Fünf professionelle Profiling-Werkzeuge für PHP

Fünf professionelle Profiling-Werkzeuge für PHP

[expand title="mehr lesen…"]

Auf Touren

- Tideways
- New Relic APM
- Blackfire
- Xhprof
- Xdebug
- Profiling-Demo
- Faker
- ab – Apache HTTP server benchmarking tool

Fünf professionelle Profiling-Werkzeuge für PHP

Auf Touren

Jörn Wagner, Christian Ehmke

Ob Java, .NET, Ruby oder PHP – größere Websites sind längst mit normalen Softwareprojekten vergleichbar. Wie bei diesen helfen Testwerkzeuge und Profiler bei der Analyse. Ein Überblick über fünf solcher Profiling-Tools für PHP.

iX-TRACT

Profiling-Werkzeuge sollen bei der Analyse von Software in Bezug auf Laufzeit, Speicherverbrauch, Datenbankabfragen und weitere Parameter helfen.

Fünf für die PHP-Entwicklung gedachte Produkte hat *iX* eine Demo-Anwendung durchlaufen lassen, um ihre Effizienz zu prüfen.

Xdebug und Xhprof (Open Source) sowie Tideways, Blackfire und New Relic (kommerziell) eignen sich für unterschiedliche Szenarien und Geldbeutel.

iX-Wertung

Blackfire

⊕ einfache Installation und Bedienung

⊕ kostenlose Produktvariante, auch dauerhaft

⊕ Kollaboration im Team

⊕ Datenspeicherung in Europa

⊖ Team-Profile nicht öffentlich teilbar

⊖ kein Profiling auf Windows-Plattformen

New Relic

⊕ einfache Installation und Bedienung

⊕ kostenlose Produktvariante, auch dauerhaft

⊖ rudimentäres Profiling

⊖ keine Berücksichtigung nativer PHP-Funktionen

⊖ kein Profiling auf Windows-Plattformen

Tideways

⊕ einfache Installation und Bedienung

⊕ umfangreiche und übersichtliche Analysen

⊕ Berücksichtigung beliebter Frameworks

⊕ Datenspeicherung in Deutschland

⊖ kein Profiling auf Windows-Plattformen

⊖ nach Ablauf des Testzeitraums kostenpflichtig

Xdebug

⊕ kostenlos

⊕ einfache Installation auf eigenem System

⊖ hoher Overhead

⊖ schwierige Bedienung

⊖ Auswertung abhängig von Drittwerkzeugen

Xhprof

⊕ kostenlos

⊕ aussagekräftiges Auswertungswerkzeug mitgeliefert

⊕ Installation auf eigenem System

⊖ Installation mäßig aufwendig

⊖ unübersichtliche Auswertung

⊖ nicht zu vernachlässigender Overhead

Profiler sind wichtige Werkzeuge für jeden Entwickler, mit denen er das Laufzeitverhalten einer Anwendung oder eines Dienstes analysieren und Schwachstellen aufdecken kann. Unter dem Begriff Profiling versteht man die Erstellung eines Profils der Software in Bezug auf Laufzeit, Speicherverbrauch,

Datenbankabfragen und weitere Parameter. Der häufigste Anwendungsfall bezieht sich auf das Zählen und Messen von Aufrufen der beteiligten Funktionen und Methoden. Ziel ist es, die Stellen im Code zu finden, an denen die meiste Zeit verbraucht wird, und diese zu optimieren. Daher ist es wichtig, nach Änderungen einen guten Vergleich ziehen zu können, um Aussagen über die Wirksamkeit der vermeintlichen Verbesserungen treffen zu können.

Im Ökosystem der weitverbreiteten Webprogrammiersprache PHP waren bis vor wenigen Jahren Profiler nicht übermäßig etabliert. Eine routinemäßige Überprüfung des Laufzeitverhaltens einer Anwendung, die Performance-Probleme schon während der Entwicklung hätte aufdecken können, fand oft nicht statt. Meist erzeugte erst ein echter Performance-Engpass in der Produktivumgebung die Notwendigkeit zur Analyse. Dazu war es durchaus üblich, Konstrukte zur Erfassung genauer Zeitstempel wie manuelle Zeitmessungen im Code – gerne mit Stoppuhr-Klassen oder -Bibliotheken – zu nutzen, um den sogenannten Flaschenhals aufzuspüren. Dieses Trial and Error lag womöglich an den wenigen verfügbaren Lösungen zum Erstellen von Profilen mit PHP. Und von denen waren die meisten noch mit viel Overhead bei der Ausführung verbunden und erzeugten schwer verständliche Output-Dateien, sodass eine manuelle Analyse schnellere Erfolge zu verheißen schien, aber bei fortschreitender Komplexität immer mühsamer und ineffizienter wurde.

Mit Blackfire und Tideways sind nun kürzlich gleich zwei neue kommerzielle Profiler für PHP veröffentlicht worden, die sich zu den Open-Source-Lösungen Xdebug und Xhprof sowie der ebenfalls kommerziellen Lösung New Relic gesellen. Grund genug, diese Profiler unter die Lupe zu nehmen.

Profiling-Begriffe erklärt

Profiling kann man grundsätzlich auf zwei Arten durchführen: über statistische Auswertung (Sampling) oder Instrumentierung.

Bei ersterer bleibt der Ablauf des ausgeführten Programms oder Skriptes unverändert, vielmehr hält ein Programm in periodischen Abständen über Betriebssystemaufrufe die Ausführung kurz an und zeichnet Messwerte zum aktuellen Codeblock und Ressourcenverbrauch auf. Diese Vorgehensweise beeinträchtigt die Laufzeit des zu messenden Programms so gut wie gar nicht, kann aber nur stichprobenartige Mittelwerte liefern und insbesondere keinen Call Graph erzeugen (eine hierarchische Aufstellung des Codepfads mit genauen Ausführungszeiten). Daher verwenden viele Profiling-Tools, darunter alle hier vorgestellten, die Methode der Instrumentierung. Dabei werden in den Code Zählaufrufe injiziert, die genauere Messungen ermöglichen. Allerdings leidet darunter in der Folge die Ausführungsgeschwindigkeit, teilweise enorm. Ein guter Profiler sollte keinen zu großen Overhead produzieren, da dieser die Ergebnisse stark verzerrt und beispielsweise Nebenläufigkeitsprobleme gar nicht auffallen oder Timeouts auftreten, die es ohne Profiling nicht gegeben hätte, und er dadurch die Vergleichbarkeit der Messungen verhindert.

Von größtem Interesse ist zunächst die benötigte Zeit (Wall-Clock Time), die auf der Uhr an der Wand von Anfang bis Ende der Programmausführung verstreicht. Daneben kann noch die CPU-Zeit von Interesse sein – die Zeit, in der das Programm aktiv CPU-Ressourcen verbraucht hat, ohne die Sekunden, die zwischenzeitlich mit Warten auf I/O-Ressourcen oder Antwort von anderen Diensten verstrichen sind. Ebenfalls von Belang ist der benötigte Umfang an RAM, vor allem wenn es darum geht, Speicherlecks aufzuspüren.

Bei den Profiling-Ergebnissen wird oft zwischen interner und externer Zeit differenziert, verschiedene Profiler verwenden unterschiedliche Begriffe. Die interne oder exklusive Zeit besagt, wie lange die Funktion oder Methode für ihre Ausführung gebraucht hat. Externe oder inklusive Zeit bezeichnet den Ablauf von Anfang bis Ende der Methode, das

heißt inklusive aller Aufrufe anderer Funktionen. Eine Funktion mit hoher externer, aber geringer interner Zeit ist daher gar nicht selbst der Verursacher des hohen Zeitverbrauchs, sondern von ihr aufgerufene Funktionen, die ihrerseits viel Zeit benötigen.

Demo-Anwendung für den Vergleich

Damit Interessierte die Werkzeuge unter gleichen Bedingungen testen und vergleichen können, haben die Autoren eine Demo-Anwendung mit dem PHP-Framework Symfony der Version 2.7 erstellt, die bei GitHub zur Verfügung steht (die URL ist über den blauen Balken „Alle Links“ zu finden). Mit dem ORM-Framework Doctrine wurden drei Entitäten erzeugt: *Person*, *Calendar* und *Entry* (Kalendereintrag). Faker (ebenfalls über den genannten blauen Balken zu erreichen) diente dazu, 50 Testdatensätze zu Personen mit insgesamt 200 Kalendern und 563 Kalendereinträgen zufällig zu erzeugen und in einer SQLite-Datenbank zu speichern. Dem MVC-Paradigma folgend lädt die Demo alle Personen zunächst in den Controller und übergibt die Daten an die View-Komponente. Dort erzeugt die Template-Engine Twig alle Einträge als HTML, liefert sie an den Controller wieder zurück, und der schickt sie an den Client (Webbrowser).

Zur Verdeutlichung sind bewusste, aber in der Praxis häufig beobachtbare Performance-Fehler eingebaut. Durch die Verwendung des Lazy Loading von Doctrine ORM erzeugt jeder Schleifenaufruf mit dem Zugriff auf die Eigenschaft einer Person mit einer Fremdschlüsselbeziehung eine erneute Datenbankabfrage, ebenso jeder Zugriff auf die Einträge eines Kalenders. Darüber hinaus erzeugt das Programm die Kalendereinträge als Sub-Template und bindet sie mit einer Twig-basierten *include*-Anweisung ein. Zudem ermittelt sie über einen Singleton-Ansatz die Anrede abhängig vom Geschlecht und gibt sie aus, aber durch einen Implementierungsfehler instanziiert sie die Singleton-Klasse bei jedem Aufruf erneut. Das sorgt für hundertfache SQL-Abfragen sowie ineffiziente

Templating-Aufrufe und bietet damit eine brauchbare Grundlage, um auf die Suche nach Performance-Engpässen zu gehen.

Die Demo-Anwendung wird auf einem virtualisierten Server mit zwei CPU-Kernen und 2 GByte RAM betrieben. Als Betriebssystem ist Ubuntu 14.04.3 LTS installiert. Apaches HTTP-Server 2.4.7 dient als Webserver mit PHP 5.5.9 und aktiviertem Zend OPcache in der Version 7.0.3.

Xdebug

Historisch betrachtet das erste Werkzeug zum Aufspüren von Performance-Problemen in PHP-Code ist die Erweiterung Xdebug. Seit 2002 als Open-Source-Software verfügbar, liefert sie dem PHP-Entwickler viele wertvolle Informationen nebst aussagekräftigeren Fehlermeldungen und erlaubt das Debugging per Breakpoints. Zudem kann man den Profiler fallweise über den Request-Parameter *XDEBUG_PROFILE* aktivieren; das Programm erzeugt in diesem Fall Dateien, die mit dem C-Profiling-Werkzeug Cachegrind kompatibel sind. Installiert wird Xdebug als PHP-Extension über das PECL-Repository (PHP Extension Community Library).

QCacheGrind zeigt sich beim Auswerten der Demo-Anwendung auffallend bunt und ist eines der aussagekräftigeren Analysewerkzeuge (Abb. 1).
Die Auswertung erfolgt über eins der grafischen Werkzeuge zur Analyse von Cachegrind-Dateien, wie KCacheGrind für Linux, WinCacheGrind oder QCacheGrind für Windows. Je nach Werkzeug stehen mehr oder weniger intuitive und aussagekräftige Auswertungsmöglichkeiten zur Verfügung, so gibt es nicht für jedes Cachegrind-kompatible Programm eine Call-Graph-Darstellung. Die Analyse verlangt ein Einarbeiten in das jeweilige Visualisierungswerkzeug, aber mit einiger Erfahrung lassen sich die Gründe für die hohe Ausführungszeit der Demo-Anwendung finden.

Xdebug verwendet Instrumentierung zur Codeanalyse. Der größte Nachteil beim Einsatz ist der anfallende Overhead (siehe unten). Schon mit aktivierter Xdebug-Extension sinkt die Performanz rapide ab, beim Erstellen eines Profils beträgt sie nicht einmal mehr ein Viertel des ursprünglichen Wertes. Das erschwert nicht nur das Profiling selbst, an einen Produktiveinsatz ist bei solch dramatischen Einbußen für den laufenden Betrieb der Anwendung nicht zu denken.

Xhprof

2009 veröffentlichte Facebook sein selbst entwickeltes Werkzeug zum Auffinden von Performanzschwachstellen als Open Source. Xhprof unterstützt Sampling sowie Instrumentierung und lässt so dem Anwender die Wahl zwischen geringem Overhead und genauer Messung. Die Möglichkeiten beim Sampling sind allerdings stark eingeschränkt, so ist das Sampling-Intervall fest auf 0,1 Sekunden eingestellt und liefert lediglich den Call-Stack zu den gemessenen Zeitpunkten. Außerdem sind die Flags zum Sammeln von CPU- und Speicherverbrauchsinformationen nicht verfügbar, sodass die einzigen Informationen darin bestehen, wie lange die gesamte Ausführung gedauert hat und in welcher Methode sich die Anwendung zu bestimmten Zeiten befand.

Wesentlich aussagekräftiger sind da schon die Informationen aus der Instrumentierung. Xhprof liefert zur Analyse gleich eine HTML-basierte Oberfläche mit, die die gewonnenen Informationen tabellarisch zeigt und Sortierung ermöglicht. Die Einbindung ist zunächst nicht besonders komfortabel, bis man die ersten Ergebnisse bekommt. Xhprof installieren Entwickler ebenfalls als PHP-Erweiterung (Extension), sie müssen sie aber im Code noch durch Aufrufe aktivieren. Ohne weitere Zusätze liefert Xhprof nur ein Array zurück, das Einträge zu Funktionen und je nach aktivierten Flags zwei bis fünf Messwerten zur Anzahl der Aufrufe, zu verstrichener Gesamtzeit, CPU-Zeit sowie Speicherverbrauch (Durchschnitts-

und Spitzenwert) enthält.

Zur weiteren Verarbeitung müssen PHP-Verantwortliche Bibliotheken aus dem *utils*-Verzeichnis der Xhprof-Installation einbinden und das erstellte Profil wegschreiben. Beim Symfony-Framework besteht der Vorteil darin, dass alle Anfragen über einen zentralen Front-Controller laufen, in den man den Xhprof-Code einfügen oder in dem man sogar einen eigenen Profiling-Front-Controller anlegen kann. Für die Demo haben die Autoren sich für Letzteres entschieden, um den Overhead einfach ermitteln zu können. Dieser ist ohne Profiling-Code vernachlässigbar, allerdings während des Profiling signifikant: Die Ausführungszeit verdoppelt sich.

Von Xhprof generierter Call Graph, deutlich erkennbar die rot eingefärbten Performance-Engpässe (Abb. 2)

Man kann außerdem eine tabellarische Darstellung ansehen, die alle Informationen übersichtlich aufbereitet und Details über jede Funktion per Klick auf ihren Namen liefert. Zunächst ist diese Liste auf die 100 meistaufgerufenen Funktionen bezogen, lässt sich aber beliebig erweitern. Bei installiertem Graphviz-Paket stellt Xhprof die Aufrufhierarchie als übersichtlichen Call Graph dar und färbt kritische Stellen gelb beziehungsweise rot ein. So sieht man auf den ersten Blick die beiden eingebauten Engpässe der Demo-Anwendung, an denen man mit Optimierungen beginnen würde. Xhprof bietet die Option, verschiedene Ausführungsläufe einander gegenüberzustellen und so die Verbesserungen (oder Verschlechterungen) übersichtlich anzusehen. Der sogenannte Diff-Report vergleicht zwei Läufe und listet die als Regressions und Improvements bezeichneten Verluste beziehungsweise Zuwächse an Performanz auf. Besonders interessant ist dabei der Diff-Call-Graph, der die Veränderungen grafisch darstellt.

Tideways

Tideways ist ein kommerzieller Realtime Profiler von der Qafoo GmbH für die PHP-Versionen 5.3 bis 5.6. Pünktlich zur Release von PHP 7 soll Tideways mit der neuen Version umgehen können. Zurzeit unterstützt sie nur Linux-Plattformen. Kontinuierlich sammelt Tideways Informationen zu HTTP- und I/O-Zugriffen, Datenbankabfragen und dem Cache. Diese Daten übermittelt das Werkzeug mit SSL-Verschlüsselung an die bei Qafoo in Berlin stehenden Server. Persönliche Daten der Webseitenbenutzer überträgt die Software laut Tideways nicht. Bei Bedarf können Anwender diese Sicherheitseinstellungen ändern. Qafoo bereitet die Daten auf und stellt sie webbasiert zur weiteren Analyse zur Verfügung.

Nach dem kostenlosen Probezeitraum müssen Kunden einen volumenabhängigen Preis zahlen. Das minimale Paket kostet 40,00 Euro pro Monat. In diesem Paket hält Qafoo die Daten für 24 Stunden vor. Sollen sie länger zur Verfügung stehen, aggregiert Qafoo sie deutlich stärker.

Zu installieren bedeutet, die Tideways-Extension einzubinden und einen Daemon zu installieren. Die Software überwacht anschließend die gesamte Apache-PHP-Umgebung. Das Tideways Commandline Interface kann ein Entwickler nutzen, um einzelne Skripte im Dateisystem oder gezielt das Profiling eines HTTP-Aufrufs auszulösen. Optional kann er eine Chrome Extension nutzen, um die explizite Kontrolle über das Profiling und die damit verbundene Datengenerierung zu behalten. Schließlich landen die Daten auf dem Tideways-Server.

Tideways fasst alle gesammelten Informationen unter einer Transaktion zusammen, die einen spezifischen Programmablauf mit einem Namen identifiziert – mehrere HTTP-Requests an die gleiche URL unter derselben Transaktion. Dies ermöglicht unter anderem einen einfacheren Vergleich von Performance-Engpässen im Zeitablauf und eine Analyse der Gegenmaßnahmen. Den Transaktionsnamen legt eine automatisch ausgeführte

Controller-Action-Kombination fest. Keiner Transaktion zugeordnete Daten fasst die Software im Sammelbecken *default* zusammen. Tideways unterstützt Zend Framework 1 und 2, Symfony2 Framework und Components, Shopware, Oxid, WordPress und Laravel. Weitere sollen folgen.

Standardmäßig führt das Programm für 10 % aller erfolgten HTTP-Requests ein Profiling durch. Diesen Wert kann man über einen Eintrag beispielsweise in *php.ini* individuell konfigurieren. Die kontinuierliche Analyse des Systems ermöglicht es, im Bedarfsfall entsprechende Gegenmaßnahmen zeitnah einzuleiten. Tideways eignet sich folglich durchaus fürs Monitoring.

Die Webanwendung von Tideways (tideways.io) kann Organisationen, Anwendungen und Infrastrukturen logisch verwalten. Sie ist in der Lage, unterschiedliche Infrastrukturen mit derselben Anwendung zu analysieren und zu vergleichen. Beim Betrieb mehrerer PHP-Anwendungen in derselben Umgebung führt die Software alle Profiling-Ergebnisse unter derselben Rubrik auf. Alternativ kann man einen API-Key in der konkreten PHP-Anwendung konfigurieren. Dazu muss jemand den Quellcode der PHP-Anwendung um eine Zeile Quellcode erweitern. Auf der Einstiegsseite erhält der Benutzer eine Kurzübersicht über seine Anwendungen sowie aktuelle Informationen zur Reaktionszeit und aufgetretenen Fehlern. Exceptions und Fatal Errors erkennt das Tool ebenfalls und stellt sie mit Details in einer eigenen Rubrik dar.

Weitere Informationen zu einem Request können Anwender über eine Detailansicht abrufen. Grundsätzliche Indikatoren mit Angaben zu Gesamtdauer, Speicherverbrauch, Anzahl von Datenbankabfragen, Anzahl von HTTP-Requests und Wartezeit für I/O-Operationen zeigt Tideways an prominenter Stelle an. Es gruppiert Informationen nach *slow calls*, *controllers* und *views*, was bei dem verwendeten MVC-Framework Symfony2 sinnvoll ist.

Detaillierte Informationen über den Programmablauf einer Transaktion mit Angabe der einzelnen Funktionsaufrufe, ihre Dauer und Häufigkeit erhält man lediglich bei einem manuell ausgelösten Profiling. Dazu kann entweder die Chrome Extension, das Tideways Commandline Interface oder ein benutzerspezifischer GET-Parameter in der URL dienen.

Erst bei diesem Vorgehen stellt die Software den Programmablauf durch einen Call Graph grafisch dar. Performance-Engpässe sind dort auf Anhieb erkennbar. Zusätzlich stehen für jeden Funktionsaufruf weitere Informationen zur Verfügung, etwa die Ausführungszeit (Wall Time) einer Funktion inklusive und exklusive der Zeit von Kinderfunktionen. Auch den Aufruf nativer PHP-Funktionen im Quellcode bietet die Software in der Übersicht. New Relic APM dagegen zeigt native PHP-Funktionen nicht an.

Vollständige SQL-Querys stellt Tideways zurzeit noch nicht dar, aber schon bald soll eine neue Version des Tools zur Verfügung stehen, die die vollständigen SQL-Querys anzeigt und PHP 7 unterstützt.

Blackfire

SensioLabs, die Macher des Symfony Frameworks, haben Ende Juli 2015 einen Profiler namens Blackfire in der Version 1.0 veröffentlicht. Das Werkzeug besteht aus mehreren Komponenten: Der Probe genannte Teil liefert als PHP-Erweiterung die Rohdaten der Analyse, ist aber nur aktiv, wenn ein Profil erstellt wird, und produziert im Übrigen keinen Overhead. Der Agent läuft als Daemon auf dem Server, aggregiert die Daten, bereitet sie auf und schickt sie an die Plattform blackfire.io zur Darstellung und Auswertung. Mit dem Companion, einer Erweiterung für den Chrome-Browser, können Anwender Auswertungen direkt vom Browser aus steuern.

Zum Einstieg bietet SensioLabs nicht nur einen begrenzten Testzeitraum für die Premium-Variante, sondern außerdem eine

dauerhaft kostenlose Produktversion namens Hack an. Mit dieser lassen sich immerhin zwei Referenzprofile für die Dauer eines Tages speichern. Weitere Funktionen schlagen mit mindestens 82,50 Euro pro Monat für den Premium-Zugang zu Buche. Im Gegensatz zu Tideways misst Blackfire nicht automatisch, sondern nur bei expliziter Aktivierung. Die geschieht entweder über den Companion oder auf der Kommandozeile per *blackfire curl*. Der Companion nimmt Kontakt zur momentan besuchten Website auf und versucht, eine Verbindung mit dem Blackfire-Konto des Benutzers herzustellen. Hierzu müssen Anwender aus Sicherheitsgründen zuvor die Server Credentials genannten Sicherheitstoken in der Konfiguration des Blackfire-Agenten hinterlegen. Der Companion initiiert danach die Datensammlung des Agenten, nimmt hierzu zehn Samples und ermittelt die Mittelwerte. Das ermöglicht Profiling auf Knopfdruck; nach dem Aktivieren des Profilers und einer kurzen Wartezeit bereitet Blackfire die Ergebnisse auf der blackfire.io-Plattform übersichtlich und interaktiv auf.

Über den Call Graph lassen sich sofort die kritischen Stellen erkennen und ein Klick auf die Funktionen öffnet Detailinformationen zu Zeit-, CPU- und Speicherverbrauch. Blackfire bietet zusätzlich Analysen zu Netzwerk- und SQL-Abfragen an, genau wie Tideways, aber im Gegensatz zu den Open-Source-Lösungen Xdebug und Xhprof. Diese Funktion steht allerdings nur in den kostenpflichtigen Premium- oder Enterprise-Versionen zur Verfügung. Ebenfalls aufpreispflichtig ist Team, eine Funktion, mit der sich Anwendungen von mehreren Benutzern profilen und die Ergebnisse im Team analysieren lassen. Allerdings können Anwender Team-Profile im Gegensatz zu Einzelprofilen nicht öffentlich über einen geheimen Link teilen, sodass man sich vorher über den Verwendungszweck im Klaren sein sollte.

Ein Vergleich zweier Profile in Blackfire nach Optimierung der ORM-Abfrage (Abb. 3)

Für den Vergleich zwischen Profiling-Aufrufen, insbesondere

nach Änderungen am Code, gibt es die sogenannten Referenzprofile. Ein Profil kann Blackfire als Referenz speichern und erlaubt den Vergleich mit anderen aus der „Timeline“. Genauso wie bei Xhprof zeigt ein Call Graph die Änderungen visuell. Der Vorteil zum statischen PNG-Graphen von Xhprof ist hierbei die optionale Interaktion, da der Call Graph von Blackfire SVG nutzt. Man kann zoomen, den Ausschnitt verschieben sowie einzelne Teilbäume in den Fokus rücken und beim Klick auf einen Knoten die Detailinformationen auf der linken Seite anzeigen lassen.

Als Vorteil von Blackfire stellt SensioLabs heraus, dass Kernfunktionen in PHP, die Entwickler ohnehin nicht optimieren können, im Vorhinein aggregiert sind und die Auswertung sich somit auf die interessanten Funktionsaufrufe beschränkt. Dies ist insbesondere im Vergleich zu Xdebug und Xhprof ein Vorteil, die alle Funktionen gleichberechtigt darstellen.

SensioLabs-Geschäftsführer Fabien Potencier sieht Profiling nach Unit-Tests als nächsten bedeutenden Schritt für Entwickler. Blackfire soll dabei unterstützen, indem es Profiling mit wenig Overhead produziert. Dieses Versprechen löst die Software ein, zumindest kann die Probe-Extension im Produktivbetrieb aktiviert bleiben. Beim Erstellen des Profils generiert Blackfire allerdings einen Overhead von circa 25 %, der die gemessenen Zeiten leicht verfälscht.

New Relic

New Relic gehört primär zur Klasse der Application-Performance-Management-Werkzeuge, bietet aber Funktionen für das Profiling von Anwendungen. Das Tool steht für die Plattformen PHP, Ruby, Java, .NET, Node.js und Python zur Verfügung und kann auf Linux- und Windows-Servern arbeiten. Um die Profiling-Funktion zu nutzen, muss der Kunde die Pro-Variante für 149 US-$ monatlich pro Host erwerben.

Zunächst muss man den sogenannten Agenten auf dem jeweiligen

System installieren, der die Umgebung überwacht und alle relevanten Informationen sammelt. Jede Minute überträgt er die so gesammelten Daten an die Relic-Server. Erst die bereiten die Daten auf und stellen sie dem Benutzer webbasiert zur Verfügung. New Relic verarbeitet und speichert Kundendaten in den USA. Zusätzlich nimmt sich New Relic das Recht heraus, diese Daten auch außerhalb der USA zu verarbeiten. Dieses Recht ist in den AGB verankert, aber laut kürzlich erfolgter Entscheidung des EuGH keine gesetzliche Ausnahme mehr.

Alle wichtigen Informationen, potenzielle Flaschenhälse, Fehler oder Störungen liefert das Dashboard. Die dargestellten Informationen können Anwender im Hinblick auf den betrachteten Zeitraum individuell auswählen. Ein Vergleich mit historischen Daten ermöglicht es, die Auswirkung von Änderungen am Quellcode zu bewerten.

New Relic APM fasst ebenfalls Informationen über einen HTTP-Request unter dem Begriff Transaktion zusammen. Eine Top-5-Anzeige der zeitaufwendigsten Transaktionen ermöglicht einen schnellen Einstieg in die Analyse. Auffällig ist, dass das Werkzeug keine Informationen zum Speicherverbrauch auf dieser Ebene anzeigt.

Die langsamsten Komponenten, üblicherweise Funktionsaufrufe, erscheinen in einer tabellarischen Übersicht – in der Demo der konkrete Klassenname, Funktionsname und deren Aufrufhäufigkeit. Mehr als ein Indiz dafür, dass der häufige Funktionsaufruf unnötig ist, gibt diese Information leider nicht her. Eine grafische Darstellung des Programmablaufs (Call Graph) sucht man vergebens. Alle Informationen zum Programmablauf stehen in einer baumartigen Tabelle. Verwendete PHP-Funktionen wie *sleep(5)*, die Ursache des Flaschenhalses sein könnten, stellt New Relic APM nicht dar. Die aufgeführten Informationen beziehen sich in der Demo-Anwendung immer auf Klassen und deren Funktionen.

Tabellarische Übersicht des Programmablaufs bei New Relic, deutlich erkennbar der rote Performance-Engpass (Abb. 4)
Aber nicht für jede Transaktion stehen Ablaufinformationen direkt zur Verfügung. Ob detailliertere Daten überhaupt gesammelt werden, hängt von dem zu konfigurierenden Schwellenwert Apdex T ab. Der beeinflusst den Wert, ab dem eine beobachtete Reaktionszeit als tolerierbar eingestuft wird. Trifft das zu, löst New Relic APM keine Benachrichtigung beziehungsweise Warnung aus und sammelt keine weiteren Informationen zur Ablaufsteuerung.

Angaben zum Speicherverbrauch oder zu I/O-Zeiten gibt es nicht. Der Fokus von New Relic liegt hier bei der Darstellung von Antwortzeiten. Ob die Ursache einer schlechten Reaktionszeit eine komplexe Berechnung ist oder der wiederholte Zugriff auf das Dateisystem, kann nur der Blick auf den Quellcode zeigen.

Im Gegensatz zu den anderen Tools registriert New Relic nicht behandelte Exceptions und Warnings und stellt sie prominent im Dashboard dar – im günstigsten Fall außerdem inklusive der jeweiligen Stacktrace.

Die Verwendung von Profilern während der Entwicklung einer PHP-Anwendung ist sicherlich empfehlenswert, um frühzeitig eventuelle Flaschenhälse zu entdecken und geeignete Gegenmaßnahmen einzuleiten. Aber oft treten im produktiven Einsatz Situationen ein, die zuvor niemand in Betracht gezogen hat oder die aufgrund der wechselnden Betriebsumgebung außerhalb des Einflusses des Entwicklers liegen.

Profiling-Werkzeuge im produktiven Betrieb

Für eine grundsätzliche Aussage, ob der Einsatz eines der hier vorgestellten Profiling-Werkzeuge im produktiven Betrieb sinnvoll ist, haben die Autoren die Performance der Demo-

Anwendung überprüft. Ausgehend von einer Konfiguration ohne Profiler haben sie den aktiven Einsatz jedes Profiling-Werkzeugs mit dem Apache HTTP Server Benchmarking Tool (*ab*, siehe wiederum den Balken „Alle Links“) verglichen.

Tideways und New Relic beeinflussen die Performance der PHP-Anwendung mit 11 und 13 % zwar erkennbar, aber dennoch so wenig, dass bei ausreichenden Systemressourcen ein Einsatz auf dem produktiven System in Betracht gezogen werden kann. Vor allem wiegen die Vorteile des umfassenden Monitorings den geringen Overhead auf, aufkommende Schwierigkeiten lassen sich mit beiden Tools im Produktivsystem in Echtzeit erkennen und fördern so proaktives Handeln. Ein Einsatz von Blackfire im produktiven Betrieb ist trotz 27 % Overhead gerade noch denkbar. Zwar erstellt Blackfire nur ein Profil, sobald man es aktiv dazu aufruft, dennoch sollten Anwender in diesem Fall den Einfluss auf das System berücksichtigen. Ebenso verhält es sich mit Xhprof. Ein Einsatz von Xdebug auf einer produktiven Maschine ist angesichts des gewaltigen Overhead nicht empfehlenswert.

Fazit

Profiler sind wertvoll für die Analyse von Anwendungsperformanz. Mittlerweile steht eine Vielzahl an Profiling-Werkzeugen für PHP zur Verfügung. Welches am besten geeignet ist, kommt auf die jeweilige Anwendung und die zu beantwortenden Fragestellungen an. Jedenfalls braucht niemand mehr individuelle Zeitnahmen im Code selbstständig zu implementieren. Die Tools sind leicht zu installieren und Anwender können sie ohne oder mit wenigen Anpassungen am System sofort einsetzen. Selbst der Einsatz bestimmter Profiler auf Produktivumgebungen ist denkbar, um auftretende Störungen live analysieren zu können.

Die kostenlos verfügbaren Werkzeuge bieten schon einen ausreichenden Funktionsumfang und Entwickler können sie vollständig in der eigenen Umgebung einsetzen. Kommerzielle

Produkte punkten mit Zusatzfunktionen wie SQL-Analyse, verbesserten Analysemethoden oder Teamfunktionen. Dafür verlangen die Dienste, dass die Anbieter die Daten nur auf ihren Plattformen verarbeiten und die Kunden nach Ablauf des Abonnements ihre Ergebnisse nicht mehr abrufen können. Wenn man einen solchen Service nutzt, sollte man sich zudem darüber im Klaren sein, dass die übermittelten Informationen Klassen- und Methodennamen enthalten. Der Transport geschieht zwar verschlüsselt, und die Betreiber versprechen, dass sie diese Daten nicht weiterverarbeiten, aber bei manchem bleibt Skepsis. Wirkliche Wirtschaftsspionage ist mit diesen Informationen kaum realistisch, aber unter Umständen können Vertraulichkeitsvereinbarungen mit dem Kunden der Nutzung dieser Dienste einen Riegel vorschieben.

Blackfire in der Hack-Variante oder Xhprof eignen sich besonders, wenn erste Erkenntnisse gewonnen werden sollen – oder bei schmalem Budget. Tideways bietet fürs Geld übersichtliche Analysen und eine einfache Bedienung. APM-Werkzeuge wie New Relic gehen weit über das reine Profiling hinaus und eignen sich vor allem für Betreiber großer Webseiten. (hb) Jörn Wagner, Christian Ehmke arbeiten als Softwarearchitekten bei der Explicatis GmbH in den Bereichen Beratung, Konzeption und Realisierung von Web- und Anwendungssoftware.

[/expand]