signal fx

AMAZON ELB + SIGNALFX INTEGRATION

What Is Amazon ELB?

Amazon Elastic Load Balancing (ELB) allows websites and web services to serve more requests from users by adding more servers based on need. A load balancer distributes load across all your servers to ensure even usage of capacity, taking into account the type of services offered by each server, whether each server is healthy, and the demand on the server.

One key benefit of load balancing is that it provides your application or website with fault tolerance. If any of the servers is unhealthy or encounters a critical error, the load balancer will stop routing to that server and deliver requests to healthy servers instead. This makes your application or website more reliable because it can adapt to failures while still delivering a good user experience.

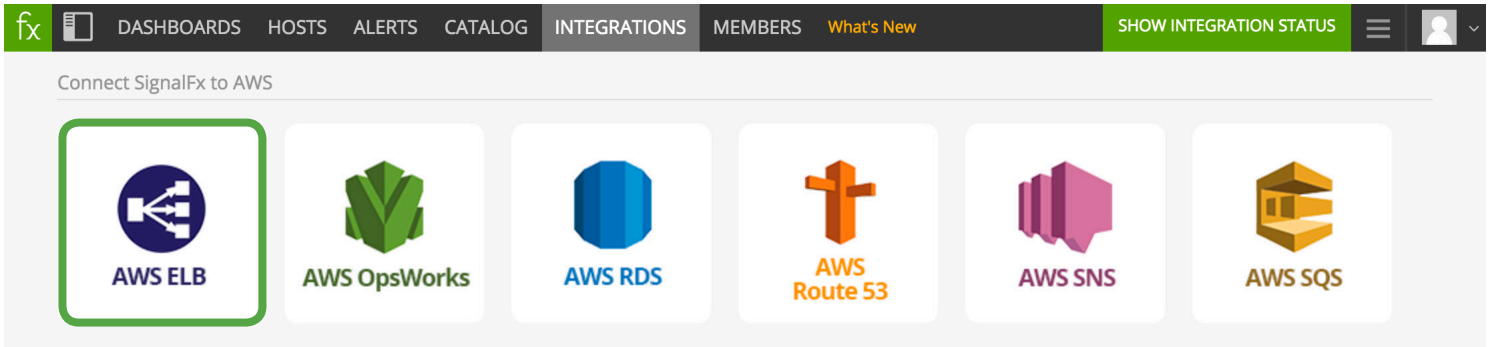
The ability to grow by adding more servers is where ELB gets the name Elastic. You can automatically add more servers based on demand, also known as autoscaling. This can help prevent the “hug of death” when an app experiences a sudden spike in traffic that your existing servers cannot handle. Rather than getting an alert at odd hours of the day to manually add servers, you can rest assured your app will react automatically to deliver good service.

Amazon offers two types of load balancers, each with its own strengths. Classic load balancers have been available for many years. They can balance both HTTPS and TCP traffic, and they have basic health check and routing abilities. Application load balancers (ALBs) are a new type recently released by Amazon. They offer the ability to route requests based on their content, which is great for applications that comprise several containers or microservices per host. ALBs also add support for HTTP/2, WebSockets, and offer enhanced metrics for monitoring.

Monitoring Amazon ELB

Problems with ELB that may cause an outage include configuration errors with the load balancer, network or security settings, and problems with your backend service. Your monitoring tools will give you the information needed to troubleshoot and fix these issues, but the type of data and the speed to insight will vary greatly depending on which tool you use.

An unhealthy ELB can cause your website or application to go offline or slow to a crawl. The right dashboards and meaningful metrics provide insights to remediate issues faster, and a powerful analytics engine makes alerts smarter.



SignalFx offers pre-built content straight out of the box so that you can easily see aggregate data from many load balancers and instances. Going well beyond the basic metrics offered by Amazon CloudWatch, SignalFx data provides rankings and distributions as well as comparisons over time to remove some of the guesswork and eliminate the noise of traditional monitoring.

Amazon ELB Metrics

# LBs	Latency Over Last Minute
LBs with Worst Average Latency (ms)	Total Requests/min
Requests/min	Top LBs by Requests/min
Top Frontend Errors/min	Highest Backend Error %
Top Backend Connection Errors/min	LBs with Highest Unhealthy Host %
Requests/min 7d Change %	Latency 7d Change %

LATENCY OVER LAST MINUTE: It's important to keep a close eye on latency because it's directly tied to user experience. If your load balancing latency is too high, your application or website could frustrate users, and you might lose opportunities and miss SLAs.

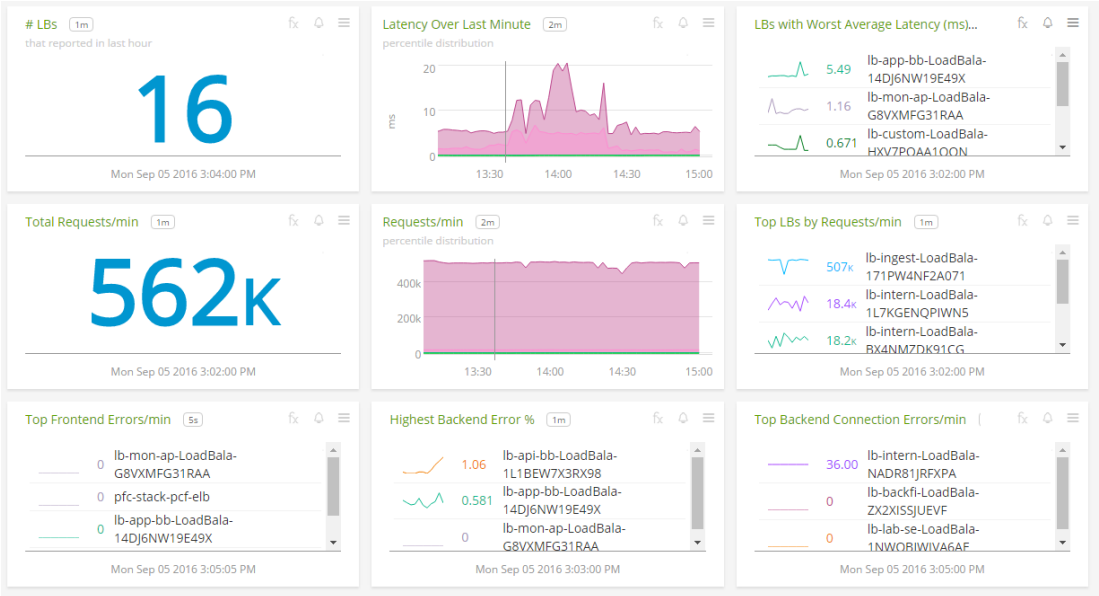
If you only looked at average latency over time, you wouldn't see crucial details about user experience. We recommend visualizing latency as percentiles so you have more insight into the best—and worst—performers. In the SignalFx chart, you can see the maximum latency in dark pink, the 90th percentile in light pink, and the minimum in green. The max latency of 20 seconds spiked around 14:00, but the 90th percentile was flat, indicating that this was a temporary spike affecting less than 10% of requests.

TOTAL REQUESTS/MIN: This chart shows a total sum of requests across all load balancers. An increase in requests per minute might be correlated with an increase in latency. Also, you're probably used to operating your service within a certain range of requests per minute. If this number is way outside of that range, there are likely problems with routing or upstream on the client side.

TOTAL LOAD BALANCERS BY REQUESTS/MIN: When the number of requests is higher than you usually expect, or there's a spike in requests, the next step towards narrowing down the cause is to determine which load balancer is affected. Here you can see that lb-ingest is taking the brunt of the traffic. You actually expect lb-digest to take way more traffic than your lb-intern, so this chart is showing normal behavior.

Alone, any of these Requests/min charts can indicate a moment-in-time issue that could be cause for concern. But looking at them collectively as part of a dashboard and applying solid alerting logic provides a stronger case for identifying a problematic trend and quickly isolating the cause before it leads to a load issue that impacts performance.

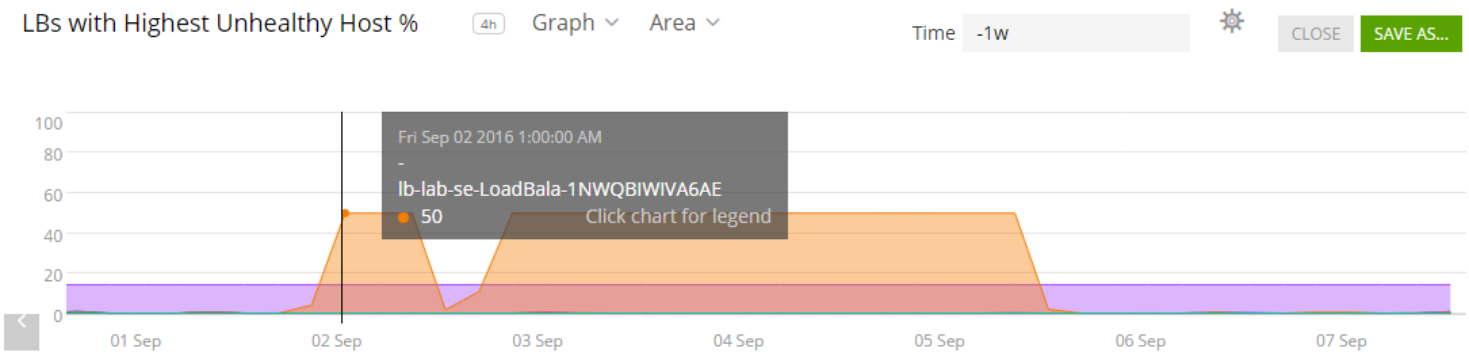
HIGHEST BACKEND ERROR %: Backend errors are defined as errors between the load balancer and the server. When an error code is returned from a backend server call, ELB may retry the call. Additionally, this count includes errors returned during health checks. Thus, the number of backend errors may be higher than the number of frontend errors. In this case we see that the load balancer with the highest percentage of errors is lb-app-bb.



The SignalFx Difference

UNHEALTHY HOST %: One really good predictive measure for load balancers is the unhealthy host percentage. If you fired an alert on a simple threshold of unhealthy host count, your alerts would probably not indicate cluster health when the cluster autoscales over time. A smarter alert would take into account the size of your cluster and calculate the percentage of hosts that are unhealthy.

However, most alerting tools still prioritize health checks and host or node up/down. Anyone who has ever been on-call knows the pain of getting paged at 2 a.m., just to learn the alert was practically useless. SignalFx makes it easy to calculate percentile with a derived metric and set a detector that alerts based on the dynamic conditions of your specific environment.



AGGREGATES & CORRELATIONS: Alone, any Requests/min charts can indicate a moment-in-time issue that could be cause for concern. But looking at multiple collectively as part of a dashboard and applying solid alerting logic provides a stronger case for identifying a problematic trend and quickly isolating the cause before it leads to a load issue that impacts performance.

TRENDS OVER TIME: Latency over the last seven days can be a helpful way to determine if there has been a concerning pattern due to daily or weekly performance. You can compare this with the requests over the same period. Additionally, daily or weekly batch jobs like deployments or cleanup can affect latency. Deployments are especially important to watch because new versions of your service could cause slower or faster performance.

About SignalFx

SignalFx is the most advanced monitoring and alerting solution for modern infrastructure and applications. Our mission is to help cloud-ready organizations drive high levels of availability in today's elastic, agile, distributed environments. With SignalFx, development and operations teams gain a real-time view of, interact with, and take action on the infrastructure and application metrics that matter. We have enterprise customers including Yelp, Cisco, Zuora, and Hubspot and thousands of users analyzing billions of metrics every day. SignalFx was founded in 2013 by former Facebook and VMware executives, launched in 2015, and is backed by Andreessen Horowitz and Charles River Ventures.