



Scout APM

New Relic Comparison Guide



Table of Contents

01 **App Overview**

02 **Language Support & Pricing**

03 **Transaction Traces**

04 **Web Endpoints**

05 **Web Endpoint Detail**

06 **Database Monitoring**

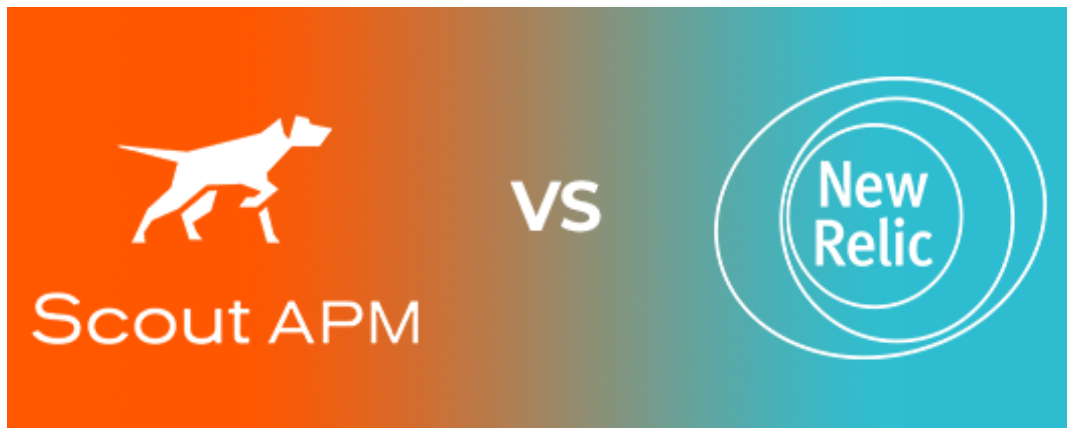
07 **Background Jobs**

08 **Development Profiler**

09 **Weekly Trends Email**

10 **Agent, Alerting & Error Analytics**

11 **Conclusion**



Scout vs New Relic

There's no type of monitoring tool that can get to the heart of a performance problem faster than Application Performance Management (APM).

If you're making this decision, you may be choosing between New Relic and [Scout](#). There are cases where New Relic is a better fit, clear cases where Scout fits like a glove, and some murky situations as well.

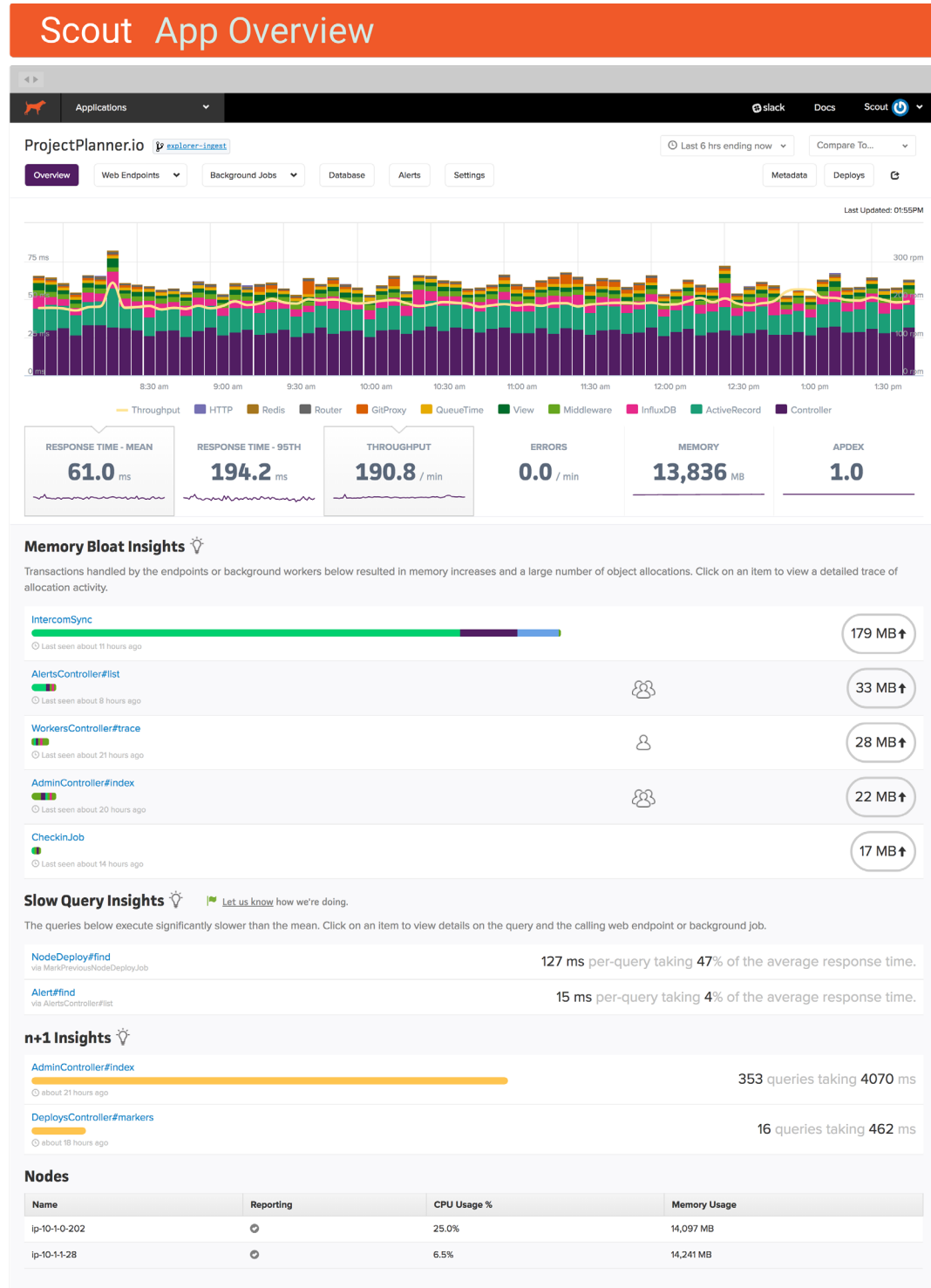
Where do you fall?

In this guide, we will summarize the factors to consider when choosing your best APM product.

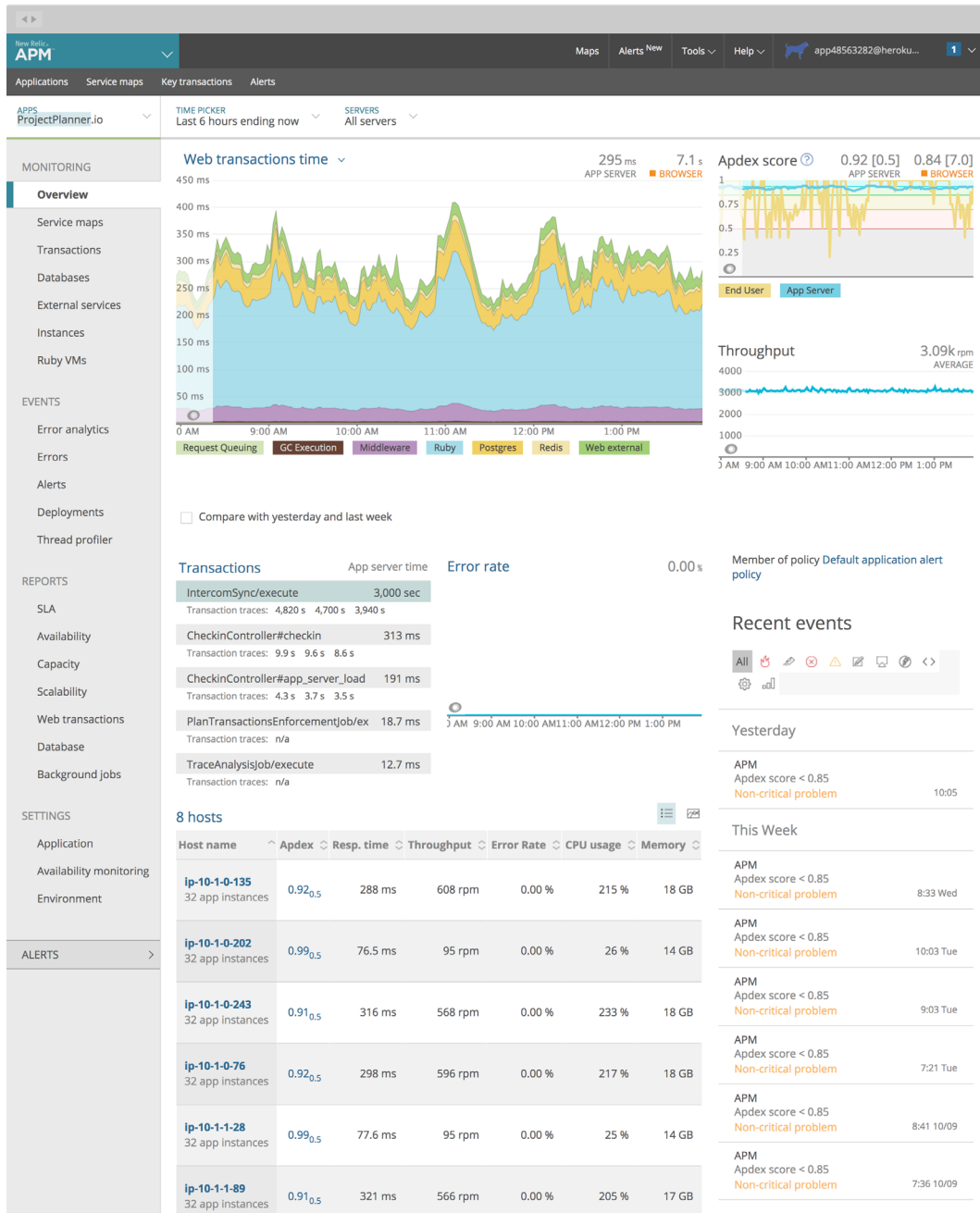
CHAPTER 1

App Overview

This is the first page you see when checking the health of your app. Scout's version is below and New Relic's on the following page:



New Relic App Overview



The top portion of the page is similar between New Relic and Scout: a breakdown of time spent by category (ex: Ruby, Database, External HTTP services, etc) over time. You can view data across similar timeframes in both Scout and New Relic (New Relic offers three months of data in their Pro package and Scout can do the same in their custom plans).

Both let you compare data to the past. Scout gives a few more options than New Relic, which is restricted to yesterday and last week comparisons.

The biggest difference with Scout's approach: several algorithms dig through your data and generate insights on performance directly on this page. For example, Scout identifies [slow database queries](#), [N+1 database queries](#), and [memory bloat](#). These are ordered by impact, which help you decide where to get most improvements in the least amount of time.

Scout short-circuits the work a Site Reliability Engineer (SRE) often performs when investigating the health of an app.

Let's move from the 10,000 foot view to the soul of application monitoring.

CHAPTER 2

Language Support & Pricing

Scout supports [Ruby](#), [Elixir](#), [PHP](#), [Node.js](#), and [Python](#) with a [Core Agent API](#) that can be used to instrument any language. **New Relic supports** Ruby, Java, Node.js, PHP, .NET, Python, and Go.

Scout's monthly fee is about 30% lower than the monthly fee in a **New Relic** annual contract. In most cases, Scout will also buy your New Relic contract.

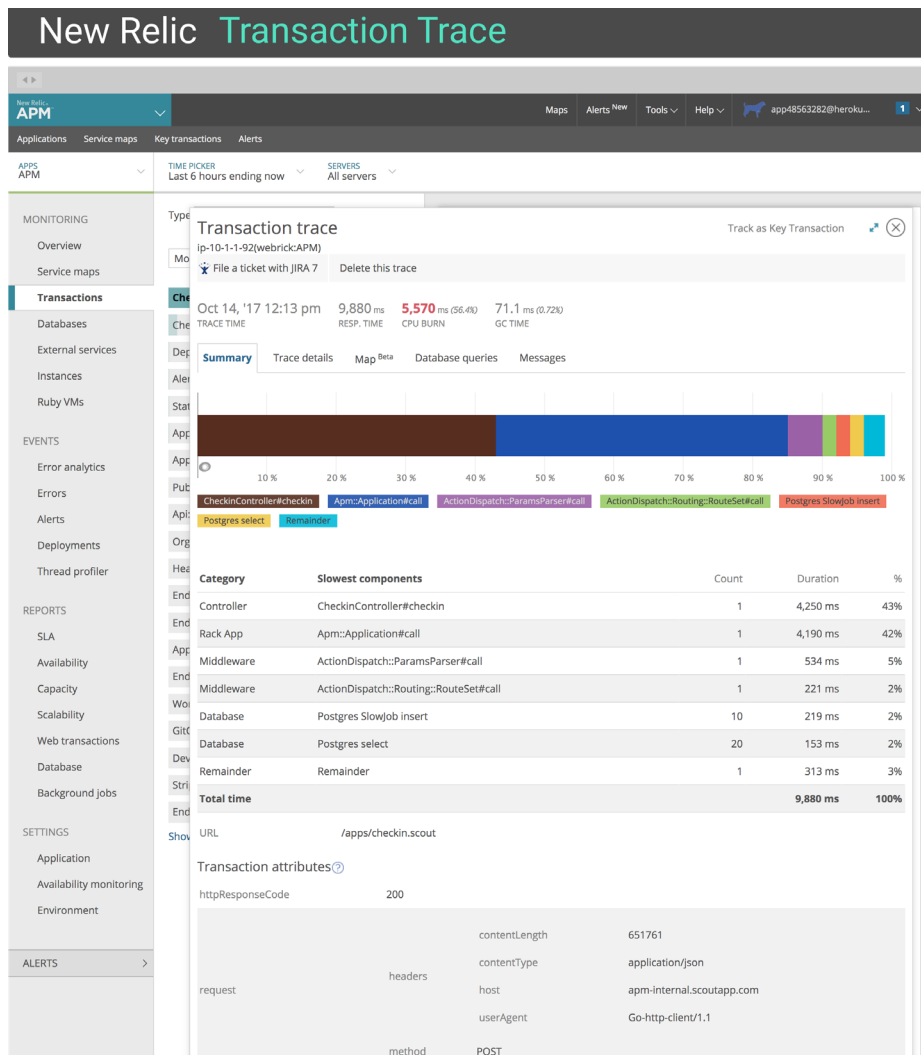
CHAPTER 3

Transaction Traces

The soul of app monitoring is the transaction trace: all the metrics and analysis an APM platform delivers originates from data collected in a trace. A transaction trace is a breakdown of time spent in a single web request or execution of a single background job.

A Scout trace is below and New Relic is on the next page:

The screenshot displays the Scout APM interface for a transaction trace. At the top, the application is identified as 'ProjectPlanner.io' with a 'master' branch. The navigation bar includes 'Overview', 'Web Endpoints', 'Background Jobs', 'Database', 'Traces', 'Errors', 'Alerts', 'Reports', 'Settings', and 'Metadata'. The current view is for the endpoint 'ServicesController#index' on 'Mon 4:21:11 PM - /apps/1/services'. The 'Time Breakdown' section shows a total response time of 2,732 ms, with a breakdown: ActiveRecord (41%, 298 calls), InfluxDB (40%, 5 calls), View (11%, 57 calls), HTTP (5%, 1 call), and Other (3%). Below this is a 'Timeline' view showing the sequence of operations: Middleware, Router/Rails, ServicesController#index, User#find, App#find, Membership#find, and Org#find. A detailed view of the 'SQL#find' operation is shown, highlighting the line of code in 'app/models/org.rb' at line 233: `.first`. A callout box points to this line with the text 'See the line-of-code and author'. The trace also shows other database calls like 'App#find', 'Provisioning::HerokuInstall#find', 'AgentSupport.from_app(cu...p).database_monitoring?', 'AppMetadata#find', 'current_org.addons.active(:db).any?', 'SQL#other', 'InfluxRouter#find', and 'InfluxDB/Query'.



There are some differences here as well:

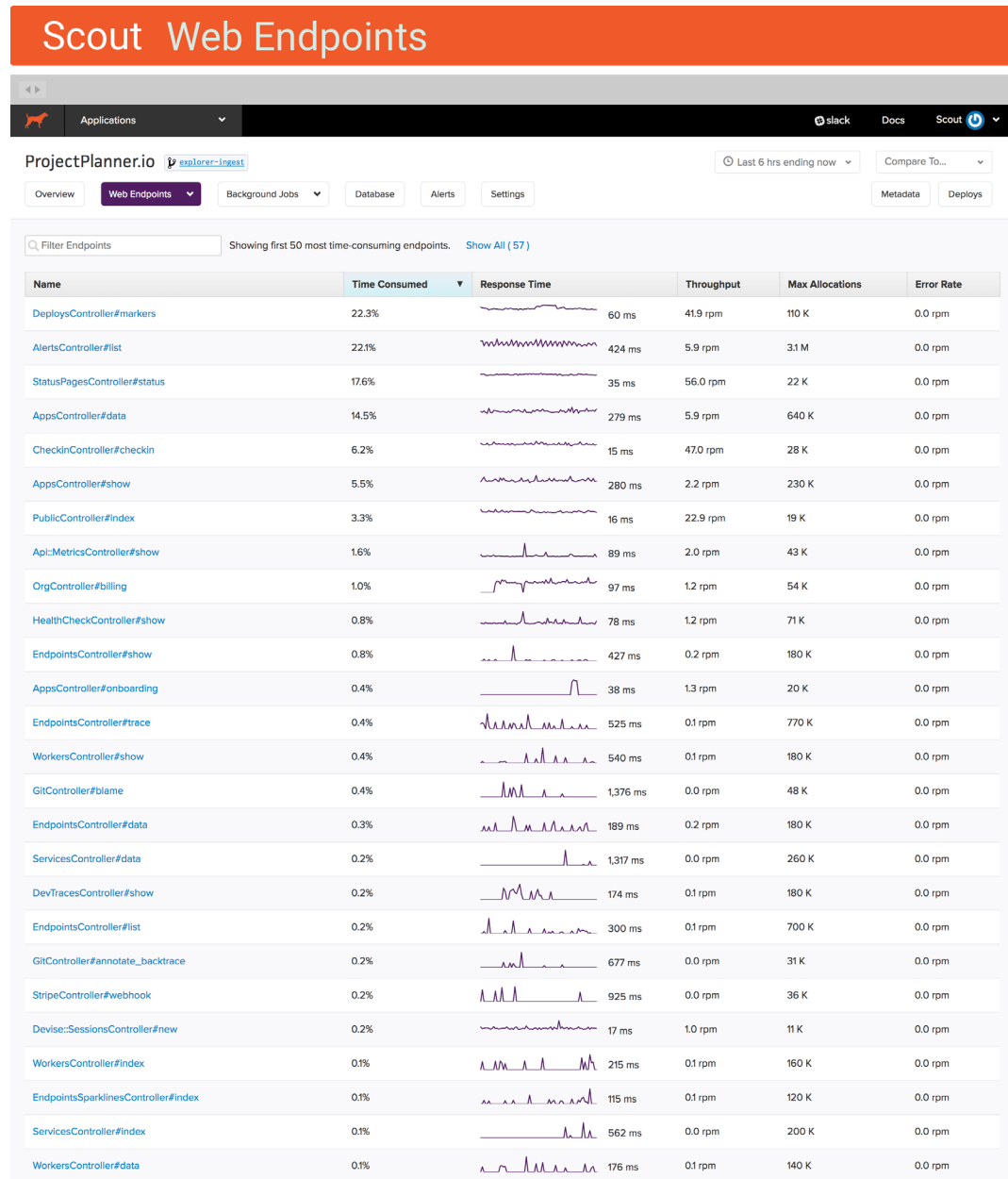
- Scout provides a breakdown of memory allocations in addition to timing metrics. New Relic does not.
- New Relic offers cross-application tracing if your app touches another app you've monitored with New Relic. Scout does not.
- Scout is more likely to provide backtraces to slow method calls and can also display the code directly in the browser.
- Scout indicates the number of rows returned by SQL queries
- Scout can breakdown the time spent in custom code via ScoutProf. New Relic requires custom instrumentation.

Scout typically collects more traces than New Relic. Both New Relic and Scout have similar, one-week retention periods for traces.

CHAPTER 4

Web Endpoints

Both New Relic and Scout let you view summary data on web endpoints in list format. Scout is listed first, New Relic second:



New Relic Web Endpoints

The screenshot shows the New Relic APM interface for monitoring web endpoints. The left sidebar contains navigation options: MONITORING (Overview, Service maps), TRANSACTIONS (selected), DATABASES, EXTERNAL SERVICES, INSTANCES, RUBY VMs, EVENTS (Error analytics, Errors, Alerts, Deployments, Thread profiler), REPORTS (SLA, Availability, Capacity, Scalability, Web transactions, Database, Background jobs), SETTINGS (Application, Availability monitoring, Environment), and ALERTS.

The main content area is titled "Top 5 web transactions" and shows a table of transactions sorted by percent of wall clock time. The table lists the following transactions:

Transaction	Percent of wall clock time
CheckinController#checkin	94.1%
CheckinController#app_server_load	4.46%
DeploysController#markers	0.34%
AlertsController#list	0.33%
StatusPagesController#status	0.27%
AppsController#data	0.26%
AppsController#show	0.09%
PublicController#index	0.05%
Api::MetricsController#show	0.03%
OrgController#billing	0.02%
HealthCheckController#show	0.02%
EndpointsController#show	0.01%
EndpointsController#trace	0.01%
AppsController#onboarding	0.01%
EndpointsController#data	0.01%
WorkersController#show	0.01%
GitController#blame	0.01%
DevTracesController#show	0%
StripeController#webhook	0%
EndpointsController#list	0%

Below the table is a "Show all transactions table..." link.

The "Throughput (rpm)" chart shows a steady increase in throughput over time, with a peak around 11:00 AM. The chart is split into "Non-Web" and "Web" categories.

The "Transaction traces" table shows sample performance details for the "CheckinController#checkin" transaction. The table lists the following traces:

Date	Transaction / Details	App server
12:13 — about 2 hours ago	CheckinController#checkin /apps/checkin.scout	9,877 ms
12:12 — about 2 hours ago	CheckinController#checkin /apps/checkin.scout	8,401 ms
11:10 — about 3 hours ago	CheckinController#checkin /apps/checkin.scout	7,806 ms
11:07 — about 3 hours ago	CheckinController#checkin /apps/checkin.scout	7,909 ms
10:15 — about 4 hours ago	CheckinController#checkin /apps/checkin.scout	8,603 ms
09:51 — about 4 hours ago	CheckinController#checkin /apps/checkin.scout	7,831 ms
08:39 — about 6 hours ago	CheckinController#checkin /apps/checkin.scout	9,615 ms

A "Show more slow transactions" link is located at the bottom right of the transaction traces table.

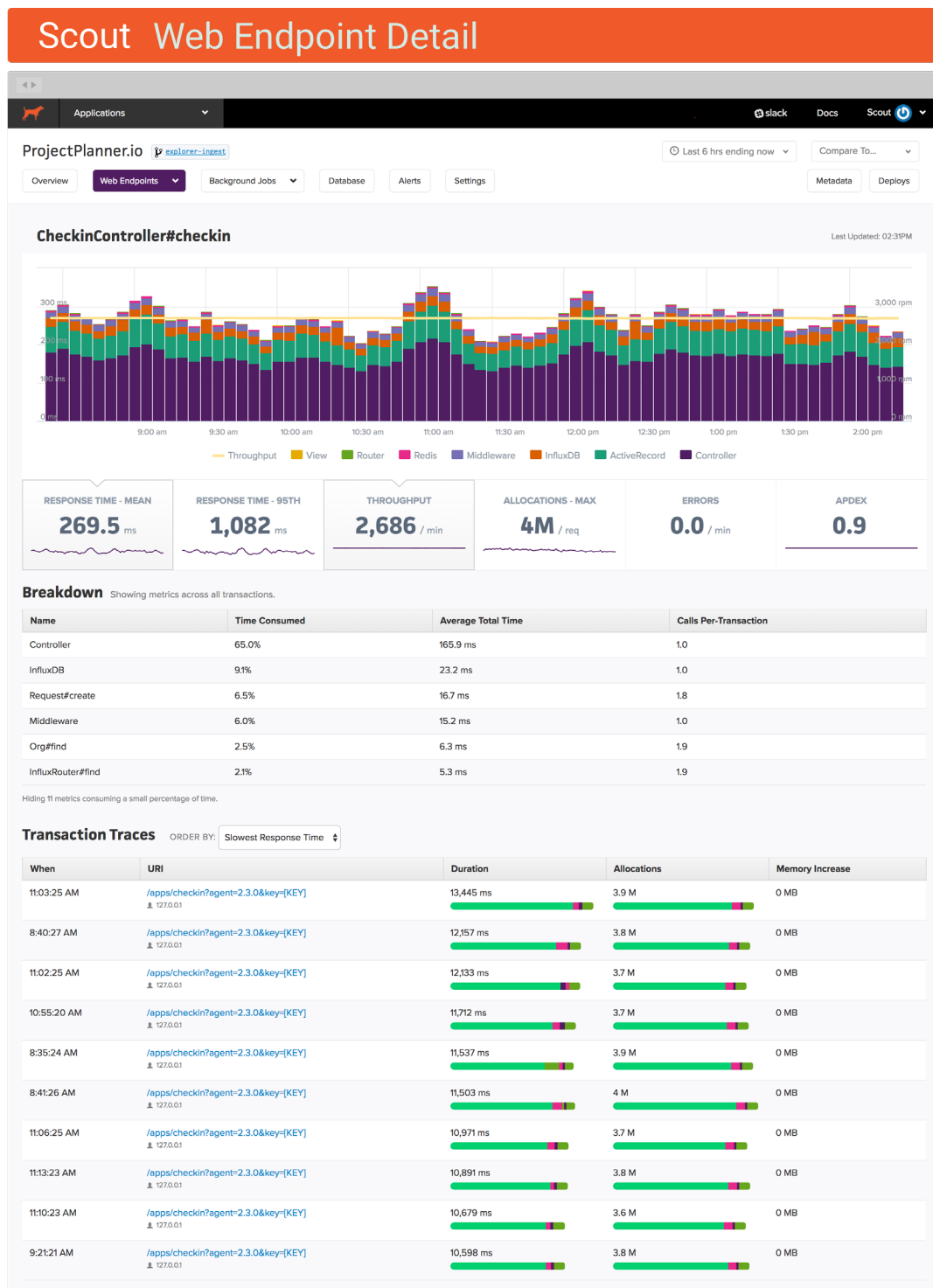
New Relic opts for several different visualizations on this page. Scout goes with a filterable, sortable view.

Both sort web endpoints by percent time consumed by default, helping you focus on the controller-actions that consume the greatest amount of time.

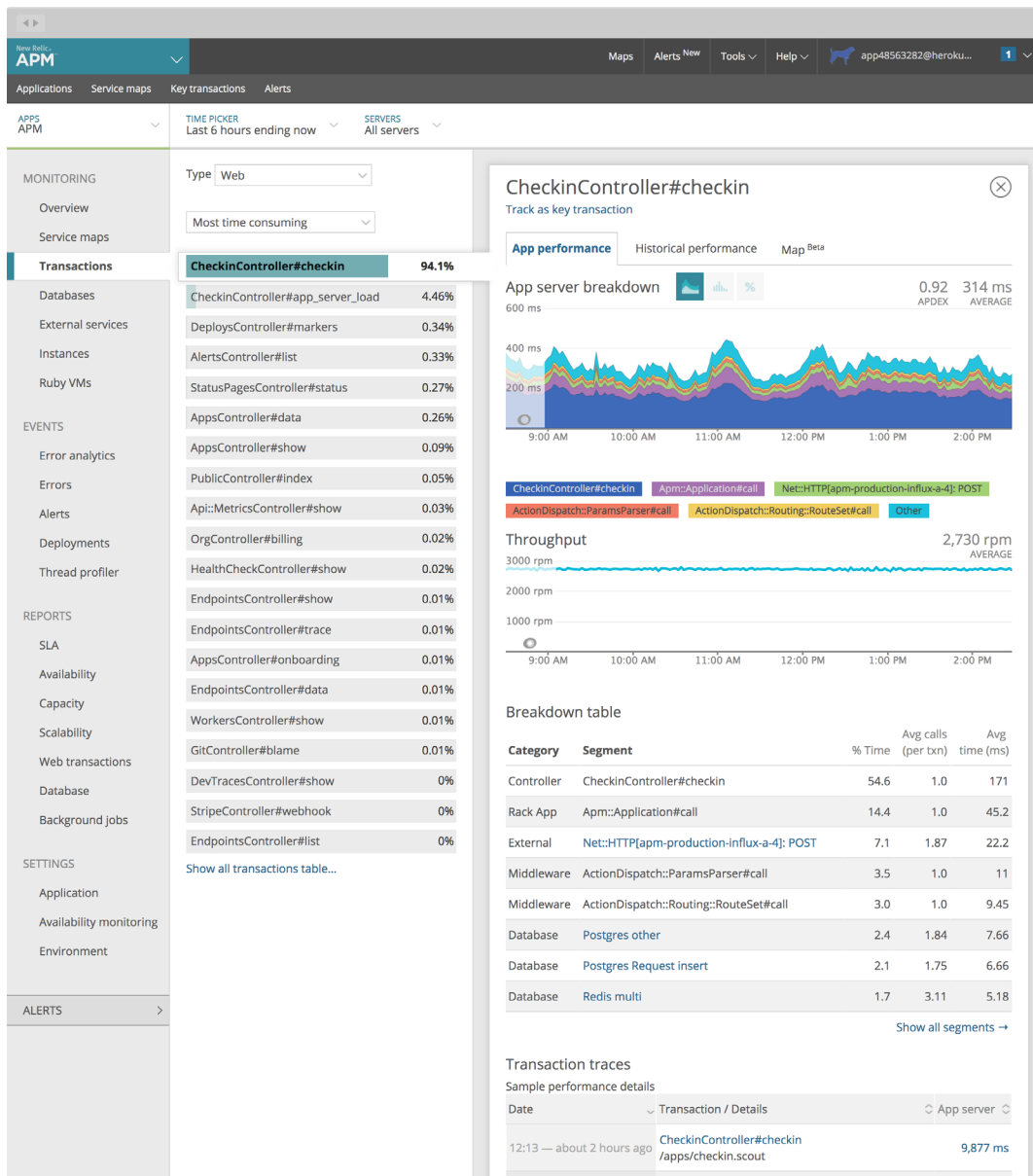
CHAPTER 5

Web Endpoint Detail

In the same way the app overview provides a breakdown of time spent, both New Relic and Scout offer endpoint-specific views for this:



New Relic Web Endpoint Detail



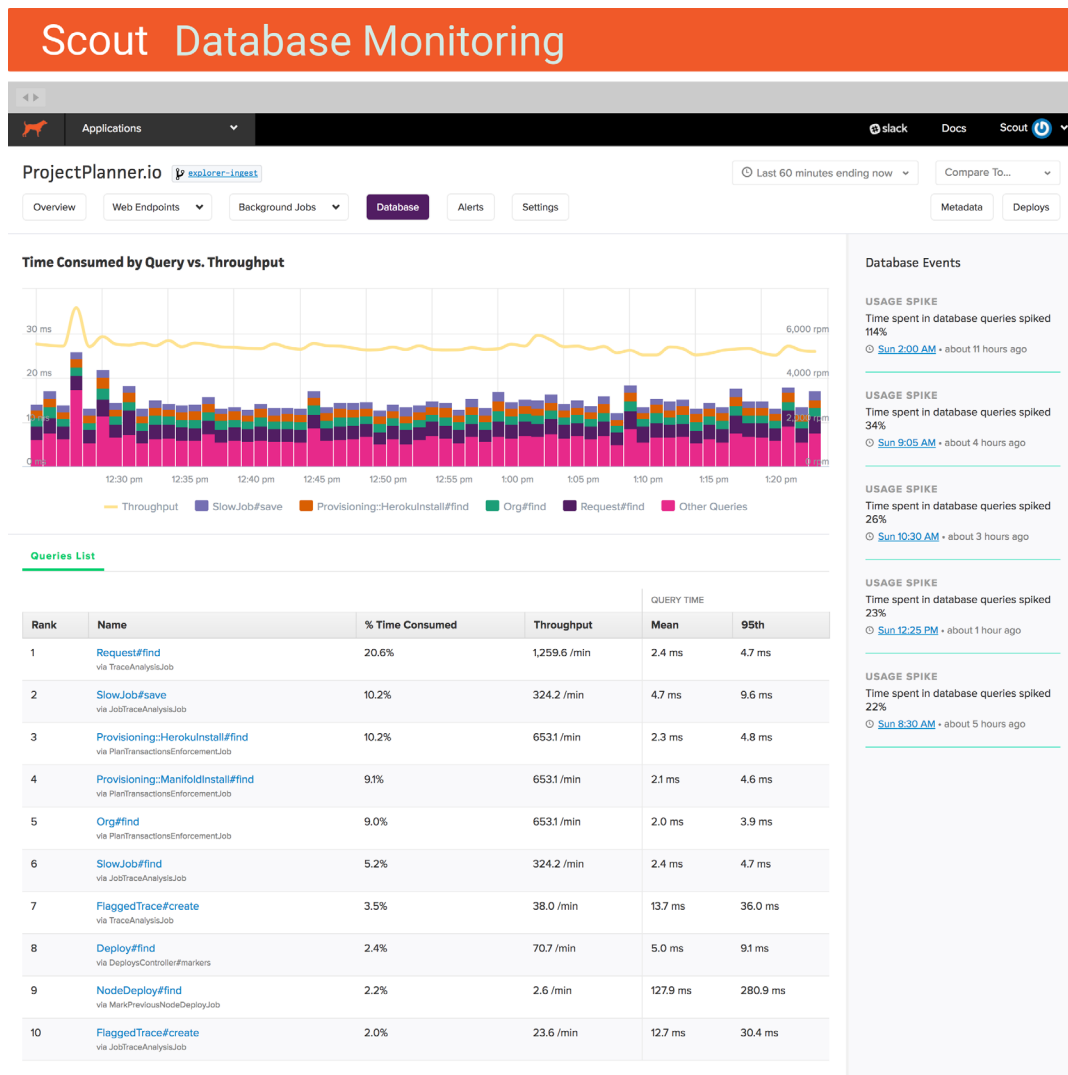
There are some differences on this page:

- New Relic offers a histogram display option of response times and Scout does not.
- Scout offers memory allocation metrics (more allocations lead to increased memory usage) while New Relic does not.
- New Relic's breakdown provides more categories, while Scout focuses the breakdown on database queries.
- Scout typically provides more transaction traces (up to 10 per-minute) and more options for sorting and selecting traces.

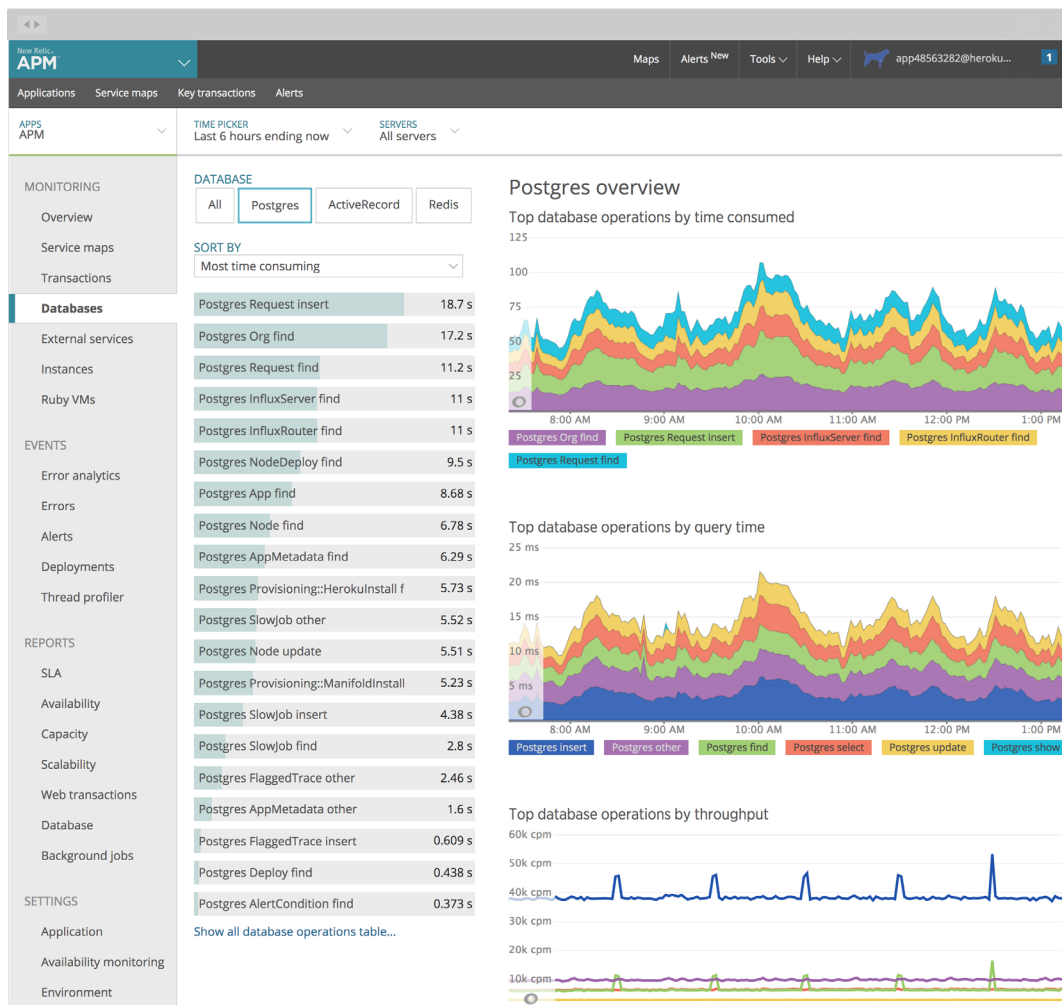
CHAPTER 6

Database Monitoring

The database is the most common bottleneck for web applications. Since a database is a shared resource, one expensive query may cause many other types of queries to run slower. Both tools provide additional analytics around database query performance:



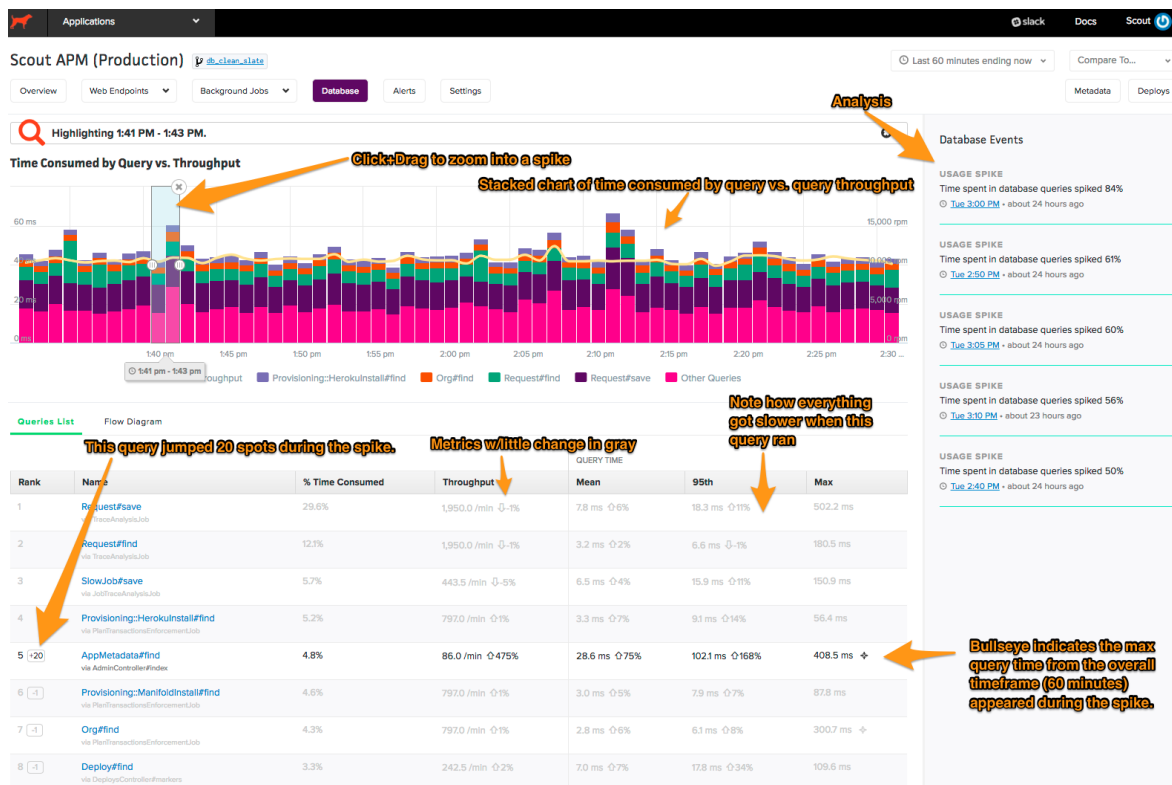
New Relic Database Monitoring



Both of the UIs resemble each tool's web endpoints display but made specifically for database queries. There are some subtle yet important differences between the two:

- Scout's chart displays data across every query - not just the top five most expensive queries - by combining less expensive queries into an "other queries" metric. This gives a more complete picture of performance.
- New Relic provides monitoring beyond ActiveRecord. Scout focuses on ActiveRecord.
- Scout's list of queries includes the calling controller-action or background job. Expensive queries are frequently triggered by a few hotspots in the code: this makes it easier to identify those spots at a glance.

Finally, Scout makes it easy to compare performance in a small slice of time to the normal performance. This isn't possible with New Relic - you can change the entire timezone, but you can't compare between a small slice of time and a larger one. Here's how this looks within Scout:



CHAPTER 7

Background Jobs

Both New Relic and Scout monitor the core Ruby background job frameworks: Sidekiq, Delayed Job, and Resque. The UI is similar to endpoints, so I haven't provided a screenshot here.

CHAPTER 8

Development Profiler

When you are actively resolving a performance issue, it's helpful to get immediate feedback on the results as you code. Otherwise, you need to deploy and wait for fresh data to verify the change had the intended impact.

New Relic used to have a developer mode, but it has since been removed from the agent.

Scout DevTrace gives immediate access to traces in your local development environment by clicking on a speed badge in the lower left of your browser:



Slow feedback cycles - like a long-running test suite - are painful. DevTrace eliminates a lot of this as you can instantly verify that an optimization - like fixing an N+1 query with ``includes`` - is working as expected before deploying.

CHAPTER 9

Weekly Trends Email

A recurring report of your app's health is a great way for managers to stay in the loop on performance and to uncover slowly building problems. New Relic puts more emphasis on high-level numbers. Scout digs more into the source of problems and trends:

Scout Weekly Email

Dec 8 Performance Digest

Across 5 apps, web and background job execution times improved by 20% Thursday vs. the prior week.

We've highlighted ProjectPlanner.io below.

Summary of web & background job performance for the app.

Were there any deploys yesterday?

WEB		BACKGROUND JOBS	
Response Time	Throughput	Execution Time	Throughput
↓10% 50 ms	↑16% 500 /min	↓10% 25 ms	↓2% 1,000 /min

Response times were slowest from 9:00 AM - 11:00 AM. During this period, ActiveRecord call times increased 30% vs. the rest of the day.

Insignificant changes are faded for less attention.

Ties performance trends to a specific deploy, if possible.

Drills deep into app performance, looking for specific categories of metrics with slower response times.

Identifies endpoints that are triggering an abnormal number of slow requests.

Calls out the source of the bottleneck.

- UsersController#show** is 30% faster, with a mean response time of 200 ms.
Performance improved following a deploy with commits from [chris@scoutapp.com](#) at 11:34 AM.
- PhotosController#index** is generating a number of slow requests, including a **7 second request** vs. a mean response time of 355 ms.
A single Photos#find call took 6 seconds.
- CitiesController#show** is 26% slower, with a mean response time of 47 ms.
Response times began increasing around 8:00 PM.

New Relic Weekly Email

New Relic in partnership with Heroku Add-Ons

Monday metrics for app48563282@heroku.com

Because Mondays need a little help. Brought to you by Heroku Add-Ons.

Monday 10/02/2017 through Sunday 10/08/2017

Skip ahead to all apps report

ProjectPlanner.io See report at New Relic

1.71 sec Page load time ↓ 6.56 %	23.2 k Views ↓ 3.73 %	0.99 Apdex no change	0.0022 % Error rate ↑ 29.4 %
↓ 34.7 %	↓ 9.02 %	↓ 3.13 %	↓ 55.1 %
From last week			
From 12 weeks ago			

Two month throughput pattern

CHAPTER 10

Agent, Alerting & Error Analytics

Both the New Relic and Scout agents are battle-tested. In [Scout's open-sourced benchmarks](#), **Scout's agent was shown to have lower overhead:**

APM Agent	Response Time (Mean)	Response Time (95th Percentile)	Response Time (Max)	Overhead
None	55.6 ms	106.4 ms	2,174.1 ms	
New Relic	80.4 ms	149.5 ms	2,263.5 ms	44.5%
Scout APM	56.8 ms	102.7 ms	2,168.7 ms	2.2%

The impact of agent overhead increases as CPU resources on your app servers becomes more scarce.

Scout and New Relic have comparable **alerting functionality**.

Scout tracks error rates and integrates with third-party services like Rollbar to provide details on exceptions. New Relic can provide this data, including backtraces to exceptions.

CHAPTER 11

Conclusion

New Relic and Scout require little configuration - outside adding a dependency - to try. You can run Scout and New Relic at the same time without causing conflicts.

Our suggestion? [Try Scout's free 14-day trial](#) alongside New Relic, then ask: "which tool helps your team solve performance issues faster?"