Monitoring workloads across AWS services

Your organization uses many different services that AWS provides. You need to monitor these services to ensure successful workloads. However, monitoring the performance of an entire stack of services can get overwhelming.

The majority of AWS workloads behind the scenes are dependent on a core set of services: EC2 (the compute service), EBS (block storage), and ELB (load balancing). For most organizations, these services are at the foundation of their AWS deployments, so understanding each of these services allows you to easily monitor them.

First, you’ll need to follow some steps to get your AWS data integrated with Splunk Infrastructure Monitoring. Then, you’ll need to look at into each of these services and inspect their metrics to understand better how they are performing.

The metrics and services described here are the basics of AWS service monitoring. Depending on your deployment, you may wish to track several other metrics for each service, such as cloud spend.

How to use Splunk software for this use case

Integrate your AWS data

1. Get an access token to connect AWS to Splunk Observability Cloud. With a free trial account, an access token named Default has already been created for you. Alternatively, you can follow these steps to create a new access token.
2. Log into Splunk Observability Cloud and navigate to Data Setup.
3. On the AWS Setup page, select New integration to open the AWS integration wizard.
4. Click + Add Connection to configure an integration for one of your AWS accounts and follow the steps needed to create your connection. After it’s connected, Splunk Infrastructure Monitoring lists all of your AWS services.
5. Navigate to the Infrastructure page and select Amazon Web Services to see a list of all AWS resources in a single pane of glass.

Inspect the services and their metrics

EC2, EBS and ELB monitoring within Splunk Infrastructure Monitoring work the same way. Splunk Infrastructure Monitoring provides you with an overview of each of these volumes, color-coding key metrics for you to filter and choose from. You can group all of your volumes by common characteristics, such as such as region, state, or OS type. After you have identified problematic volumes, you can click to drill down and gather specific information about them.

EC2 Metrics

The EC2 compute service lets you run virtual machines in the AWS cloud, although there are a few bare-metal EC2 instance types available too. If you host any kind of application or service in AWS, it likely runs on EC2. Even if you host...
it in a service like EKS, the AWS Kubernetes platform, in most cases it’s still running on an EC2 instance.

Splunk Infrastructure Monitoring provides you with an overview of all your EC2 metrics, as well as displaying your Kubernetes deployment with Kubernetes navigator.

There are three key metrics to track for each EC2 instance:

- **CPU Utilization**: The total number of CPU units used, expressed as a percentage of the total available. If this metric exceeds about 80 percent for more than a brief period, you’ll want to investigate whether you need to increase the CPU capacity allocated to your workload. Or, there may be a problem with your application that is causing excessive CPU usage.

- **DiskReadOps**: The total completed read operations by the EC2 instance in a given period of time. When this metric deviates from the historical baseline average, it could signify that something is wrong with the application running inside the instance.

- **DiskWriteOps**: The total completed write operations by the EC2 instance in a given period of time. Like spikes in DiskReadOps, DiskWriteOps data that deviates from the norm could signal an application problem.

**EBS Metrics**

EBS is Amazon’s solution for workloads that require block-level storage. EBS volumes tend to be especially important as storage for EC2 instances.

There are three key metrics to track for each EBS instance:

- **Volume State (aws_state)**: AWS performs health checks on EBS volumes and returns a status in the form of one of the following: creating, available, in-use, deleting, deleted, or error. If the volume state is showing error, it may be best to investigate. Other states to consider investigating are available, deleting or deleted, depending on the scenario.

- **Total IOPS**: This is the total read and write operations in a set period of time. High metrics beyond your normal baseline can indicate application bottlenecks or poor storage selection.

- **Average Queue Length**: Volume queue length is the number of pending I/O requests measured by its latency. This latency shows the time elapsed between sending an I/O to EBS and receiving an acknowledgment from EBS that the I/O read or write is complete. High latency on an EBS volume might show the need for a possible well-suited volume such as an SSD-backed volume.

**ELB Metrics**

ELB, AWS’s load balancing service, offers several types of load balancers that distribute application traffic across different EC2 instances.

There are three key metrics to track for each ELB instance:

- **Request Count**: This is the total requests that ELB handles in a set period. While it’s natural for request counts to vary as demand for your application ebbs and flows, sudden spikes or decreases inconsistent with historical traffic patterns at a specific time of day or day of the week could signal a problem like the inability of users to reach your application.

- **Latency**: A measure of the time it takes for one of your instances to start the response to a request from ELB. High latency could be a sign of problems such as an issue with the network or an under-provisioned EC2 instance.
struggling to handle all of its requests.

- **Unhealthy Host Count**: ELB performs health checks on instances and uses this metric to count those that it deems unhealthy, meaning that they are not ready to handle requests and may be down. Monitor this metric to ensure you don’t run out of sufficient healthy instances to handle application demand.

---

**Next steps**

The content in this guide comes from [a previously published blog](#), one of the thousands of Splunk resources available to help users succeed. In addition, these Splunk resources might help you understand and implement this use case:

- .Conf Talk: [Best practices for deploying Splunk on AWS](#)
- .Conf Talk: [Splunk + AWS: Two great things even better together](#)
- Blog: [How to optimize your cloud spend using Observability](#)