagهای اختیاری ab

ab تعدادی گزینه‌ی اختیاری مفید دارد که به شما اجازه می‌دهد پاسخ را درون جدول‌های HTML قالب‌بندی کنید، cookie ست کنید، اطلاعات basic authentication و نوع محتوا (content type) را از میان گزینه‌های دیگر ست کنید. فهرست کامل گزینه‌های اختیاری ab در جدول ۳-۱ نشان داده شده است:

جدول ۳-۱

Flag	توضیحات
-A <username>:<password>	برای تأمین اطلاعات مربوط به احراز هویت سرور به کار می‌رود. Username و password توسط ":" از یکدیگر جدا می‌شوند. رشته‌ی فرستاده شده در مبنای ٦٤ رمزنگاری می‌شود.
-c <concurrency number>	تعداد درخواست‌ها برای شبیه‌سازی در یک زمان. ۱ به صورت پیش فرض ست می‌شود. عدد مورد نظر نمی‌تواند بیشتر از n باشد.
-C Cookie-name=value	فلگ قابل تکرار که شامل اطلاعات کوکی می‌باشد.
-d	"درصد به کار رفته در جدول [ms]XX" را پنهان می‌کند.
-e	مسیر فایل csv. را ایجاد می‌کند. این فایل شامل نتایج مربوط به اجرای تست کارایی (benchmark) است که به دو ستون درصد و زمان بر حسب ثانیه تقسیم می‌شود. بیش از فایل "gnuplot" توصیه می‌شود.
-g	مسیر فایل "gnuplot" یا TSV را ایجاد می‌کند. خروجی benchmark در این فایل ذخیره خواهد شد.
-h	لیستی از گزینه‌های موجود برای استفاده از ab را نشان می‌دهد.
-H custom-header	هدرهای معتبر سفارشی را به همراه درخواست در قالب یک جفت field-value ارسال می‌کند.
-i	یک درخواست HEAD به جای درخواست پیش فرضGET اجرا

Flag	توضیحات
	می‌کند.
-k	ویژگی Keep-Alive را فعال می‌کند. چندین درخواست را با یک HTTP session پاسخ می‌دهد. این ویژگی به طور پیش فرض غیر فعال است.
-n requests	کل تعداد درخواست‌هایی که باید اجرا شوند.
-p POST-file	مسیر به فایلی که شامل اطلاعات استفاده شده برای یک درخواست HTTP POST می‌باشد. محتوا باید شامل key=value باشد که با & از یکدیگر جدا می‌شوند.
-P username:password	رشته‌ی رمزنگاری شده بر مبنای ٦٤ است. این رشته شامل اعتبار اولیه، نام کاربری و رمز عبور است که با : از هم جدا شده‌اند.
-q	زمانیکه بیش از ۱۰۰ درخواست اجرا می‌شود، پیشرفت خروجی را پنهان می‌کند.
-s	از یک پروتکل HTTPS به جای پروتکل پیش فرض HTTP استفاده می کند – توصیه نمی‌شود.
-S	مقادیر انحرافی استاندارد و متوسط را پنهان می‌کند.
-t timelimit	در صورت تعیین شدن، تست benchmark طولانی‌تر از مقدار تعیین شده طول نخواهد کشید. به طور پیش فرض هیچ محدودیت زمانی وجود ندارد.
-v verbosity-level	مقدار عددی: مقدار ۲ و بالاتر، warning و info را چاپ خواهد کرد؛ ۳ کدهای HTTP response را چاپ خواهد کرد. ٤ و بالاتر اطلاعات هدر را چاپ خواهد کرد.
-V	نسخه‌ی ابزار ab را نشان می دهد.
-w	نتایج را در یک جدول HTML چاپ خواهد کرد.
-x	رشته‌ای که نشان‌دهنده‌ی attributeهای HTML است که درون تگ

Flag	توضیحات
<table-attributes>	<table> زمانی که w– استفاده می‌شود قرار می‌گیرند.
-X proxy[:port]	یک proxy server با پورت دلخواه را مشخص می‌کند.
-y <tr-attributes>	رشته‌ای که نشان‌دهنده‌ی attributeهای HTML است که درون تگ <tr> زمانی که w– استفاده می‌شود قرار می‌گیرند.
-z <td-attributes>	رشته‌ای که نشان‌دهنده‌ی attributeهای HTML است که درون تگ <td> زمانی که w– استفاده می‌شود قرار می‌گیرند.

برای بهینه‌سازی PHP script بهتر است بر روی تنها چند گزینه‌ای که در زیر آمده است تمرکز کنیم:

- **n**: تعداد درخواست‌های شبیه‌سازی شده
- **c**: تعداد درخواست‌های شبیه‌سازی شده‌ی همزمان
- **t**: مدت زمان اجرای شبیه‌سازی

در دستور پیشین، از flag n استفاده کردیم، اکنون می‌خواهیم از آپشن‌های دیگری استفاده کنیم تا ببینیم چه تغییری در کارایی ایجاد می‌شود.

تست‌های همزمانی

مدت زمانی که یک کاربر بر روی برنامه سپری می‌کند، بسته به web application می‌تواند از چند ثانیه تا چند دقیقه متغیر باشد. جریان کاربران ورودی، می‌تواند از مقدار بسیار کم تا حجم بسیار زیاد ترافیک، نوسان داشته باشد، چه به‌صورت گذری و تنها در حد یک کلمه از وب‌سایت باشد و چه ناشی از حمله‌ی گسترده‌ای همچون DOS attack باشد.

برای سنجش عملکرد وب‌سایت در مقابل حجم‌های گوناگونی از ترافیک، نیاز است که حجم واقعی از ترافیک را بر روی سایت شبیه‌سازی کنیم.

ما می‌خواهیم تست همزمانی را، به‌صورتی که تعداد ۱۰۰ درخواست از ۱۰ تا کاربر همزمان، به سایت ارسال می‌شود، شبیه‌سازی کنیم.

هنگامی که از دو گزینه‌ی c و n در دستور استفاده می‌کنیم، باید در نظر داشته باشیم که مقدار c (تعداد درخواست‌های همزمان) از مقدار n (تعداد کل درخواست‌ها) کوچک‌تر باشد.

برای این کار، دستور زیر را اجرا می‌کنیم:

Ab –n 100 –c 10 http://www.example.com/

پس از اجرای دستور باید پاسخی همانند شکل ٦-١ را دریافت کنید.

```
c:\xampp\apache\bin>ab -n 100 -c 10 http://www.keramatifar.ir/index.php
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking www.keramatifar.ir (be patient).....done


Server Software:        nginx
Server Hostname:        www.keramatifar.ir
Server Port:            80

Document Path:          /index.php
Document Length:        20551 bytes

Concurrency Level:      10
Time taken for tests:   32.720 seconds
Complete requests:      100
Failed requests:        0
Write errors:           0
Total transferred:      2088500 bytes
HTML transferred:       2055100 bytes
Requests per second:    3.06 [#/sec] (mean)
Time per request:       3271.987 [ms] (mean)
Time per request:       327.199 [ms] (mean, across all concurrent requests)
Transfer rate:          62.33 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:      294  319  37.5    303     460
Processing:   723 2748 443.0   2821    3140
Waiting:      350 1288 606.5   1207    2795
Total:       1127 3068 435.1   3145    3442

Percentage of the requests served within a certain time (ms)
  50%   3145
  66%   3208
  75%   3227
  80%   3235
  90%   3348
  95%   3380
  98%   3391
  99%   3442
 100%   3442 (longest request)

c:\xampp\apache\bin>
```

شکل ۶-۱

با توجه به شکل بالا، اگر به فیلد Request per second نگاه کنیم، متوجه می‌شویم که وب‌سرور می‌تواند ٣.٠٦ درخواست (کاربر) را در هر ثانیه ساپورت کند.

با آنالیز متریک‌های ارتباط و ستون‌های max و min از فیلد total به این نتیجه می‌رسیم که سریع‌ترین پاسخ‌گویی ٤٦٠ میلی ثانیه بوده است؛ درحالی که کندترین درخواست پاسخ داده شده، ٣٤٤٢ میلی‌ثانیه طول کشیده است.

اما می‌دانیم که حجم بالای ترافیک می‌تواند برای چند دقیقه، چند ساعت و یا حتی چند روز باقی بماند. بیایید برای اینکه این موضوع را تست کنیم یک simulation یا شبیه‌سازی را اجرا کنیم.

تست‌های زمانی

شاید متوجه شوید هر روز، نزدیک ظهر وب‌سایت‌تان به مدت ده دقیقه با افزایش ترافیک روبه‌رو می‌شود. در این وضعیت وب‌سرور شما چگونه عمل می‌کند؟

گزینه‌ی بعدی که می‌خواهیم استفاده کنیم t است. فلگ t به شما اجازه می‌دهد که عملکرد وب‌سایت خود را در هر بازه‌ی زمانی بررسی کنید.

دستور زیر ۱۰ درخواست همزمان را در بازه‌ی زمانی ۲۰ ثانیه شبیه‌سازی می‌کند:

```
ab –c 10 –t 20 http://www.example.com
```

با اینکه این دستور شامل فلگ n نمی‌باشد، اما به‌صورت پیش‌فرض زمانی که از گزینه‌ی t استفاده می‌کنیم، توسط ab مقدار ۵۰۰۰۰ را به خود می‌گیرد. در برخی موارد زمانی‌که از گزینه‌ی t در دستور استفاده می‌کنیم تعداد درخواست‌ها می‌تواند به مقدار ۵۰۰۰۰ برسد که در این حالت simulation به پایان می‌رسد.

با اجرای دستور بالا، خروجی همانند شکل ۷-۱ را خواهید داشت.

```
c:\xampp\apache\bin>ab -c 10 -t 20 http://www.keramatifar.ir/index.php
This is ApacheBench, Version 2.3 <$Revision: 655654 $>
Copyright 1996 Adam Twiss, Zeus Technology Ltd, http://www.zeustech.net/
Licensed to The Apache Software Foundation, http://www.apache.org/

Benchmarking www.keramatifar.ir (be patient)
Finished 55 requests


Server Software:        nginx
Server Hostname:        www.keramatifar.ir
Server Port:            80

Document Path:          /index.php
Document Length:        20551 bytes

Concurrency Level:      10
Time taken for tests:   20.389 seconds
Complete requests:      55
Failed requests:        0
Write errors:           0
Total transferred:      1181443 bytes
HTML transferred:       1162405 bytes
Requests per second:    2.70 [#/sec] (mean)
Time per request:       3707.134 [ms] (mean)
Time per request:       370.713 [ms] (mean, across all concurrent requests)
Transfer rate:          56.59 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:      296  307  19.2    296     421
Processing:  2699 2805  84.7   2777    3120
Waiting:      437 1427 657.9   1498    2683
Total:       2995 3112  85.5   3081    3432

Percentage of the requests served within a certain time (ms)
  50%   3073
  66%   3104
  75%   3167
  80%   3182
  90%   3198
  95%   3292
  98%   3416
  99%   3432
 100%   3432 (longest request)

c:\xampp\apache\bin>
```

شکل ۷-۱: خروجی benchmark برای ۱۰ کاربر همزمان در مدت زمان ۲۰ ثانیه

نتیجه‌ی این شبیه‌سازی به این اشاره می‌کند که زمانی که ده کاربر به‌طور همزمان ظرف مدت ۲۰ ثانیه، درخواست ارسال می‌کنند، کارایی کاهش می‌یابد. سریع‌ترین پاسخ‌گویی به درخواست ۲۹۹۵ میلی‌ثانیه و طولانی‌ترین آن ۳٤۳۲ میلی‌ثانیه، طول کشیده است.

پیچ و خم‌های ab

هنگام استفاده از ab چندین اخطار وجود دارد. اگر به دستور اجرا شده نگاهی بیاندازید، می‌بینید که در انتهای اسم دامین، یک نماد / درج کرده‌ایم. هنگامی که سند خاصی از وب‌سایت مد نظر نمی‌باشد، استفاده از این نماد، لازم است. همچنین، گاهی ممکن است ab توسط وب‌سرور بلاک شده باشد که در این مورد، هیچ داده‌ای دریافت نمی‌کنید. برای حل این مشکل، در دستور ab از گزینه‌ی H- استفاده می‌کنیم، به این ترتیب اطلاعات موجود در هدر Request ارسالی را، سفارشی می‌کنیم تا برای وب‌سرور قابل شناسایی بوده و به ab اجازه‌ی اجرا دهد.

مثلا اگر درخواست خود را از طریق مرورگر Chrome ارسال کنیم می‌توانیم از دستور ab زیر استفاده کنیم:

```
ab -n 100 -c 5 -H "Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US) AppleWeb
Kit/534.2 (KHTML, like Gecko) Chrome/6.0.447.0 Safari/534.2"
http://www.example.com
```

Siege

دومین ابزاری که برای تست کارایی استفاده می‌کنیم Seige می‌باشد که مانند ab به شما امکان شبیه‌سازی بار ترافیکی بر روی وب‌سرور را می‌دهد اما بر خلاف ab می‌توانید این شبیه‌سازی را بر روی فهرستی از URLهای خاص که در یک فایل text مشخص کرده‌اید، اجرا کنید.

همچنین به شما اجازه می‌دهد یک درخواست را پیش از اجرای درخواست دیگر، sleep کنید؛ یعنی کاربر پیش از رفتن به صفحه‌ی دیگری از web application شما، امکان خواندن صفحات پیشین را داشته باشد.

نصب Seige

به دو روش می‌توان Siege را نصب کرد؛ یا با دانلود کردن سورس کد آن از وب‌سایت‌های رسمی www.joedog.org/index/siege-home یا http://freshmeat.net/projects/siege و یا با استفاده

از port یا aptitude که با دستور sudo port install siege یا sudo aptitude install siege انجام می‌شود.

با استفاده از یکی از این دستورات، Siege به‌طور اتوماتیک همه‌ی پکیج‌های لازم برای اجرا را نصب می‌کند.

متأسفانه کاربران ویندوز بدون کمک Cygwin قادر به استفاده از Siege نیستند. اگر از ویندوز استفاده می‌کنید، نخست Cygwin را دانلود و نصب کنید و پس از آن آغاز به نصب Siege کنید.

اگر تصمیم به نصب برنامه از طریق source دارید ممکن است با دانلود پکیج مشکل داشته باشید که در این صورت terminal را باز و دستور زیر را تایپ کنید:

Wget ftp://ftp.joedog.org/pub/siege/siege-latest.tar.gz

این دستور کل پکیج را بر روی سیستم دانلود می‌کند. پس از اینکه دانلود پکیج به پایان رسید دستورهای زیر را اجرا کنید:

- tar xvfz siege-latest.tar.gz
- /cd siege-2.69
- ./configure
- make
- sudo make install

این دستورها، source را کانفیگ خواهند کرد، پکیج نصب می‌سازند و در آخر این پکیج را بر روی سیستم شما نصب می‌کنند. به محض اینکه پکیج نصب شد، دایرکتوری خود را به /usr/local/bin تغییر دهید. باید Siege script را در این دایرکتوری ببینید.

اکنون بیایید یک تست ساده روی دامین http://www.example.com انجام دهیم تا نتیجه را ببینیم.

اجرای Siege

نخستین مثال ما یک تست لود ساده بر روی دامین http://www.example.com است. Siege هم مانند ab از یک فرمت syntax خاص پیروی می‌کند:

siege [options] [URL]

با استفاده از siege یک تست لود با ۵ کاربر همزمان و برای ۱۰ ثانیه را بر روی وب‌سایت http://www.example.com شبیه‌سازی می‌کنیم. به‌عنوان یک نکته باید گفت که مفهوم همزمانی[1] در siege، با نام transaction (تراکنش) بیان می‌شود.

بنابراین تستی که شبیه‌سازی خواهیم کرد دارای یک وب‌سرور است که ۵ تراکنش همزمان را در مدت زمان ۱۰ ثانیه با استفاده از دستور Siege پاسخ‌گو می‌باشد.

```
siege –c 5 –t10S http://www.example.com/
```

دستور بالا از دو گزینه‌ی اختیاری استفاده می‌کند: c flag مختص همزمانی و t flag برای زمان.

Flag همزمانی امکان تست یک درخواست برای X کاربر که به‌طور همزمان از سایت بازدید می‌کنند را فراهم می‌کند. X می‌تواند هر عدد دلخواهی باشد تا زمانی که سیستم، اجرای آن را پشتیبانی کند.

Flag زمان(t) هم، مدت زمان پاسخ‌گویی را چه بر حسب ثانیه (S)، چه دقیقه (M) و چه ساعت (H) نشان می‌دهد و نباید هیچ فاصله‌ای بین عدد و حرف در دستور باشد.

هنگامی که دستور اجرا می‌شود باید خروجی همانند شکل ۸–۱ را ببینید.

```
Lifting the server siege...      done.
Transactions:                    102 hits
Availability:                 100.00 %
Elapsed time:                   9.71 secs
Data transferred:               0.04 MB
Response time:                  0.02 secs
Transaction rate:              10.50 trans/sec
Throughput:                     0.00 MB/sec
Concurrency:                    0.24
Successful transactions:         102
Failed transactions:               0
Longest transaction:            0.03
Shortest transaction:           0.02
```

شکل ۸–۱: پاسخ siege روی www.example.com با ۵ درخواست همزمان ظرف ۱۰ ثانیه

بررسی نتایج

نتایج ابزار siege به دو بخش تقسیم می‌شود:

- جزئیات درخواست (Individual request details)
- متریک‌های تست

---

[1] concurrency

الف- جزئیات درخواست

بخش جزئیات درخواست، تمام درخواست‌های تولید و اجرا شده به‌وسیله‌ی ابزار تست را نشان می‌دهد. هر خط، نشان‌دهنده‌ی یک دستور خاص است و شامل سه ستون است که در شکل ۹-۱ نشان داده شده است.

```
** SIEGE 2.69
** Preparing 5 concurrent users for battle.
The server is now under siege...
HTTP/1.1 200   0.03 secs:      438 bytes ==> /
HTTP/1.1 200   0.03 secs:      438 bytes ==> /
HTTP/1.1 200   0.02 secs:      438 bytes ==> /
HTTP/1.1 200   0.03 secs:      438 bytes ==> /
```

شکل ۹-۱

ستون‌های بالا موارد زیر را بیان می‌کنند:

- کد وضعیت پاسخ HTTP
- مدت زمان کامل شدن یک درخواست
- مقدار داده‌ی دریافت شده به‌عنوان پاسخ (به استثنای داده‌ی موجود در هدر)

ب- متریک‌های تست

جدول ٤-۱ همه‌ی فیلدهای موجود را فهرست و توضیح می‌دهد. ما تنها فیلدهای Longest transaction، Transaction rate، Data transferred و Shortest transaction را مورد بررسی قرار می‌دهیم و تنها بر روی این خصیصه‌های (attribute) خاص تمرکز می‌کنیم؛ زیرا اینها شاخصی برای نشان دادن چگونگی بهینه‌سازی وب‌سایت ما می‌باشند.

نام فیلد	توضیحات	مثالی از مقدار
Transactions (تراکنش‌ها)	تعداد کل transactionهای کامل شده	۱۰۲ بار
Availibility (قابلیت دسترسی)	مقدار زمان ممکن برای ارسال درخواست به یک صفحه‌ی تحت وب	٪۱۰۰
Elapsed Time (مدت زمان سپری شده)	کل مدت زمانی که صرف می‌شود تا تست کامل شود	۹.۷۱ ثانیه
Data Transferred (داده‌ی منتقل شده)	کل داده‌ی منتقل شده از طریق پاسخ- به جز داده‌ی موجود در هدر	0.0.4 M

نام فیلد	توضیحات	مثالی از مقدار
Response Time (زمان پاسخ‌گویی)	متوسط زمان پاسخ‌گویی نسبت به کل تست	۰.۰۲ ثانیه
Transaction Rate (سرعت تراکنش)	تعداد تراکنش‌های پاسخ داده شده در هر ثانیه	۱۰.۵۰ تراکنش بر ثانیه
Throughput (توان عملیاتی)	کل زمان صرف شده برای پردازش داده و پاسخ	00.0 M/S
Concurrency (همزمانی)	متوسط تعداد اتصالات همزمان است که با کاهش کارایی سرور افزایش می‌یابد.	۵
Successful Transactions (تراکنش‌های موفق)	تعداد تراکنش‌های اجرا شده‌ی موفق در طول تست	۱۰۲
Failed Transactions (تراکنش‌های ناموفق)	تعداد تراکنش‌های اجرا شده‌ی ناموفق در طول تست	۰
Longest Transactions (طولانی‌ترین تراکنش)	طولانی‌ترین بازه‌ی زمانی سپری شده برای پاسخ به یک درخواست	۰.۰۳
Shortest Transactions (کوتاه‌ترین تراکنش)	کوتاه‌ترین بازه‌ی زمانی سپری شده برای پاسخ به یک درخواست	۰.۰۲

قسمت Data transferred حجم داده‌ای که هر درخواست به‌عنوان پاسخ دریافت می‌کند را با مقیاس مگابایت بر می‌گرداند.

Transaction rate تعداد تراکنش‌های همزمانی را که وب‌سرور در زیر بار می‌تواند سرویس‌دهی کند، برمی‌گرداند. در این مورد، هنگامی که ۵ درخواست همزمان در مدت زمان ۱۰ ثانیه روی وب‌سرور اجرا می‌شوند، وب‌سرور می‌تواند ۱۰.۵۰ تراکنش را در هر ثانیه سرویس‌دهی کند.

فیلدهای Shortest transaction و Longest transaction، کمترین و بیشترین مدت زمان پاسخ‌گویی به یک درخواست را بر حسب ثانیه نشان می‌دهند.

فلگ‌های اختیاری Siege

Siege همچنین شامل تعداد بسیاری فلگ‌های اختیاری است که با دستور زیر می‌توان به هر یک از آنها دسترسی داشت:

Siege –h

تست URLهای بسیار

بیایید روی دو flag جدید تمرکز کنیم: internet flag(i) و file flag(f)

هنگامی که از flagهای t و i استفاده می‌کنیم به siege این اجازه را می‌دهیم که به‌صورت تصادفی یک URL را از فایل متنی انتخاب کرده و یک درخواست به سمت آن وب‌سایت ارسال کند. هرچند، هیچ تضمینی وجود ندارد که همه‌ی URLهای موجود در فایل متنی، بازدید شوند؛ اما این ضمانت را می‌کند که یک تست واقعی انجام داده‌اید.

برای استفاده از یک فایل مشخص، از flag f استفاده می‌کنیم. فایلی که siege به صورت پیش فرض استفاده می‌کند در آدرس SIEGE_HOME/etc/urls.txt قرار دارد؛ اما می‌توان این مسیر را با دادن آدرس فایل متنی مورد نظر خود به flag، تغییر داد.

فرمت URL و فایل (URL Format and File)

اکنون می‌خواهیم تست بعدی را با اجرای دو دستور انجام دهیم. برای این کار، یک فایل آزمایشی بر روی سیستم خود ایجاد کنید. من فایل خود را در زیر مسیر HOME_DIR/urls.txt ایجاد کردم و سه URL همسان با فرمت نشان داده شده در لیست ۱–۱ قرار داده‌ام. نمونه‌ی کامل فایل urls.txt در لیست ۲–۱ نشان داده شده است.

لیست ۱–۱: ساختار فرمت url در siege

```
[protocol://] [servername.domain.xxx] [:portnumber] [/directory/file]
```

لیست ۲–۱: فایل urls.txt

```
http://www.example.com
http://www.example.org
http://www.example.net
```

urlهای بالا در سه دامین گوناگون هستند که معمولا به این شکل نمی‌خواهیم و اغلب لیستی از web documentهایی را که در یک دامین یکسان هستند، درخواست می‌کنیم. اکنون با استفاده از دستور زیر تست را اجرا می‌کنیم:

```
siege –c 5 –t10S –i –f HOME_DIR/urls.txt
```

همان‌گونه که می‌بینید خروجی، شباهت زیادی به شکل ۸-۱ دارد؛ با این تفاوت که urlهایی که تست شده‌اند به‌صورت رندوم از فایل urls.txt انتخاب شده‌اند.

اکنون هر دوی ابزار ab و siege را اجرا کرده‌ایم و مایلیم بدانیم این اعداد چه تأثیری دارند.

تأثیر خصوصیات تست کارایی

پنج عامل عمده وجود دارد که بر روی زمان پاسخ‌دهی و تست کارایی تأثیر دارند:

- موقعیت جغرافیایی و مشکلات شبکه
- حجم پاسخ
- پردازش کد
- رفتار مرورگر (browser behaviour)
- تنظیمات وب‌سرور

الف. موقعیت جغرافیایی

موقعیت جغرافیایی وب‌سرور، نقش مهمی در مدت زمان پاسخ‌گویی به یک درخواست دارد. اگر وب‌سرور شما در آمریکا باشد و کاربرانتان از کشورهایی همچون چین، اروپا یا آمریکای لاتین درخواست خود را بفرستند، مسافتی که درخواست باید طی کند تا به مقصد برسد، منتظر وب‌سرور بماند تا document خواسته شده را fetch کند و سپس دوباره به مبدأ برگردد، همه‌ی این موارد بر سرعت برنامه‌ی تحت وب تأثیر خواهد گذاشت.

موضوع بر سر تعداد کل روترها، سرورها و در برخی موارد اقیانوس‌هایی است که درخواست باید پشت سر بگذارد تا به مقصد برسد. هرچه تعداد روترها/سرورهایی که کاربران شما باید پشت سر بگذارند بیشتر باشد، مسافتی که درخواست باید طی کند تا به مقصد برسد طولانی‌تر و به همان نسبت مدت زمانی که طول می‌کشد تا وب‌سرور به درخواست پاسخ دهد هم بیشتر است.

ب. بسته‌های در حال جابه‌جایی (The traveling Packets)

بسته‌ها در برخی موارد متحمل هزینه می‌شوند. همان‌گونه که پیش‌تر گفته شد، زمانی که پاسخ یک وب‌سرور در قالب بسته‌های داده به سمت کاربر برگردانده می‌شود، کامپیوتر کاربر پیش از بازسازی پیام چک می‌کند که این بسته‌ها error نداشته باشند. اگر یکی از بسته‌ها error داشته باشد

یک درخواست اتوماتیک به وب‌سرور فرستاده می‌شود تا دوباره همه‌ی بسته‌ها را ارسال کند که این کار شما را وا می‌دارد تا به حجم داده‌ی خود بیاندیشید. چرا که هرچه حجم داده کمتر باشد تعداد بسته‌هایی که وب‌سرور باید دوباره ایجاد و برای کاربر ارسال کند نیز کمتر است.

ج. حجم پاسخ

بیایید امتحان کنیم که چگونه حجم داده می‌تواند بر مدت زمانی که طول می‌کشد تا یک داده به مقصد برسد تأثیر بگذارد. بسته به سرعت کانکشن کاربر، هر چه حجم داده کمتر باشد سرعت پاسخ‌گویی به آن بیشتر است.

برای نشان دادن این نکته، دو درخواست را که یکی برای یک عکس با حجم بالا و دیگری با حجم کم می‌باشند benchmark می‌کنیم تا زمان پاسخ‌گویی به آنها را با هم مقایسه کنیم.

دستور ab برای fetch کردن عکس با حجم بالا:

```
ab -n 1 http://farm5.static.flickr.com/4011/4225950442_864042b26a_s.jpg
```

دستور ab برای fetch کردن عکس با حجم پایین:

```
ab -n 1 http://farm5.static.flickr.com/4011/4225950442_864042b26a_b.jpg
```

هنگامی که اطلاعات response نشان داده شده در شکل‌های ۱۰-۱ و ۱۱-۱ را آنالیز می‌کنیم سه آیتم زیر، برجسته‌تر هستند:

- The document Lenght (طول سند)
- the Total min time (کمترین مدت زمان پاسخ‌گویی)
- the Total max time (بیشترین مدت زمان پاسخ‌گویی)

با توجه به مقادیر total max و total min به این نتیجه می‌رسیم که هرچه حجم داده‌ی درخواستی کمتر باشد، پاسخ‌گویی سریع‌تر است.

```
Document Path:          /4011/4225950442_864042b26a_s.jpg
Document Length:        3407 bytes

Concurrency Level:      1
Time taken for tests:   0.188 seconds
Complete requests:      1
Failed requests:        0
Write errors:           0
Total transferred:      3906 bytes
HTML transferred:       3407 bytes
Requests per second:    5.33 [#/sec] (mean)
Time per request:       187.500 [ms] (mean)
Time per request:       187.500 [ms] (mean, across all concurrent requests)
Transfer rate:          20.34 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       78   78   0.0     78      78
Processing:    94   94   0.0     94      94
Waiting:       94   94   0.0     94      94
Total:        172  172   0.0    172     172
```

شکل ۱۰-۱: پاسخ به درخواست برای عکسی با حجم کم

```
Document Path:          /4011/4225950442_864042b26a_b.jpg
Document Length:        308235 bytes

Concurrency Level:      1
Time taken for tests:   1.109 seconds
Complete requests:      1
Failed requests:        0
Write errors:           0
Total transferred:      308731 bytes
HTML transferred:       308235 bytes
Requests per second:    0.90 [#/sec] (mean)
Time per request:       1109.375 [ms] (mean)
Time per request:       1109.375 [ms] (mean, across all concurrent requests)
Transfer rate:          271.77 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       63   63   0.0     63      63
Processing:  1047 1047   0.0   1047    1047
Waiting:       94   94   0.0     94      94
Total:       1109 1109   0.0   1109    1109
```

شکل ۱۱-۱: پاسخ به درخواست برای عکسی با حجم زیاد

د. پردازش کد

منطقی که یک document پیاده‌سازی می‌کند نیز، بر روی پاسخ‌دهی تأثیر می‌گذارد. در تست‌های آغازین، چون یک صفحه‌ی خیلی ساده و استاتیک HTML را تست می‌کردیم، این موضوع زیاد به چشم نمی‌آمد، اما به محض اینکه مواردی چون PHP، دیتابیس جهت تعامل با برنامه و یک سری وب‌سرویس اضافه می‌کنیم، ناخواسته زمان پاسخ‌گویی به درخواست را افزایش می‌دهیم زیرا هر فعل و انفعال خارجی و پردازش PHP هزینه دارد. در فصل‌های آینده یاد خواهیم گرفت که چگونه این هزینه‌ها را کم کنیم.

ح. رفتار مرورگر

مرورگرها هم در روند پاسخ‌دهی یک وب‌سایت به کاربر، نقش دارند. هر مرورگر، روش خاص خود را برای رندر کردن javaScript، CSS و HTML دارد که می‌تواند موجب افزایش زمان پاسخ‌گویی در حد چند میلی‌ثانیه یا حتی ثانیه شود.

و. تنظیمات وب‌سرور

در آخر، تنظیمات وب‌سرور هم می‌تواند بر زمان پاسخ‌گویی تأثیر بگذارد. در حالت پیش‌فرض، بیشتر وب‌سرورها به‌صورت بهینه تنظیم نشده‌اند و به مهندسان ماهر و خبره نیاز دارند تا تنظیمات فایل‌ها و هسته را اصلاح (modify) کنند. وب‌سرور را زمانی‌که Keep-Alive آن فعال است تست

می‌کنیم تا ببینیم تنظیمات وب‌سرور چگونه بر افزایش کارایی آن تأثیر می‌گذارد. البته جزئیات این تست را در فصل بعدی خواهیم آموخت.

فعال بودن Keep-Alive به وب‌سرور اجازه می‌دهد که به جای اینکه برای هر درخواست، یک کانکشن ایجاد کند و به محض پاسخ‌گویی به درخواست آن را ببندد، یک کانکشن خاص برای چندین درخواست ایجاد کند و بتواند باز باشد تا به درخواست‌های ورودی که بعدا اضافه می‌شوند هم پاسخ دهد. با این کار سرعت برنامه افزایش می‌یابد و مقدار پردازشی که وب‌سرور باید انجام دهد کاهش می‌یابد. درنتیجه تعداد درخواست‌هایی که می‌توان پشتیبانی کرد نیز افزایش می‌یابد.

برای بهتر متوجه شدن تأثیر فعال بودن Keep-Alive، دو دستور زیر را با هم مقایسه می‌کنیم:

```
ab –c 5 –t 10 http://www.example.com
ab –c 5 –t 10 –k http://www. example.com
```

دستوری که دارای k- می‌باشد یعنی Keep-Alive در آن فعال است.

این flag به وب‌سرور اجازه می‌دهد که پنج کانکشن همزمان را باز نگه دارد و دیگر درخواست‌ها (connectionها) را به سمت این پنج کانکشن هدایت کند و از طریق آنها به درخواست‌ها پاسخ دهد که این کار، زمانی که وب‌سرور نیاز دارد تا یک کانکشن جدید ایجاد کند را بسیار کاهش می‌دهد.

شکل‌های ۱۲-۱ و ۱۳-۱ این دو دستور را با هم مقایسه می‌کنند:

```
Concurrency Level:      5
Time taken for tests:   10.063 seconds
Complete requests:      184
Failed requests:        0
Write errors:           0
Total transferred:      133400 bytes
HTML transferred:       80592 bytes
Requests per second:    18.29 [#/sec] (mean)
Time per request:       273.438 [ms] (mean)
Time per request:       54.688 [ms] (mean, across all concurrent requests)
Transfer rate:          12.95 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:       16   53 220.9     31    3031
Processing:   125  214 437.3    141    3141
Waiting:       31  142 385.6     94    3141
Total:        156  267 487.7    188    3188
```

شکل ۱۲-۱: نتایج تست ab مربوط به ۵ درخواست همزمان در مدت ۱۰ ثانیه

```
Concurrency Level:      5
Time taken for tests:   10.000 seconds
Complete requests:      1412
Failed requests:        0
Write errors:           0
Keep-Alive requests:    1412
Total transferred:      1023700 bytes
HTML transferred:       618456 bytes
Requests per second:    141.20 [#/sec] (mean)
Time per request:       35.411 [ms] (mean)
Time per request:       7.082 [ms] (mean, across all concurrent requests)
Transfer rate:          99.97 [Kbytes/sec] received

Connection Times (ms)
              min  mean[+/-sd] median   max
Connect:        0    0   2.5      0      47
Processing:    16   35  56.1     31    1234
Waiting:       16   35  56.1     31    1234
Total:         16   35  56.3     31    1234
```

شکل ۱۳-۱: نتایج تست ab با استفاده از keep-alive

با مقایسه‌ی هر دو شکل و در نظرگرفتن آیتم‌های Requests per second، total min و total max می‌بینیم که استفاده از Keep-Alive تعداد درخواست‌هایی که یک وب‌سرور در هر ثانیه می‌تواند پاسخ دهد را به شدت افزایش می‌دهد.

خلاصه‌ی این فصل

هدف از این فصل، آشنا کردن شما با ابزارهای Benchmarking و نشان دادن ویژگی‌های مهم هر یک از آنها برای بهینه‌سازی عملکرد برنامه بود.

استفاده، نصب و آنالیز داده‌ی دو ابزار Apache Benchmark و Siege را آموختید.

با چهار آیتم عمده که بر روی تست کارایی تأثیر دارند آشنا شدید و در آخر، چرخه‌ی زندگی یک درخواست HTTP را فرا گرفتید و اینکه دانستن اینکه در این چرخه چه می‌گذرد، می‌تواند به شما در بهینه‌سازی کمک بسیاری کند.