How To Maximize Your CDN Investment

An Actionable Guide To Building A Proactive Observability Strategy

catchpoint

Contents

Introduction

Content Delivery Networks (CDNs) have been around for more than two decades, and they are well-used globally. Optimal and efficient use of CDNs is one of the quickest and easiest ways to optimize application performance and deliver a reliable service. CDNs accelerate content delivery, provide caching services, optimize images, stream video delivery, and provide perimeter security functions. In other words, they transform customer experiences by streamlining, securing, and delivering content faster to the end user.

While there are many benefits to using CDNs, there are also a variety of challenges. The tens of hundreds of caching, storage, and compute nodes involved in a CDN introduce risks which need to be managed consistently. Those who do not incorporate a comprehensive monitoring plan to manage these challenges face serious risks.

Problems resulting from not implementing a consistent observability program can include:

- Losing customers to competitors (e.g., those with better edge reach).
- Limiting the ability to capitalize on emerging markets.
- Increasing employee burnout resulting from lack of ability to efficiently respond to incidents and micro-incidents.

Instrumenting telemetry on internal networks, applications, and infrastructure alone is not enough to monitor CDNs. They are, by definition, highly distributed and edge based. Instead, organizations should use an experience-based, distributed CDN monitoring and observability strategy, tapping into multiple data sources and protocols to ensure an optimal experience. This type of strategy will provide increased observability for predicting and preventing incidents, while steering traffic to the CDN that delivers the best performance.

In this eBook, we will provide the actionable knowledge you need to develop a transformational approach to observability that will result in the best possible outcome. We will dive into topics such as:

- Checking DNS resolution
- Identifying CDN mapping anomalies
- Gaining a detailed view of cache hit ratio

Why is CDN Observability Critical?

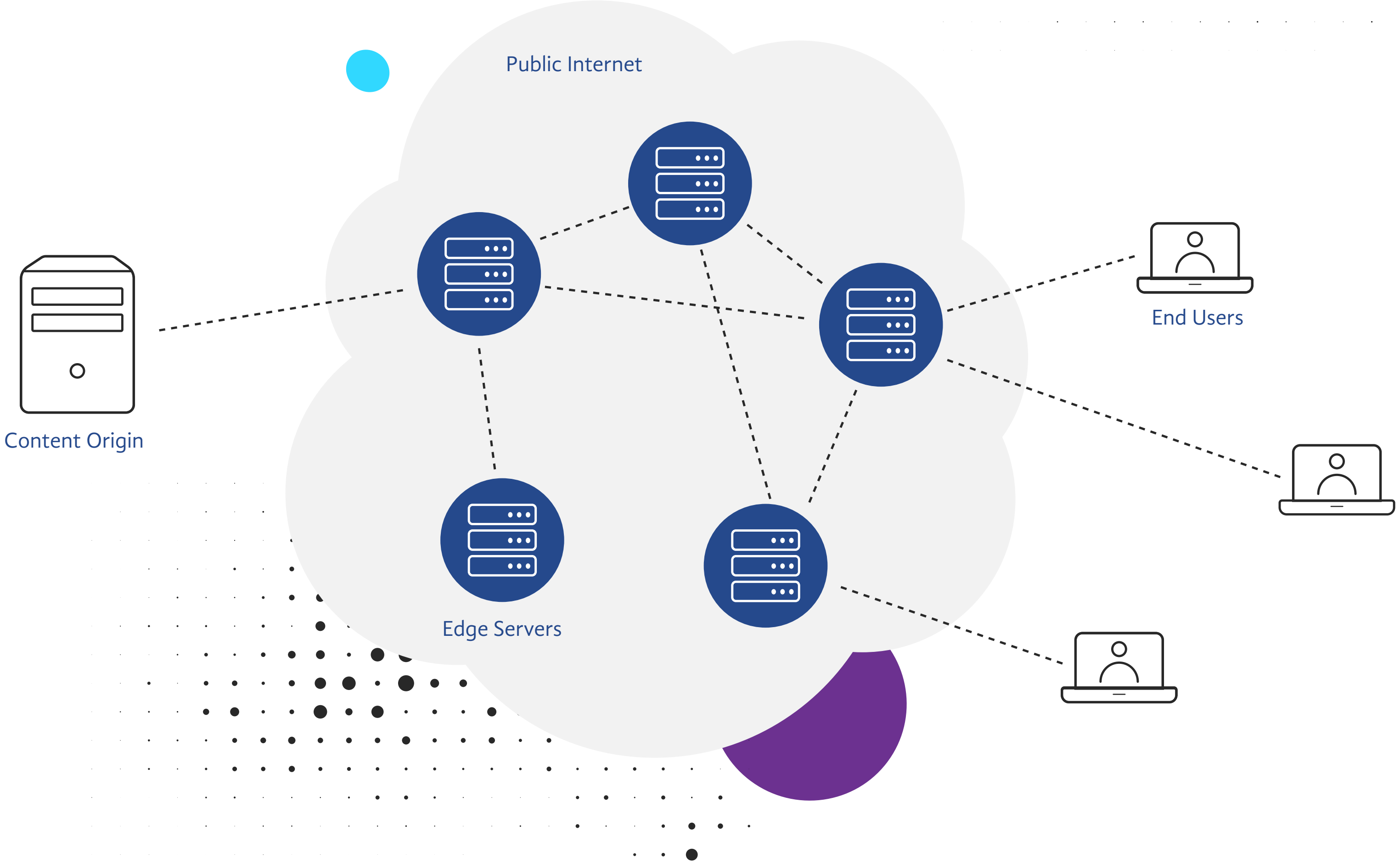
A CDN observability strategy that tracks the key features and measures CDN SLI will ensure its services are utilized effectively. Reducing the time to detect problems and drastically improving “mean time to resolution” or MTTR is an obvious outcome of this monitoring strategy.

Many organizations are now adopting a multi-CDN approach to ensure performance around the globe. Increasingly, organizations are also using geo-specific CDNs and real-time, performance-based CDN routing, which adds greater complexity to the content delivery path. These changing conditions make CDN observability even more critical. Continuous monitoring enables you to regularly assess CDN performance and evaluate new CDN vendors to fill in potential gaps.

Monitoring performance of content delivered using CDN vs. origin helps determine how much your key metrics have improved based on the presence of the CDN. Additionally, CDN observability lets you:

- **Benchmark CDN performance**
  The data aggregated over time helps identify performance patterns and areas for optimization.
- **Hold CDN vendors accountable in case of an SLA breach**
  IT teams can leverage the collated performance data to ensure their CDN vendor is offering the best possible service.
- **Quickly detect and fix performance issues**
  Monitoring origin and edge servers is key to delivering superior end-user experience.
- **Streamline incident management**
  Better manage performance incidents, quickly detect performance issues, and lower the mean time to resolution.

A proactive CDN observability strategy that specifically targets key performance indicators such as network performance, web performance, CDN infrastructure, origin to edge stability, and content integrity validation is essential to achieve your digital experience goals.



The complexity of an end user request for content when it is fetched from the origin.

How to Build a Proactive CDN Observability Strategy

CDNs have sophisticated infrastructures. There are multiple interdependent components that can create bottlenecks and performance issues. As stated, CDNs add to the existing complexity of application architecture. For example, the use of cache and optimized routing between CDN PoPs are vital to delivering content quickly. An issue with either of these functions can introduce latency issues. Monitoring all the different processes and network paths is vital to maintaining optimal CDN performance.

That said, what should you monitor and where should you start?

Understand the incident dimensions you will be observing

To begin with, you need to understand two critical dimensions when designing a CDN observability strategy. Since a CDN, by definition, is highly distributed and edge-based, these two dimensions create four possible combinations through which an observability strategy is contextually applied.

The two dimensions are:

- **Micro (short duration) versus sustained (long duration)**
  A micro-incident is short lived, generally less than 20 minutes.
  A sustained incident is of a longer duration.
- **Regional (localized) versus system wide (majority or all)**
  A regional incident applies to a specific PoP or region. A system-wide incident applies to a majority of the CDN.

If you are trying to detect and fix micro-incidents on a regional basis, consider monitoring at a higher frequency or implementing site reachability diagnostics. However, if you are only concerned with sustained incidents across the entire system, it may be acceptable to monitor at a lower frequency using a different data source (for example, monitoring directly from public cloud providers versus from Internet broadband and backbone ISP).

> “The CDN team uses Catchpoint for internal monitoring, helping inform enhancements, network changes, changes to applications, and other tweaks designed to optimize performance.
>
> At the same time, the cloud services and network team uses Catchpoint on a more granular level, to understand details such as data transfer times and DNS status.”

Joshua Johnson, Cloud Solutions Architect
Tencent Cloud North America

Read more

Check DNS resolution

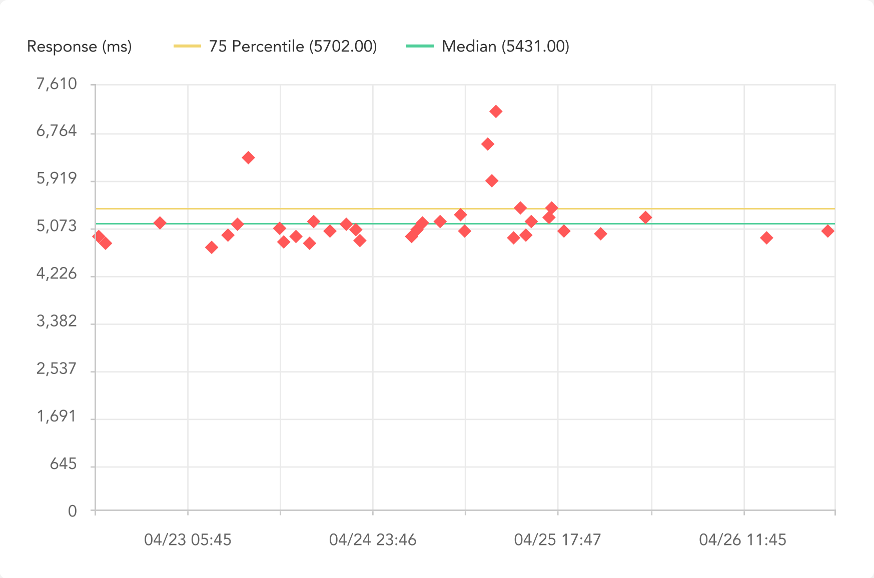
DNS resolution is a lengthy process with multiple server hops. Introducing a CDN to the equation can add additional points of failure. Using active observability allows you to emulate a client DNS query. This makes it easier to detect and quickly resolve DNS issues, for example, misconfigured nameservers in the DNS resolution chain.

If you are using a managed DNS, active observability will help you with analysis of performance and availability and enable you to identify issues across the levels of DNS resolution. Monitoring DNS will also help you identify configuration issues and can serve as the first line of defense in case of a DDoS attack.

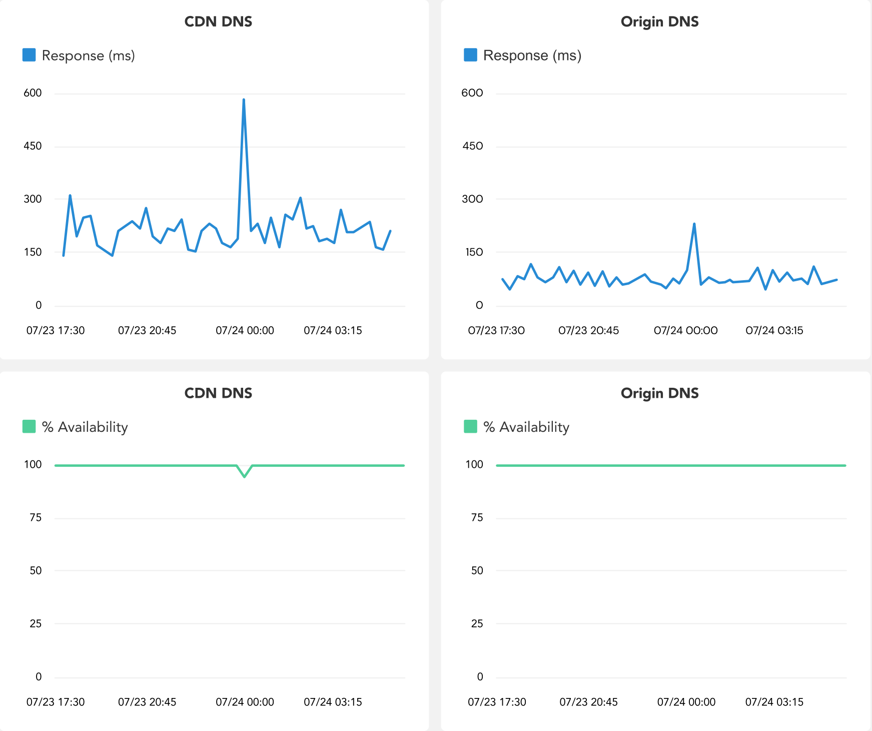
CDNs use DNS redirection methods to route requests to the appropriate edge servers. This lets webpage resources load faster. There are instances where CDN nameservers are slow to respond, resulting in performance degradation. It is important to constantly monitor such cases. The best way to do so is using active observability for DNS and the webpage itself.

The scatterplot graph on the right highlights DNS failures. In this scenario, the authoritative nameservers originating from the same subnet IP address fail to respond when requests come from a specific ISP.

You should also observe DNS performance of CDN vs. origin. The chart to the right shows a time-based line of the DNS resolution time between CDN and origin. You can correlate the dip in availability with the spike in CDN DNS response time.



Scatterplot graph – the red data points denote DNS failures.



Charts comparing the performance of CDN DNS vs. origin DNS.

Check CDN mapping

CDN providers identify and map optimal paths for requested content based on an end user's location. CDN observability will identify mapping anomalies and sub-optimal peering policies. It also verifies if the end user is being served from the nearest edge server.

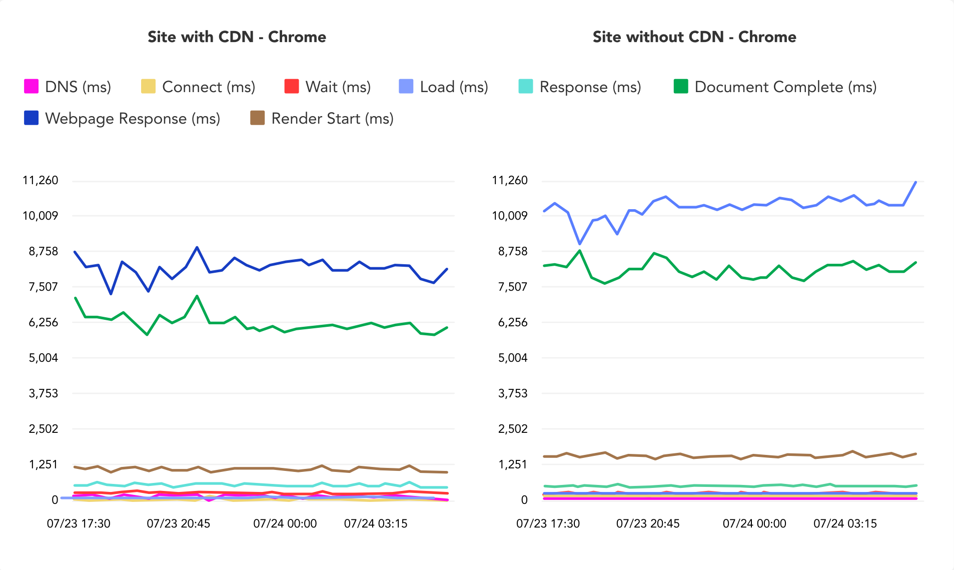
Consider the following example of an active observability test to monitor CDNs. Most CDNs either use a CNAME Resource Record (e.g., *akamaiedge.com) or a set of A records, which you need to map the domain name to. When setting up tests to evaluate CDNs, you should ideally have two sets of tests running:

- A set of tests which hit the domain names mapped to the CDN.
- A set of tests which hit domain names overriding IP addresses to that of the origin servers.

Running these active monitoring tests from multiple locations provides useful performance data that you can use to compare CDN vs. origin performance from different regions.

Additionally, analyzing traceroute data allows you to compare network round trip times, that is, the number of hops required to reach a server when using a CDN vs. otherwise. The same use case is also relevant when comparing one CDN vs. another.

Moreover, you can run tests against a domain served by a specific CDN. Overriding the DNS will allow you to test origin performance. The chart to the right is a comparative performance of origin vs. CDN. The webpage response (the total time to load the full page) for the CDN was lower than the origin.



Line chart comparing performance with and without a CDN.

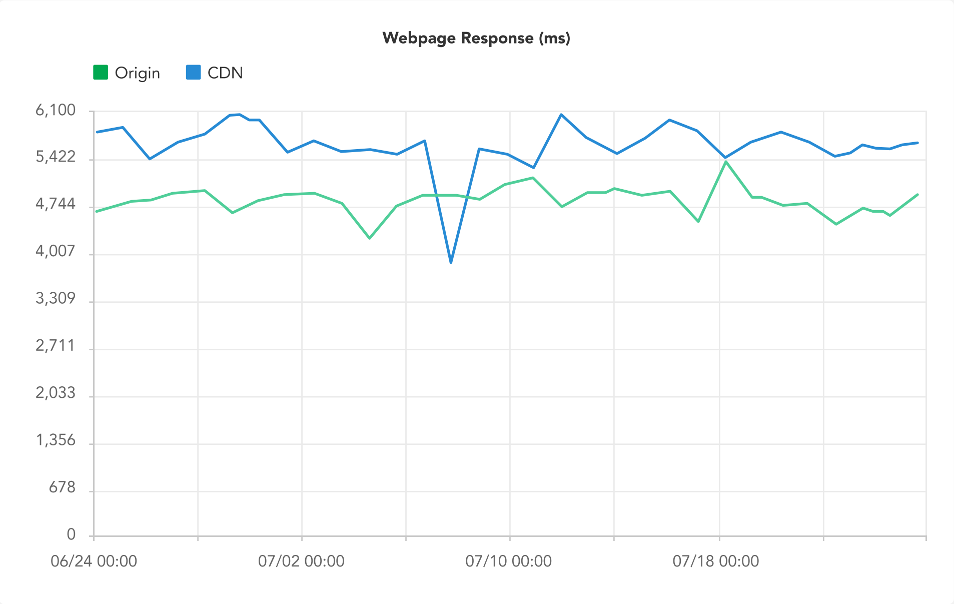


Chart comparing the performance of origin vs. CDN for webpage response time.

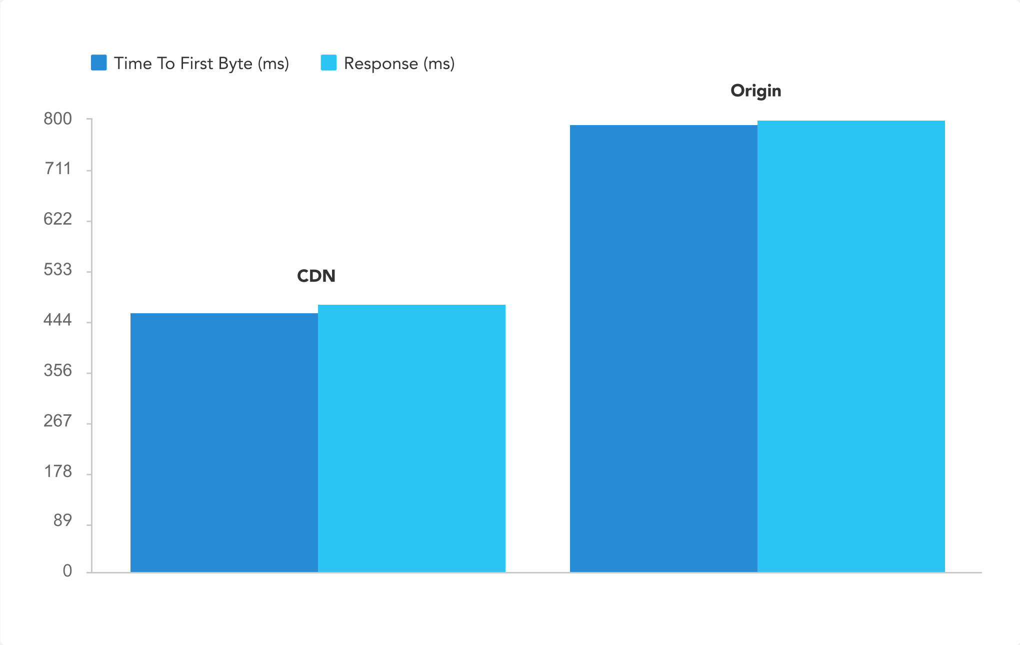
Check cache hit ratio

Cache hit ratio compares the content requests the CDN cache is able to process successfully against the total number of requests it is handling. An efficient CDN will have a high cache hit ratio. In contrast, a higher cache miss means requests are routed back to the origin, as the cache was unable to deliver the requested content.

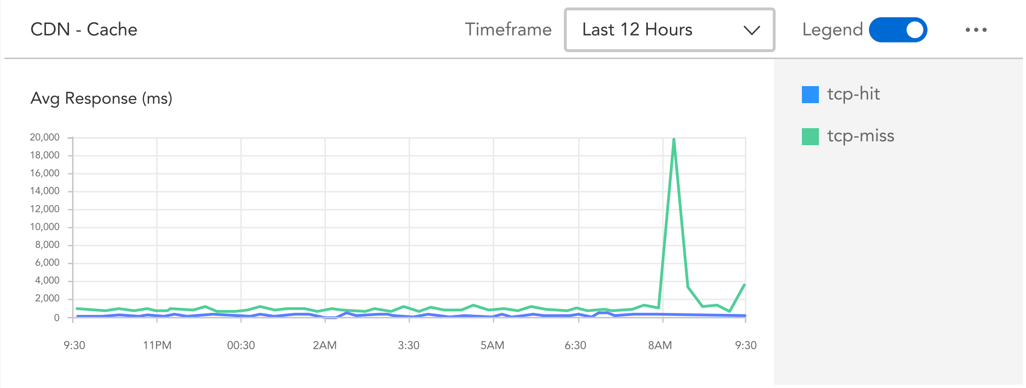
It's important that you ensure your observability strategy enables you to gain a detailed view of CDN cache vs. CDN origin. For instance, in the chart below, you can see exactly how cache hits and cache misses measure against one another in terms of latency.

Advanced monitoring capabilities allow you to compare origin vs. cache KPIs per city. You can use a number of metrics to compare performance, including average ping round trip times, average response, average connect, and so on.

You can analyze different performance metrics to compare CDN performance against origin as well. This allows you to determine optimal content mapping and routing.



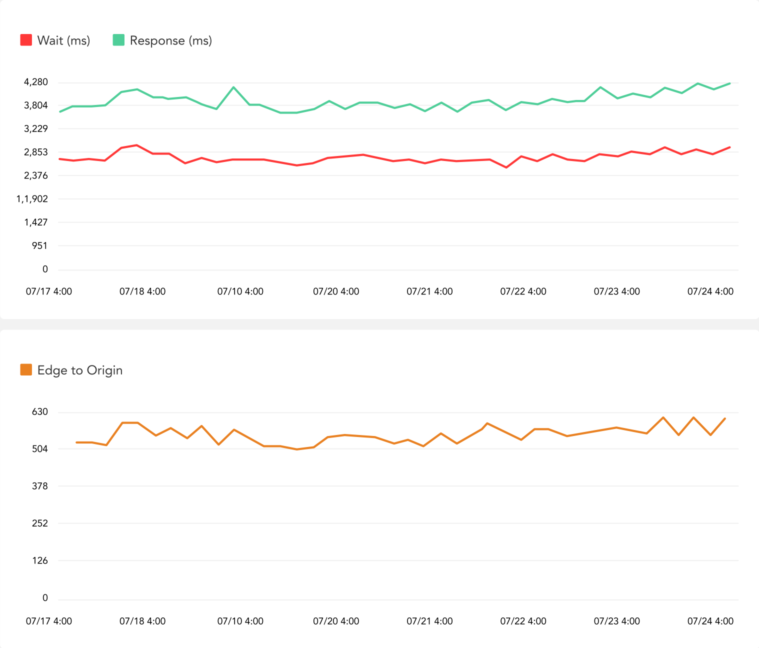
Comparing origin vs. CDN performance using Time to First Byte and Response Time.


CDN - Cache

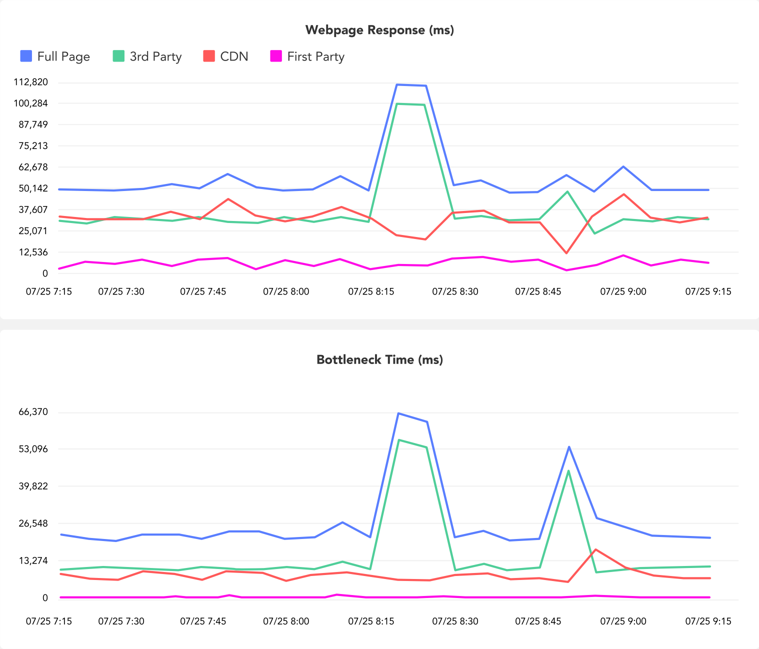
Timeframe: Last 12 Hours

Legend

Measuring CDN cache hit/miss ratio using network monitoring tools.



Measuring latency in different CDN routing paths, which will directly impact end user experience.



Using key performance metrics to determine bottleneck created by content.

Check latency

CDN observability measures end-user-to-edge location latency to track performance degradation between the end user and a specific edge server or across multiple edge servers. Edge-to- origin data center latency is also important to measure, especially if the enterprise has multiple origin data centers.

In the chart to the right, you can identify the CDN origin latency trend (wait and response time correlation).

It is also important to identify any bottlenecks on the page once the hosts on the page have been segregated based on first-party, CDN, third-party, etc. How much of a bottleneck, for example, is created by a specific zone, and how does it impact the overall performance of the page? Metrics to consider include webpage response, bottleneck time, and availability.

CDNs use load balancing to distribute requests to available servers, so traffic is routed optimally without taxing a specific server. This process is called Global Server Load Balancing (GSLB). The traffic is distributed between a set of connected servers around the world to improve reliability and reduce latency. CDN observability ensures that load balancing is optimal and can trigger an alert if there is an unusual surge in traffic.

Check image optimization

Images often account for most of the downloaded bytes on a web page and they can occupy a significant amount of visual space. As a result, optimizing images can often yield some of the largest byte savings and performance improvements for your website. The fewer the bytes a browser has to download, the lower the bandwidth consumption, allowing the resource to load faster.

CDN vendors offer image optimization as an additional feature. From simple compression to advanced optimization, the image is delivered based on end user connectivity, device, viewport, and other factors. It is important to ensure these additional services are functioning and delivering an improved end-user experience. Leveraging both active observability and real user observability enables you to utilize true experience-based monitoring, so you can capture and compare the true range of metrics relevant to image optimization.

Below is an example using Akamai Image Manager. You can set up active observer tests that will allow you to make performance comparisons before and after optimization and subsequently alert if the sizes are not within expected parameters.

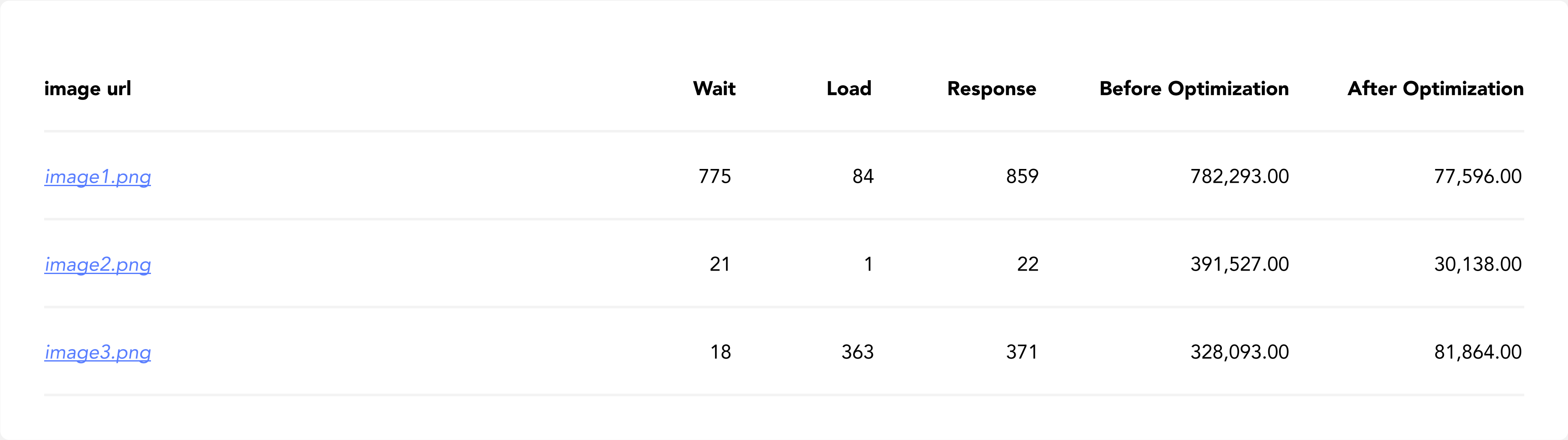


image url	Wait	Load	Response	Before Optimization	After Optimization
image1.png	775	84	859	782,293.00	77,596.00
image2.png	21	1	22	391,527.00	30,138.00
image3.png	18	363	371	328,093.00	81,864.00

Synthetic tests verifying the efficiency of image optimization techniques.

Building your Observability Strategy with Catchpoint

Active observability provides all the insights you need to optimize and maintain consistent CDN performance, delivering the most consistently positive user experience. That's why it's important to partner with an experience observability platform like Catchpoint that can support you on your CDN journey.

- Monitor the last mile network to verify optimal CDN performance and ensure it is mapping end-users to the relevant PoP.
- Track performance across multiple devices, networks, and locations to ensure consistent performance.
- Use performance data to optimize applications, including content, code, user journey, and so forth.
- Identify any bottlenecks, latency, or availability issues within the application infrastructure.
- Perform A/B test changes to content in order to evaluate how such changes impact end-user experience.
- Benchmark performance of CDNs in a multi-CDN environment.
- Track CDN performance for SLA breaches.

With Catchpoint's CDN observability solution you can:

Catchpoint's CDN observability strategy will allow you to monitor the performance of your chosen CDN, from vendor selection, to implementation, to continual optimization.

Whether you are an SRE on call or a CDN monitoring strategist, Catchpoint CDN Observability helps you proactively track CDN performance and ensure your content is being delivered efficiently, with minimum latency and without compromising digital experience, enabling you to gain a competitive advantage.

It's time to enhance CDN performance and start making the most of your CDN investment.

Trusted by
**9 out of
the Top 10**
CDN Providers

Ready to take the next step?

Visit **www.catchpoint.com** to check out
Catchpoint's CDN observability solution today!

About Catchpoint

Catchpoint is the Internet Resilience Company™. The top online retailers, Global2000, CDNs, cloud service providers, and xSPs in the world rely on Catchpoint to increase their resilience by catching any issues in the Internet Stack before they impact their business. Catchpoint's Internet Performance Monitoring (IPM) suite offers synthetics, RUM, performance optimization, high fidelity data and flexible visualizations with advanced analytics. It leverages thousands of global vantage points (including inside wireless networks, BGP, backbone, last mile, endpoint, enterprise, ISPs and more) to provide unparalleled observability into anything that impacts your customers, workforce, networks, website performance, applications and APIs.

Learn more at **www.catchpoint.com**

catchpoint