

Table of Contents

Overview

[Monitoring tools across Azure](#)

[Azure Monitor](#)

[Metrics](#)

[Alerts](#)

[Autoscale](#)

[Activity log](#)

[Action Groups](#)

[Diagnostic Logs](#)

[Partner Integrations](#)

[Azure Diagnostics Extension](#)

Get Started

[Get Started with Azure Monitor](#)

[Get Started with Autoscale](#)

[Roles Permissions and Security](#)

How to

[Use alerts](#)

[Configure alerts in Azure portal](#)

[Configure alerts with CLI](#)

[Configure alerts with PowerShell](#)

[Have a metric alert call a webhook](#)

[Create a metric alert with a Resource Manager template](#)

[Use autoscale](#)

[Best Practices](#)

[Common metrics](#)

[Common patterns](#)

[Autoscale using a custom metric](#)

[Autoscale VM Scale sets using Resource Manager templates](#)

[Automatically scale machines in a VM Scale set](#)

[Configure webhooks and email notifications on autoscale](#)

[Use the activity log](#)

[View events in activity log](#)

[Configure alerts on an activity log event](#)

[Archive activity log](#)

[Stream activity log to Event Hubs](#)

[Audit operations with Resource Manager](#)

[Create activity log alerts with Resource Manager](#)

[Use Service notifications](#)

[View service notifications](#)

[Configure alerts on service notifications](#)

[Use Action Groups](#)

[Learn about webhook schema](#)

[SMS Alert behavior](#)

[Alert Rate limiting](#)

[Create action groups with Resource Manager](#)

[Manage diagnostic logs](#)

[Archive](#)

[Stream to Event Hubs](#)

[Enable Diagnostic settings using Resource Manager templates](#)

[Use the REST API](#)

[Walkthrough using REST API](#)

[Use Azure Diagnostics extension](#)

[Send to Application Insights](#)

[Send to Event Hubs](#)

[Troubleshooting](#)

[Reference](#)

[Code samples](#)

[Sources of monitoring data](#)

[List of supported metrics](#)

[Activity Log event schema](#)

[PowerShell](#)

[.NET](#)

[REST](#)

[Azure Diagnostics extension schema](#)

[1.0](#)

[1.2](#)

[1.3 and later](#)

[Resources](#)

[Azure CLI 1.0 samples](#)

[Azure Roadmap](#)

[PowerShell samples](#)

[Pricing calculator](#)

[Videos](#)

[Quickstart templates](#)

Overview of Monitoring in Microsoft Azure

6/27/2017 • 5 min to read • [Edit Online](#)

This article provides an overview of tools available for monitoring Microsoft Azure. It applies to

- monitoring applications running in Microsoft Azure
- tools/services that run outside of Azure that can monitor objects in Azure.

It discusses the various products and services available and how they work together. It can assist you to determine which tools are most appropriate for you in what cases.

Why use Monitoring and Diagnostics?

Performance issues in your cloud app can impact your business. With multiple interconnected components and frequent releases, degradations can happen at any time. And if you're developing an app, your users usually discover issues that you didn't find in testing. You should know about these issues immediately, and have tools for diagnosing and fixing the problems. Microsoft Azure has a range of tools for identifying these problems.

How do I monitor my Azure cloud apps?

There is a range of tools for monitoring Azure applications and services. Some of their features overlap. This is partly for historical reasons and partly due to the blurring between development and operation of an application.

Here are the principal tools:

- **Azure Monitor** is basic tool for monitoring services running on Azure. It gives you infrastructure-level data about the throughput of a service and the surrounding environment. If you are managing your apps all in Azure, deciding whether to scale up or down resources, then Azure Monitor gives you what you use to start.
- **Application Insights** can be used for development and as a production monitoring solution. It works by installing a package into your app, and so gives you a more internal view of what's going on. Its data includes response times of dependencies, exception traces, debugging snapshots, execution profiles. It provides powerful smart tools for analyzing all this telemetry both to help you debug an app and to help you understand what users are doing with it. You can tell whether a spike in response times is due to something in an app, or some external resourcing issue. If you use Visual Studio and the app is at fault, you can be taken right to the problem line(s) of code so you can fix it.
- **Log Analytics** is for those who need to tune performance and plan maintenance on applications running in production. It is based in Azure. It collects and aggregates data from many sources, though with a delay of 10 to 15 minutes. It provides a holistic IT management solution for Azure, on-premises, and third-party cloud-based infrastructure (such as Amazon Web Services). It provides richer tools to analyze data across more sources, allows complex queries across all logs, and can proactively alert on specified conditions. You can even collect custom data into its central repository so can query and visualize it.
- **System Center Operations Manager (SCOM)** is for managing and monitoring large cloud installations. You might be already familiar with it as a management tool for on-premises Windows Server and Hyper-V based-clouds, but it can also integrate with and manage Azure apps. Among other things, it can install Application Insights on existing live apps. If an app goes down, it tells you in seconds. Note that Log Analytics does not replace SCOM. It works well in conjunction with it.

Accessing monitoring in the Azure portal

All Azure monitoring services are now available in a single UI pane. For more information on how to access this area, see [Get started with Azure Monitor](#).

You can also access monitoring functions for specific resources by highlighting those resources and drilling down into their monitoring options.

Examples of when to use which tool

The following sections show some basic scenarios and which tools should be used together.

Scenario 1 – Fix errors in an Azure Application under development

The best option is to use Application Insights, Azure Monitor, and Visual Studio together

Azure now provides the full power of the Visual Studio debugger in the cloud. Configure Azure Monitor to send telemetry to Application Insights. Enable Visual Studio to include the Application Insights SDK in your application. Once in Application Insights, you can use the Application Map to discover visually which parts of your running application are unhealthy or not. For those parts that are not healthy, errors and exceptions are already available for exploration. You can use the various analytics in Application Insights to go deeper. If you are not sure about the error, you can use the Visual Studio debugger to trace into code and pin point a problem further.

For more information, see [Monitoring Web Apps](#) and refer to the table of contents on the left for instructions on various types of apps and languages.

Scenario 2 – Debug an Azure .NET web application for errors that only show in production

NOTE

These features are in preview.

The best option is to use Application Insights and if possible Visual Studio for the full debugging experience.

Use the Application Insights Snapshot Debugger to debug your app. When a certain error threshold occurs with production components, the system automatically captures telemetry in windows of time called "snapshots." The amount captured is safe for a production cloud because it's small enough not to affect performance but significant enough to allow tracing. The system can capture multiple snapshots. You can look at a point in time in the Azure portal or use Visual Studio for the full experience. With Visual Studio, developers can walk through that snapshot as if they were debugging in real-time. Local variables, parameters, memory, and frames are all available. Developers must be granted access to this production data via an RBAC role.

For more information, see [Snapshot debugging](#).

Scenario 3 – Debug an Azure application that uses containers or microservices

Same as scenario 1. Use Application Insights, Azure Monitor, and Visual Studio together Application Insights also supports gathering telemetry from processes running inside containers and from microservices (Kubernetes, Docker, Azure Service Fabric). For more information, [see this video on debugging containers and microservices](#).

Scenario 4 – Fix performance issues in your Azure application

The [Application Insights profiler](#) is designed to help troubleshoot these types of issues. You can identify and troubleshoot performance issues for applications running in App Services (Web Apps, Logic Apps, Mobile Apps, API Apps) and other compute resources such as Virtual Machines, Virtual machine scale sets (VMSS), Cloud Services, and Service Fabric.

NOTE

Ability to profile Virtual Machines, Virtual machine scale sets (VMSS), Cloud Services and Services Fabric is in preview.

In addition, you are proactively notified by email about certain types of errors, such as slow page load times, by the Smart Detection tool. You don't need to do any configuration on this tool. For more information, see [Smart Detection - Performance Anomalies](#) and [Smart Detection - Performance Anomalies](#).

Next steps

Learn more about

- [Azure Monitor in a video from Ignite 2016](#)
- [Getting Started with Azure Monitor](#)
- [Azure Diagnostics](#) if you are attempting to diagnose problems in your Cloud Service, Virtual Machine, Virtual machine scale set, or Service Fabric application.
- [Application Insights](#) if you are trying to diagnostic problems in your App Service Web app.
- [Log Analytics](#) and the [Operations Management Suite](#) production monitoring solution

Overview of Azure Monitor

6/29/2017 • 5 min to read • [Edit Online](#)

This article provides an overview of the Azure Monitor service in Microsoft Azure. It discusses what Azure Monitor does and provides pointers to additional information on how to use Azure Monitor. If you prefer a video introduction, see Next steps links at the bottom of this article.

Why monitor your application or system

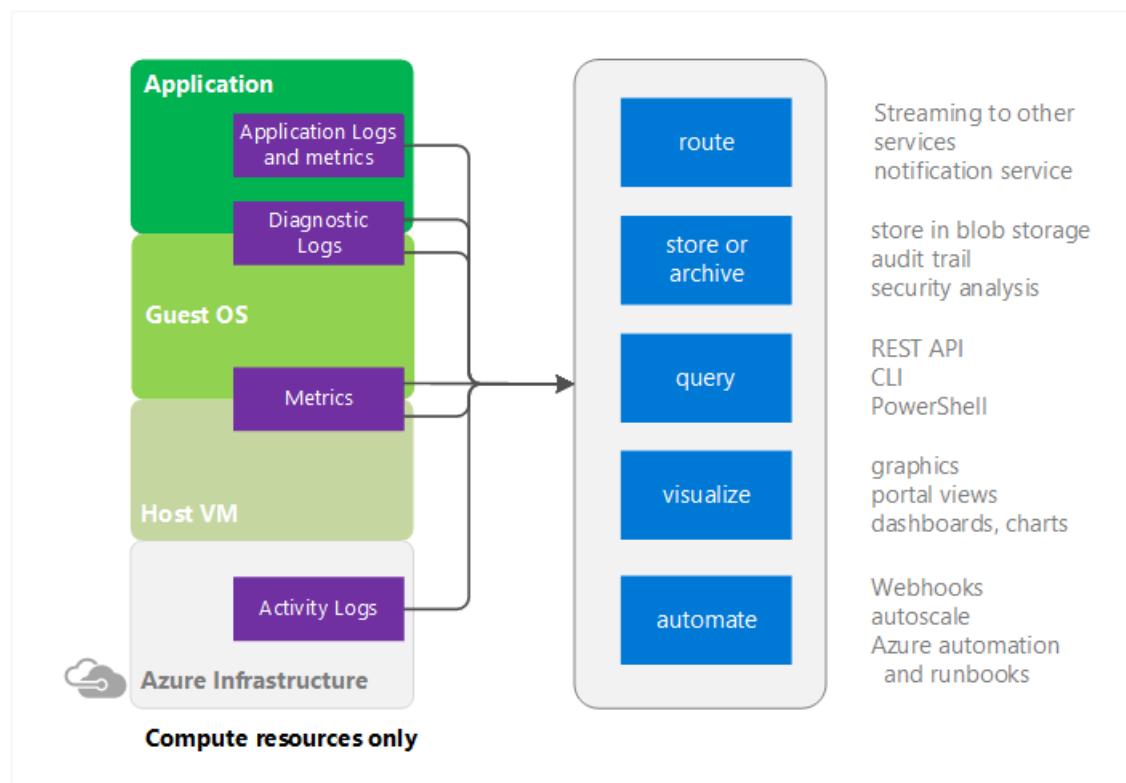
Cloud applications are complex with many moving parts. Monitoring provides data to ensure that your application stays up and running in a healthy state. It also helps you to stave off potential problems or troubleshoot past ones. In addition, you can use monitoring data to gain deep insights about your application. That knowledge can help you to improve application performance or maintainability, or automate actions that would otherwise require manual intervention.

Azure Monitor and Microsoft's other monitoring products

Azure Monitor provides base level infrastructure metrics and logs for most services in Microsoft Azure. Azure services that do not yet put their data into Azure Monitor will put it there in the future.

Microsoft ships additional products and services that provide additional monitoring capabilities for developers, DevOps, or IT Ops that also have on-premises installations. For an overview and understanding of how these different products and services work together, see [Monitoring in Microsoft Azure](#).

Monitoring Sources - Compute



The Compute services include

- Cloud Services

- Virtual Machines
- Virtual Machine scale sets
- Service Fabric

Application - Diagnostics Logs, Application Logs, and Metrics

Applications can run on top of the Guest OS in the compute model. They emit their own set of logs and metrics. Azure Monitor relies on the Azure diagnostics extension (Windows or Linux) to collect most application level metrics and logs. The types include

- Performance counters
- Application Logs
- Windows Event Logs
- .NET Event Source
- IIS Logs
- Manifest based ETW
- Crash Dumps
- Customer Error Logs

Without the diagnostics extension, only a few metrics like CPU usage are available.

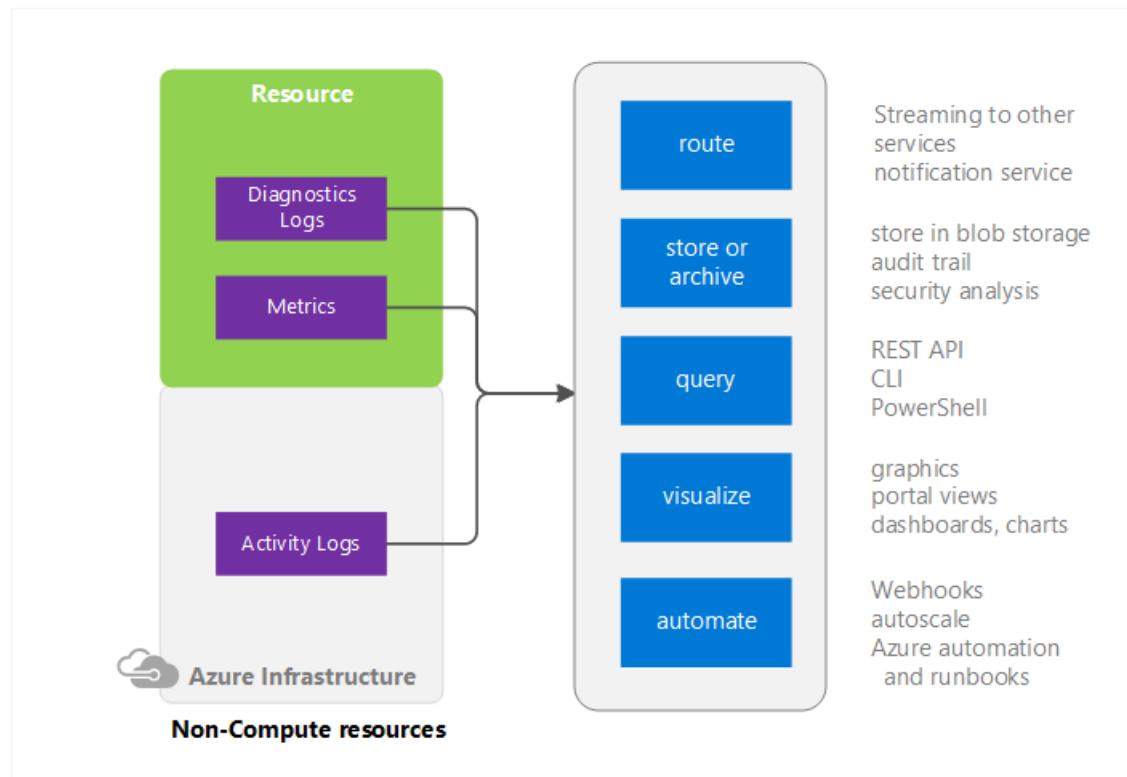
Host and Guest VM metrics

The previously listed compute resources have a dedicated host VM and guest OS they interact with. The host VM and guest OS are the equivalent of root VM and guest VM in the Hyper-V hypervisor model. You can collect metrics on both. You can also collect diagnostics logs on the guest OS.

Activity Log

You can search the Activity Log (previously called Operational or Audit Logs) for information about your resource as seen by the Azure infrastructure. The log contains information such as times when resources are created or destroyed. For more information, see [Overview of Activity Log](#).

Monitoring Sources - everything else



Resource - Metrics and Diagnostics Logs

Collectable metrics and diagnostics logs vary based on the resource type. For example, Web Apps provides statistics on the Disk IO and Percent CPU. Those metrics don't exist for a Service Bus queue, which instead provides metrics like queue size and message throughput. A list of collectable metrics for each resource is available at [supported metrics](#).

Host and Guest VM metrics

There is not necessarily a 1:1 mapping between your resource and a particular Host or Guest VM so metrics are not available.

Activity Log

The activity log is the same as for compute resources.

Uses for Monitoring Data

Once you collect your data, you can do the following with it in Azure Monitor

Route

You can stream monitoring data to other locations in real time.

Examples include:

- Send to Application Insights so you can use its richer visualization and analysis tools.
- Send to Event Hubs so you can route to third-party tools.

Store and Archive

Some monitoring data is already stored and available in Azure Monitor for a set amount of time.

- Metrics are stored for 30 days.
- Activity log entries are stored for 90 days.
- Diagnostics logs are not stored at all.

If you want to store data longer than the time periods listed above, you can use an Azure storage. Monitoring data is kept in your storage account based on a retention policy you set. You do have to pay for the space the data takes up in Azure storage.

A few ways to use this data:

- Once written, you can have other tools within or outside of Azure read it and process it.
- You download the data locally for a local archive or change your retention policy in the cloud to keep data for extended periods of time.
- You leave the data in Azure storage indefinitely for archive purposes.

Query

You can use the Azure Monitor REST API, cross platform Command-Line Interface (CLI) commands, PowerShell cmdlets, or the .NET SDK to access the data in the system or Azure storage

Examples include:

- Getting data for a custom monitoring application you have written
- Creating custom queries and sending that data to a third-party application.

Visualize

Visualizing your monitoring data in graphics and charts helps you find trends quicker than looking through the data itself.

A few visualization methods include:

- Use the Azure portal
- Route data to Azure Application Insights
- Route data to Microsoft PowerBI
- Route the data to a third-party visualization tool using either live streaming or by having the tool read from an archive in Azure storage

Automate

You can use monitoring data to trigger alerts or even whole processes. Examples include:

- Use data to autoscale compute instances up or down based on application load.
- Send emails when a metric crosses a predetermined threshold.
- Call a web URL (webhook) to execute an action in a system outside of Azure
- Start a runbook in Azure automation to perform any variety of tasks

Methods of accessing Azure Monitor

In general, you can manipulate data tracking, routing, and retrieval using one of the following methods. Not all methods are available for all actions or data types.

- [Azure portal](#)
- [PowerShell](#)
- [Cross-platform Command Line Interface \(CLI\)](#)
- [REST API](#)
- [.NET SDK](#)

Next steps

Learn more about

- A video walkthrough of just Azure Monitor is available at [Get Started with Azure Monitor](#). An additional video explaining a scenario where you can use Azure Monitor is available at [Explore Microsoft Azure monitoring and diagnostics](#) and [Azure Monitor in a video from Ignite 2016](#)
- Run through the Azure Monitor interface in [Getting Started with Azure Monitor](#)
- Set up the [Azure Diagnostics Extensions](#) if you are attempting to diagnose problems in your Cloud Service, Virtual Machine, Virtual machine scale sets, or Service Fabric application.
- [Application Insights](#) if you are trying to diagnostic problems in your App Service Web app.
- [Troubleshooting Azure Storage](#) when using Storage Blobs, Tables, or Queues
- [Log Analytics](#) and the [Operations Management Suite](#)

Overview of metrics in Microsoft Azure

7/18/2017 • 5 min to read • [Edit Online](#)

This article describes what metrics are in Microsoft Azure, their benefits, and how to start using them.

What are metrics?

Azure Monitor enables you to consume telemetry to gain visibility into the performance and health of your workloads on Azure. The most important type of Azure telemetry data is the metrics (also called performance counters) emitted by most Azure resources. Azure Monitor provides several ways to configure and consume these metrics for monitoring and troubleshooting.

What can you do with metrics?

Metrics are a valuable source of telemetry and enable you to do the following tasks:

- **Track the performance** of your resource (such as a VM, website, or logic app) by plotting its metrics on a portal chart and pinning that chart to a dashboard.
- **Get notified of an issue** that impacts the performance of your resource when a metric crosses a certain threshold.
- **Configure automated actions**, such as autoscaling a resource or firing a runbook when a metric crosses a certain threshold.
- **Perform advanced analytics** or reporting on performance or usage trends of your resource.
- **Archive** the performance or health history of your resource **for compliance or auditing** purposes.

What are the characteristics of metrics?

Metrics have the following characteristics:

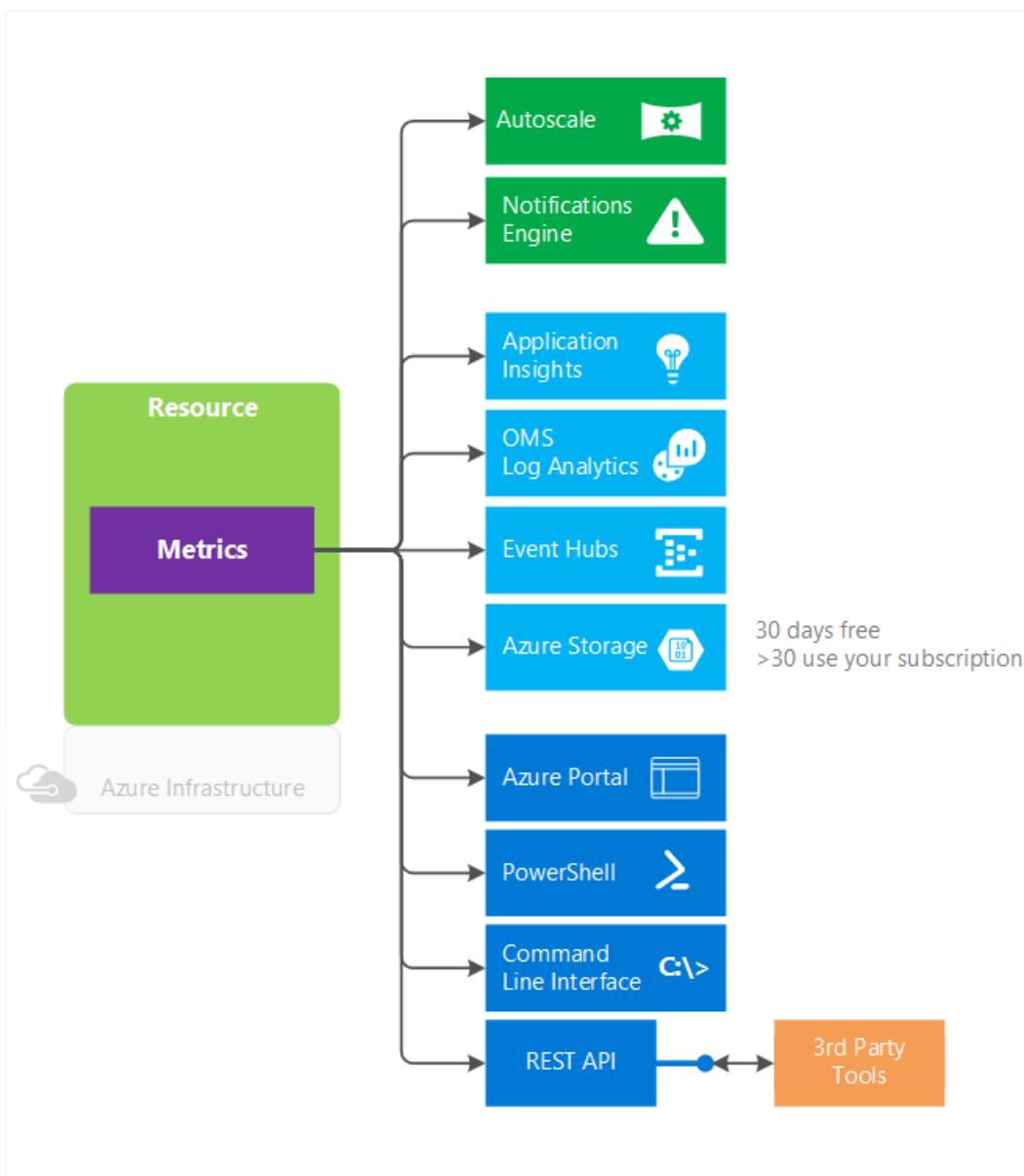
- All metrics have **one-minute frequency**. You receive a metric value every minute from your resource, giving you near real-time visibility into the state and health of your resource.
- Metrics are **available immediately**. You don't need to opt in or set up additional diagnostics.
- You can access **30 days of history** for each metric. You can quickly look at the recent and monthly trends in the performance or health of your resource.

You can also:

- Configure a metric **alert rule that sends a notification or takes automated action** when the metric crosses the threshold that you have set. Autoscale is a special automated action that enables you to scale out your resource to meet incoming requests or loads on your website or computing resources. You can configure an Autoscale setting rule to scale in or out based on a metric crossing a threshold.
- **Route** all metrics Application Insights or Log Analytics (OMS) to enable instant analytics, search, and custom alerting on metrics data from your resources. You can also stream metrics to an Event Hub, enabling you to then route them to Azure Stream Analytics or to custom apps for near-real time analysis. You set up Event Hub streaming using diagnostic settings.
- **Archive metrics to storage** for longer retention or use them for offline reporting. You can route your metrics to Azure Blob storage when you configure diagnostic settings for your resource.
- Easily discover, access, and **view all metrics** via the Azure portal when you select a resource and plot the

metrics on a chart.

- **Consume** the metrics via the new Azure Monitor REST APIs.
- **Query** metrics by using the PowerShell cmdlets or the Cross-Platform REST API.



Access metrics via the portal

Following is a quick walkthrough of how to create a metric chart by using the Azure portal.

To view metrics after creating a resource

1. Open the Azure portal.
2. Create an Azure App Service website.
3. After you create a website, go to the **Overview** blade of the website.
4. You can view new metrics as a **Monitoring** tile. You can then edit the tile and select more metrics.

To access all metrics in a single place

1. Open the Azure portal.
2. Navigate to the new **Monitor** tab, and then select the **Metrics** option underneath it.
3. Select your subscription, resource group, and the name of the resource from the drop-down list.
4. View the available metrics list. Then select the metric you are interested in and plot it.
5. You can pin it to the dashboard by clicking the pin on the upper-right corner.

NOTE

You can access host-level metrics from VMs (Azure Resource Manager-based) and virtual machine scale sets without any additional diagnostic setup. These new host-level metrics are available for Windows and Linux instances. These metrics are not to be confused with the Guest-OS-level metrics that you have access to when you turn on Azure Diagnostics on your VMs or virtual machine scale sets. To learn more about configuring Diagnostics, see [What is Microsoft Azure Diagnostics](#).

Access metrics via the REST API

Azure Metrics can be accessed via the Azure Monitor APIs. There are two APIs that help you discover and access metrics:

- Use the [Azure Monitor Metric definitions REST API](#) to access the list of metrics that are available for a service.
- Use the [Azure Monitor Metrics REST API](#) to access the actual metrics data.

NOTE

This article covers the metrics via the [new API for metrics](#) for Azure resources. The API version for the new metric definitions API is 2016-03-01 and the version for metrics API is 2016-09-01. The legacy metric definitions and metrics can be accessed with the API version 2014-04-01.

For a more detailed walkthrough using the Azure Monitor REST APIs, see [Azure Monitor REST API walkthrough](#).

Export metrics

You can go to the **Diagnostics settings** blade under the **Monitor** tab and view the export options for metrics. You can select metrics (and diagnostic logs) to be routed to Blob storage, to Azure Event Hubs, or to OMS for use-cases that were mentioned previously in this article.

The screenshot shows two overlapping windows from the Azure portal. The left window is titled 'Diagnostics settings' and contains configuration for metrics and logs. It has sections for 'Status' (On), 'Archive to a storage account' (unchecked), 'Stream to an event hub' (checked), 'Service bus namespace' (loanappsb1 (rootmanagesharedaccesskey)), 'Send to Log Analytics' (checked), 'Log Analytics' (loanappanalytics), 'LOG' (WorkflowRuntime checked), and 'METRIC' (5 minutes checked). The right window is titled 'OMS Workspaces' and lists existing workspaces: ashoms (eastus), demoAzMonWS (australiasoutheast), johnkemtest (eastus), JohnKemTestOMS (eastus), LoanAppAnalytics (eastus), loandemows (australiasoutheast), viruela1 (eastus), viruela3 (australiasoutheast), contoso1demows (australiasoutheast), and demoappgw (eastus). A 'Create New Workspace' button is also visible.

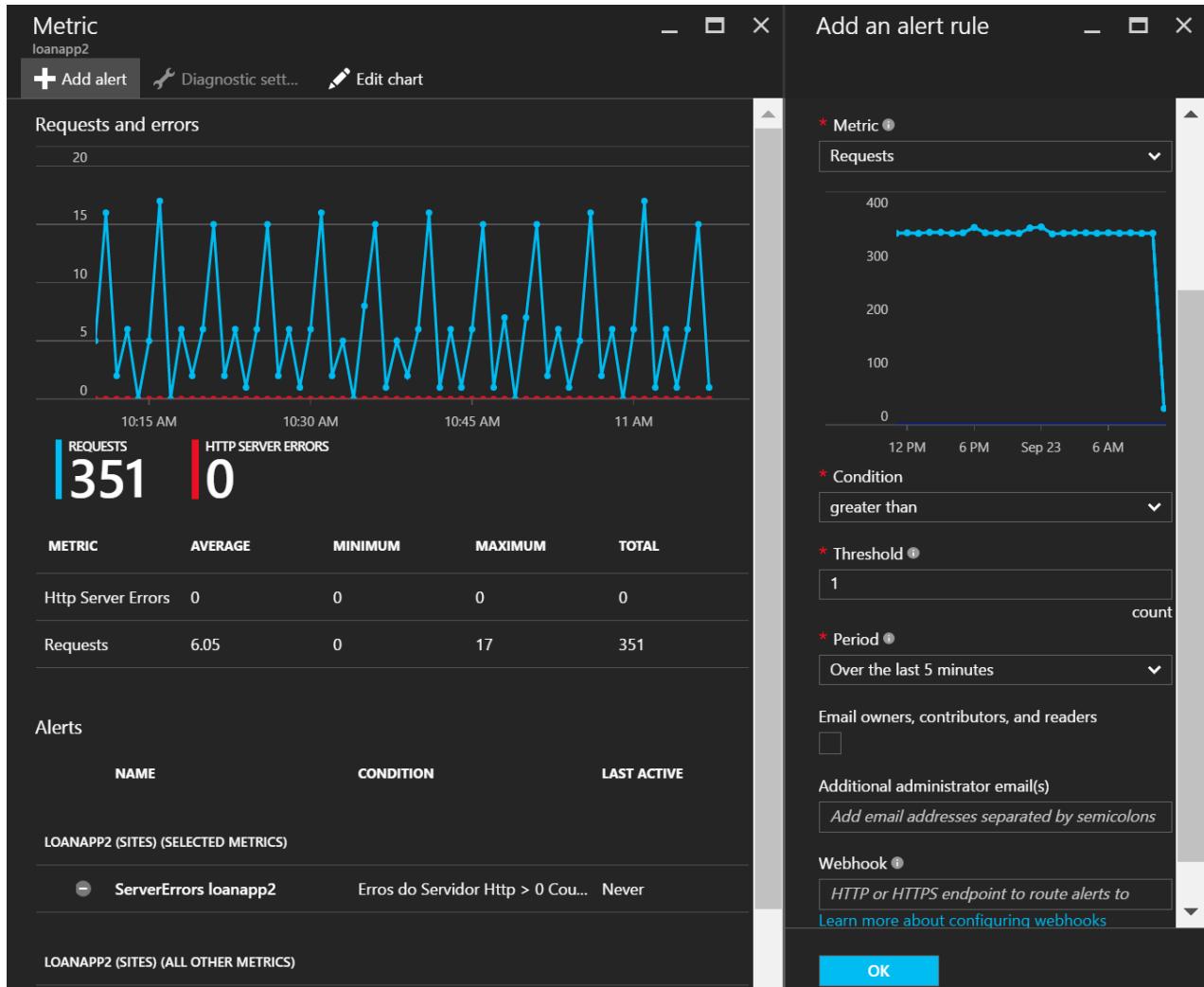
You can configure this via Resource Manager templates, [PowerShell](#), [Azure CLI](#), or [REST APIs](#).

Take action on metrics

To receive notifications or take automated actions on metric data, you can configure alert rules or Autoscale settings.

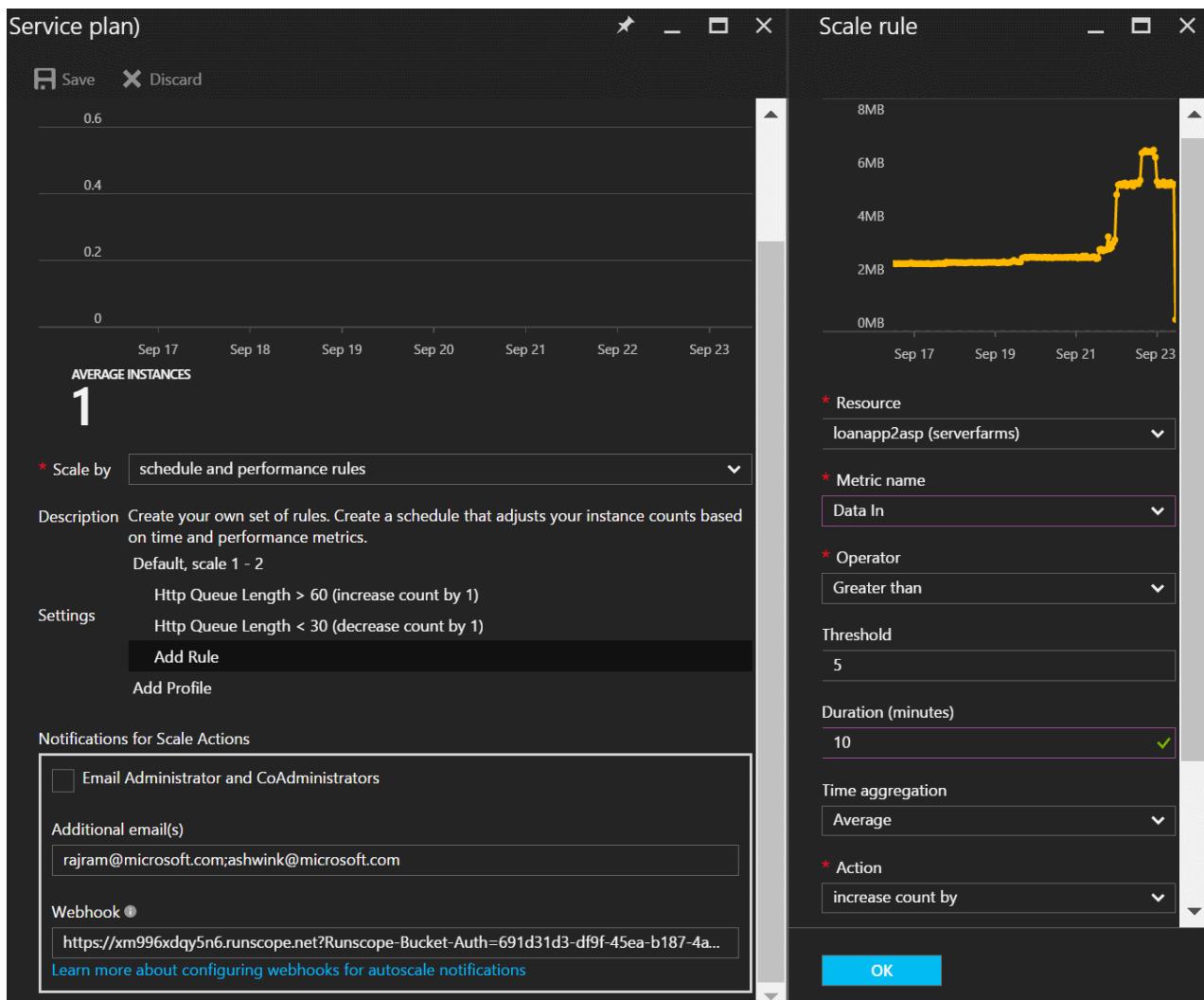
Configure alert rules

You can configure alert rules on metrics. These alert rules can check if a metric has crossed a certain threshold. They can then notify you via email or fire a webhook that can be used to run any custom script. You can also use the webhook to configure third-party product integrations.



Autoscale your Azure resources

Some Azure resources support the scaling out or in of multiple instances to handle your workloads. Autoscale applies to App Service (Web Apps), virtual machine scale sets, and classic Azure Cloud Services. You can configure Autoscale rules to scale out or in when a certain metric that impacts your workload crosses a threshold that you specify. For more information, see [Overview of autoscaling](#).



Learn about supported services and metrics

Azure Monitor is a new metrics infrastructure. It supports the following Azure services in the Azure portal and the new version of the Azure Monitor API:

- VMs (Azure Resource Manager-based)
- Virtual machine scale sets
- Batch
- Event Hubs namespace
- Service Bus namespace (premium SKU only)
- SQL Database (version 12)
- Elastic SQL Pool
- Websites
- Web server farms
- Logic Apps
- IoT hubs
- Redis Cache
- Networking: Application gateways
- Search

You can view a detailed list of all the supported services and their metrics at [Azure Monitor metrics--supported metrics per resource type](#).

Next steps

Refer to the links throughout this article. Additionally, learn about:

- [Common metrics for autoscaling](#)
- [How to create alert rules](#)
- [Analyze logs from Azure storage with Log Analytics](#)

What are alerts in Microsoft Azure?

7/18/2017 • 2 min to read • [Edit Online](#)

This article describes what alerts are, their benefits, and how to get started with using them. It specifically applies to Azure Monitor, but provides pointers to other services.

Alerts are a method of monitoring Azure resource metrics, events, or logs and being notified when a condition you specify is met.

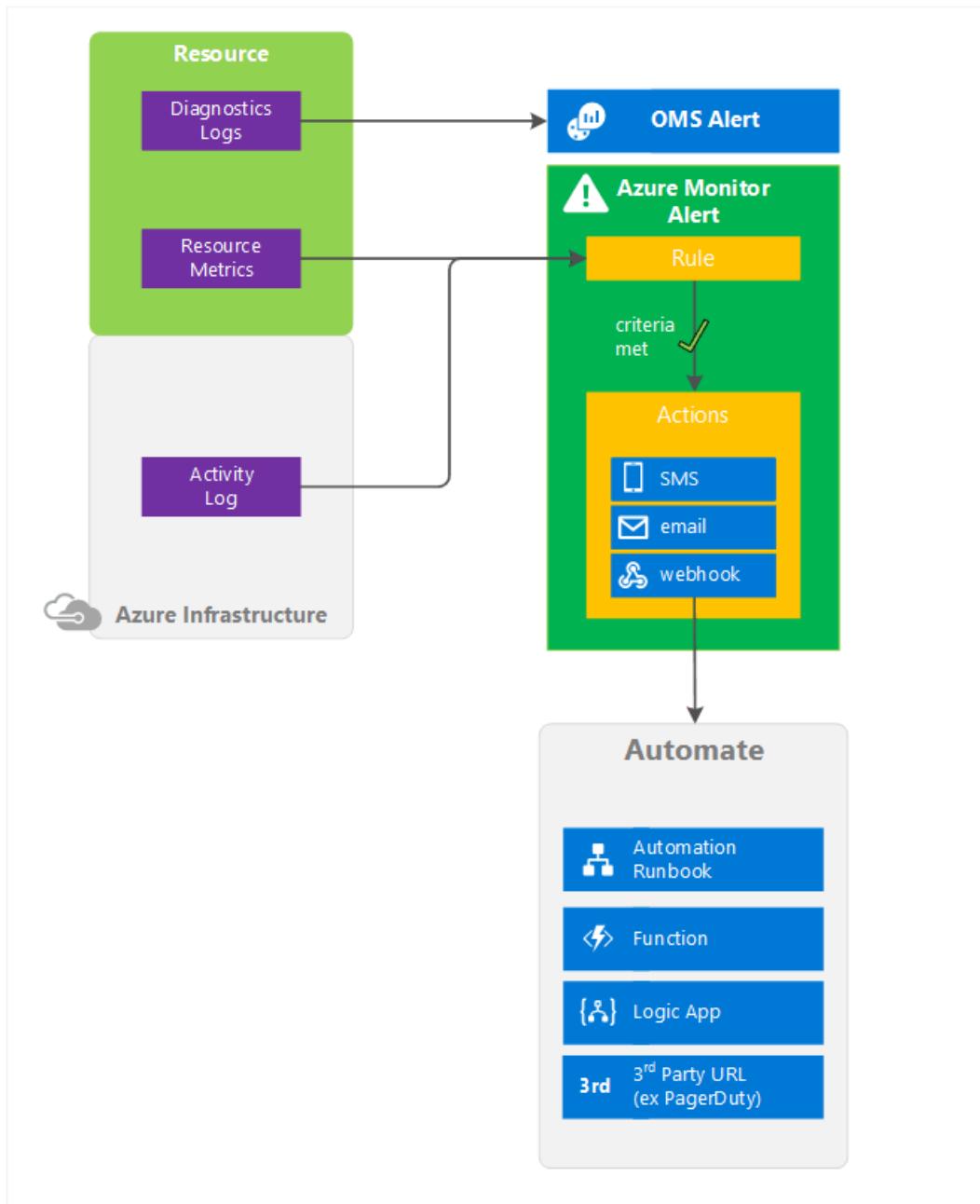
Alerts in different Azure services

Alerts are available across different services, including:

- **Application Insights:** Enables web test and metric alerts. See [Set alerts in Application Insights](#) and [Monitor availability and responsiveness of any website](#).
- **Log Analytics (Operations Management Suite):** Enables the routing of Activity and Diagnostic Logs to Log Analytics. Operations Management Suite allows metric, log, and other alert types. For more information, see [Alerts in Log Analytics](#).
- **Azure Monitor:** Enables alerts based on both metric values and activity log events. You can use the [Azure Monitor REST API](#) to manage alerts. For more information, see [Using the Azure portal, PowerShell, or the command-line interface to create alerts](#).

Visual Summary

The following diagram summarizes alerts and what you can do with them specifically in "Azure Monitor". Other actions may be available for the services listed previously. For example, currently alerts on Diagnostics Logs are only available in Log Analytics.



What can trigger alerts in Azure Monitor

You can receive alerts based on:

- **Metric values:** This alert triggers when the value of a specified metric crosses a threshold that you assign in either direction. That is, it triggers both when the condition is first met and then afterward when that condition is no longer being met. For a growing list of available metrics supported by Azure monitor, see [List of metrics supported on Azure Monitor](#).
- **Activity log events:** This alert can trigger when a particular event occurs on a resource, or when a service notification is posted to your subscription.

What can Metric Alerts do?

You can configure an alert to do the following actions:

- Send email notifications to the service administrator, to co-administrators, or to additional email addresses that you specify.
- Call a webhook, which enables you to launch additional automation actions. Examples include calling:
 - Azure Automation Runbook

- Azure Function
- Azure Logic App
- a third-party service

What can Activity Log Alerts do?

You can configure an alert to do the following actions:

- Trigger whenever a specific event occurs one of the resources under your subscription
- Trigger whenever a service notification is posted to your subscription
- Alert members of an action group via
 - SMS
 - Email
 - Webhook

Next steps

Get information about alert rules and configuring them by using:

- Learn more about [Metrics](#)
- Configure [Metric Alerts via Azure portal](#)
- Configure [Metric Alerts PowerShell](#)
- Configure [Metric Alerts Command-line interface \(CLI\)](#)
- Configure [Metric Alerts Azure Monitor REST API](#)
- Learn more about [Activity Log](#)
- Configure [Activity Log Alerts via Azure portal](#)
- Configure [Activity Log Alerts via Resource Manager](#)
- Review the [activity log alert webhook schema](#)
- Learn more about [Service Notifications](#)
- Learn more about [Action Groups](#)

Overview of autoscale in Microsoft Azure Virtual Machines, Cloud Services, and Web Apps

7/18/2017 • 4 min to read • [Edit Online](#)

This article describes what Microsoft Azure autoscale is, its benefits, and how to get started using it.

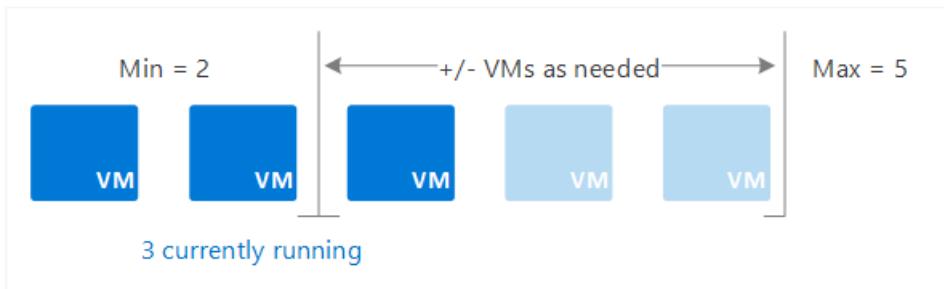
Azure Monitor autoscale applies only to [Virtual Machine Scale Sets](#), [Cloud Services](#), and [App Service - Web Apps](#).

NOTE

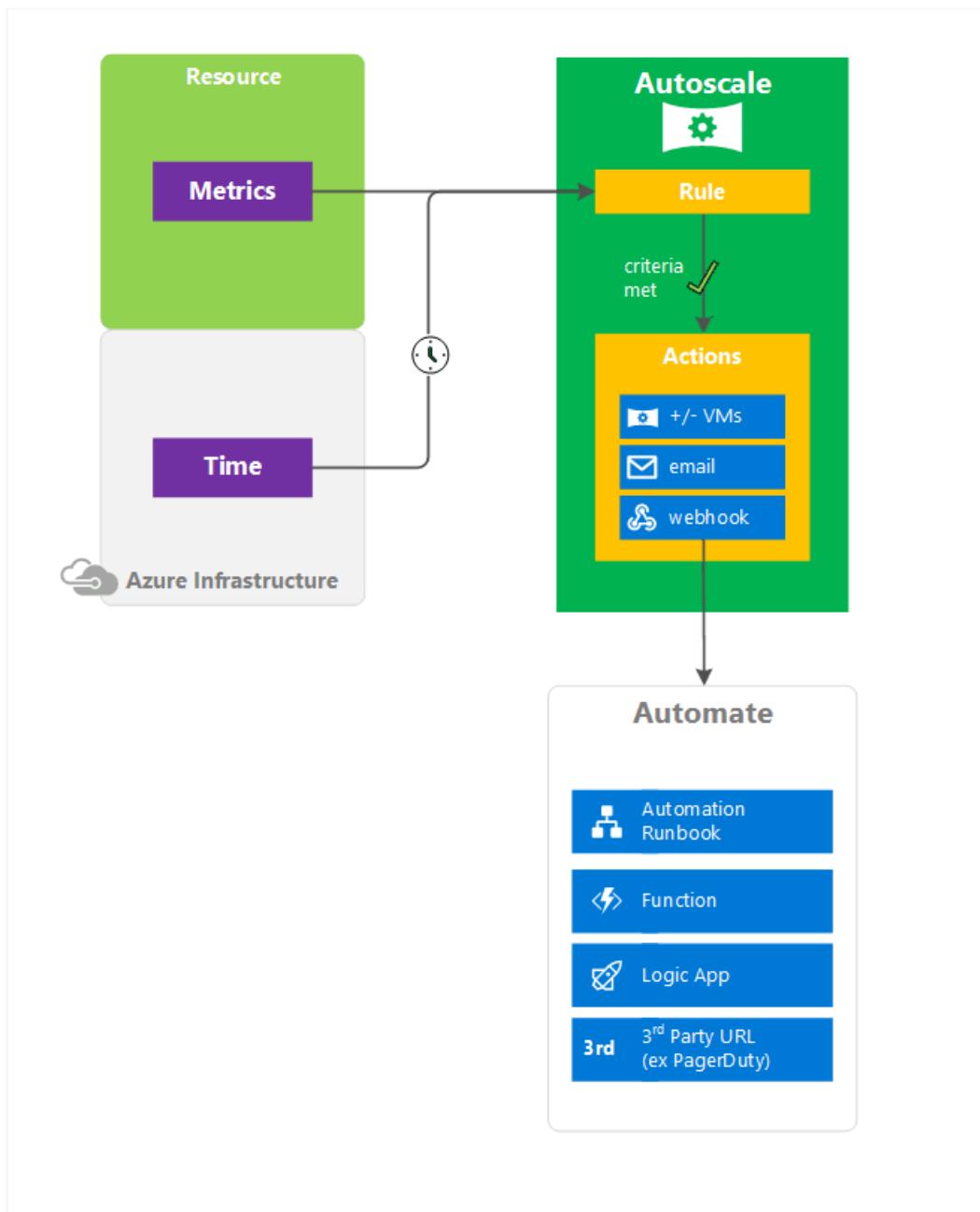
Azure has two autoscale methods. An older version of autoscale applies to Virtual Machines (availability sets). This feature has limited support and we recommend migrating to virtual machine scale sets for faster and more reliable autoscale support. A link on how to use the older technology is included in this article.

What is autoscale?

Autoscale allows you to have the right amount of resources running to handle the load on your application. It allows you to add resources to handle increases in load and also save money by removing resources that are sitting idle. You specify a minimum and maximum number of instances to run and add or remove VMs automatically based on a set of rules. Having a minimum makes sure your application is always running even under no load. Having a maximum limits your total possible hourly cost. You automatically scale between these two extremes using rules you create.



When rule conditions are met, one or more autoscale actions are triggered. You can add and remove VMs, or perform other actions. The following conceptual diagram shows this process.



The following explanation applies to the pieces of the previous diagram.

Resource Metrics

Resources emit metrics, these metrics are later processed by rules. Metrics come via different methods. Virtual machine scale sets use telemetry data from Azure diagnostics agents whereas telemetry for Web apps and Cloud services comes directly from the Azure Infrastructure. Some commonly used statistics include CPU Usage, memory usage, thread counts, queue length, and disk usage. For a list of what telemetry data you can use, see [Autoscale Common Metrics](#).

Custom Metrics

You can also leverage your own custom metrics that your application(s) may be emitting. If you have configured your application(s) to send metrics to Application Insights you can leverage those metrics to make decisions on whether to scale or not.

Time

Schedule-based rules are based on UTC. You must set your time zone properly when setting up your rules.

Rules

The diagram shows only one autoscale rule, but you can have many of them. You can create complex overlapping rules as needed for your situation. Rule types include

- **Metric-based** - For example, do this action when CPU usage is above 50%.
- **Time-based** - For example, trigger a webhook every 8am on Saturday in a given time zone.

Metric-based rules measure application load and add or remove VMs based on that load. Schedule-based rules allow you to scale when you see time patterns in your load and want to scale before a possible load increase or decrease occurs.

Actions and automation

Rules can trigger one or more types of actions.

- **Scale** - Scale VMs in or out
- **Email** - Send email to subscription admins, co-admins, and/or additional email address you specify
- **Automate via webhooks** - Call webhooks, which can trigger multiple complex actions inside or outside Azure. Inside Azure, you can start an Azure Automation runbook, Azure Function, or Azure Logic App. Example third-party URL outside Azure include services like Slack and Twilio.

Autoscale Settings

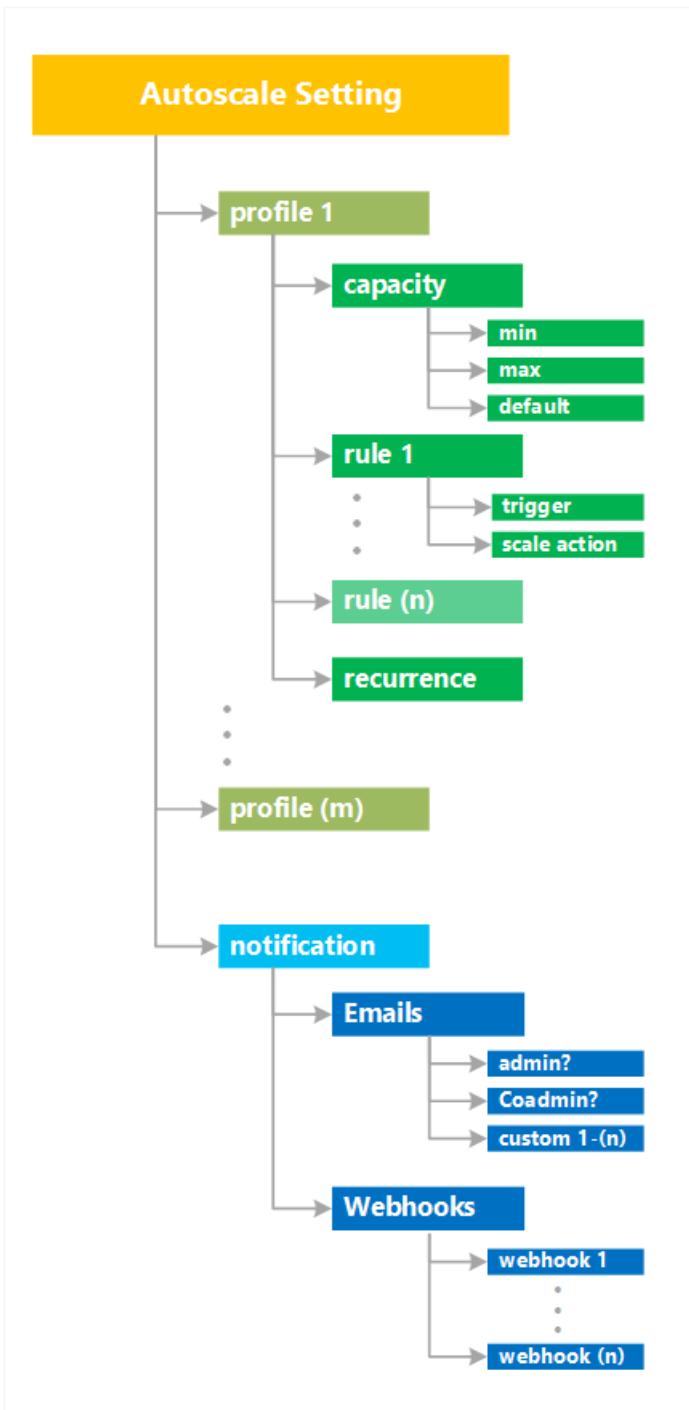
Autoscale use the following terminology and structure.

- An **autoscale setting** is read by the autoscale engine to determine whether to scale up or down. It contains one or more profiles, information about the target resource, and notification settings.

- An **autoscale profile** is a combination of a:
 - **capacity setting**, which indicates the minimum, maximum, and default values for number of instances.
 - **set of rules**, each of which includes a trigger (time or metric) and a scale action (up or down).
 - **recurrence**, which indicates when autoscale should put this profile into effect.

You can have multiple profiles, which allow you to take care of different overlapping requirements. You can have different autoscale profiles for different times of day or days of the week, for example.

- A **notification setting** defines what notifications should occur when an autoscale event occurs based on satisfying the criteria of one of the autoscale setting's profiles. Autoscale can notify one or more email addresses or make calls to one or more webhooks.



The full list of configurable fields and descriptions is available in the [Autoscale REST API](#).

For code examples, see

- [Advanced Autoscale configuration using Resource Manager templates for VM Scale Sets](#)
- [Autoscale REST API](#)

Horizontal vs vertical scaling

Autoscale only scales horizontally, which is an increase ("out") or decrease ("in") in the number of VM instances. Horizontal is more flexible in a cloud situation as it allows you to run potentially thousands of VMs to handle load.

In contrast, vertical scaling is different. It keeps the same number of VMs, but makes the VMs more ("up") or less ("down") powerful. Power is measured in memory, CPU speed, disk space, etc. Vertical scaling has more limitations. It's dependent on the availability of larger hardware, which quickly hits an upper limit and can vary by region. Vertical scaling also usually requires a VM to stop and restart.

For more information, see [Vertically scale Azure virtual machine with Azure Automation](#).

Methods of access

You can set up autoscale via

- [Azure portal](#)
- [PowerShell](#)
- [Cross-platform Command Line Interface \(CLI\)](#)
- [Azure Monitor REST API](#)

Supported services for autoscale

SERVICE	SCHEMA & DOCS
Web Apps	Scaling Web Apps
Cloud Services	Autoscale a Cloud Service
Virtual Machines: Classic	Scaling Classic Virtual Machine Availability Sets
Virtual Machines: Windows Scale Sets	Scaling virtual machine scale sets in Windows
Virtual Machines: Linux Scale Sets	Scaling virtual machine scale sets in Linux
Virtual Machines: Windows Example	Advanced Autoscale configuration using Resource Manager templates for VM Scale Sets

Next steps

To learn more about autoscale, use the Autoscale Walkthroughs listed previously or refer to the following resources:

- [Azure Monitor autoscale common metrics](#)
- [Best practices for Azure Monitor autoscale](#)
- [Use autoscale actions to send email and webhook alert notifications](#)
- [Autoscale REST API](#)
- [Troubleshooting Virtual Machine Scale Sets Autoscale](#)

Overview of the Azure Activity Log

7/25/2017 • 8 min to read • [Edit Online](#)

The **Azure Activity Log** is a log that provides insight into subscription-level events that have occurred in Azure. This includes a range of data, from Azure Resource Manager operational data to updates on Service Health events. The Activity Log was previously known as "Audit Logs" or "Operational Logs," since the Administrative category reports control-plane events for your subscriptions. Using the Activity Log, you can determine the 'what, who, and when' for any write operations (PUT, POST, DELETE) taken on the resources in your subscription. You can also understand the status of the operation and other relevant properties. The Activity Log does not include read (GET) operations or operations for resources that use the Classic/"RDFE" model.

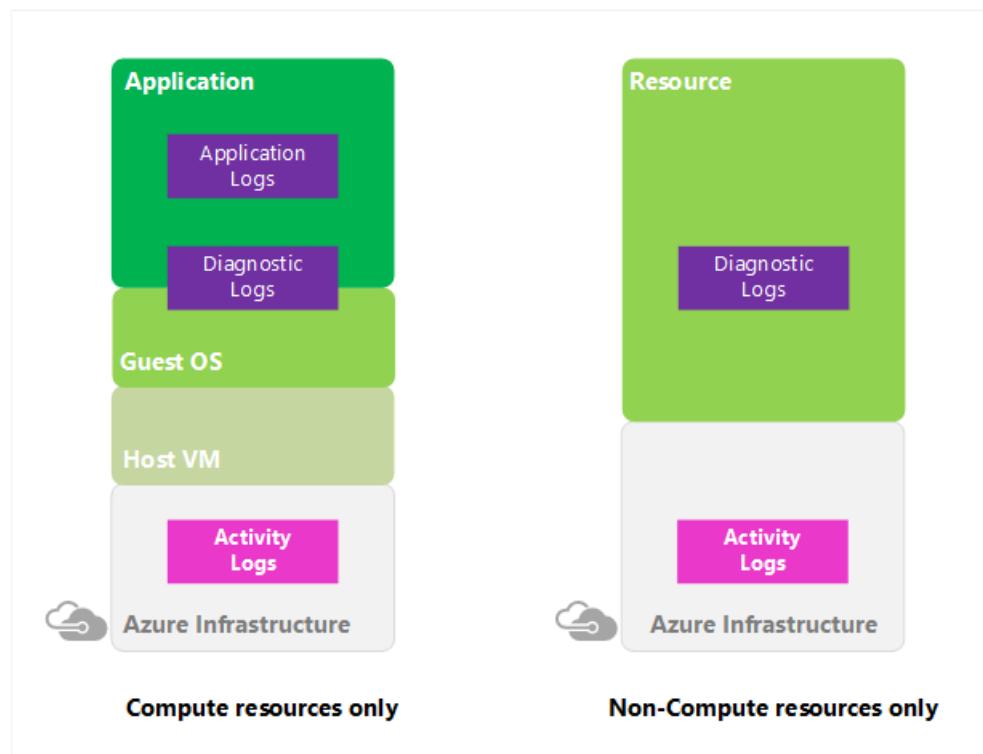


Figure 1: Activity Logs vs other types of logs

The Activity Log differs from [Diagnostic Logs](#). Activity Logs provide data about the operations on a resource from the outside (the "control plane"). Diagnostics Logs are emitted by a resource and provide information about the operation of that resource (the "data plane").

You can retrieve events from your Activity Log using the Azure portal, CLI, PowerShell cmdlets, and Azure Monitor REST API.

WARNING

The Azure Activity Log is primarily for activities that occur in Azure Resource Manager. It does not track resources using the Classic/RDFE model. Some Classic resource types have a proxy resource provider in Azure Resource Manager (for example, Microsoft.ClassicCompute). If you interact with a Classic resource type through Azure Resource Manager using these proxy resource providers, the operations appear in the Activity Log. If you interact with a Classic resource type in the Classic portal or otherwise outside of the Azure Resource Manager proxies, your actions are only recorded in the Operation Log. The Operation Log is accessible only in the Classic portal.

View the following video introducing the Activity Log.

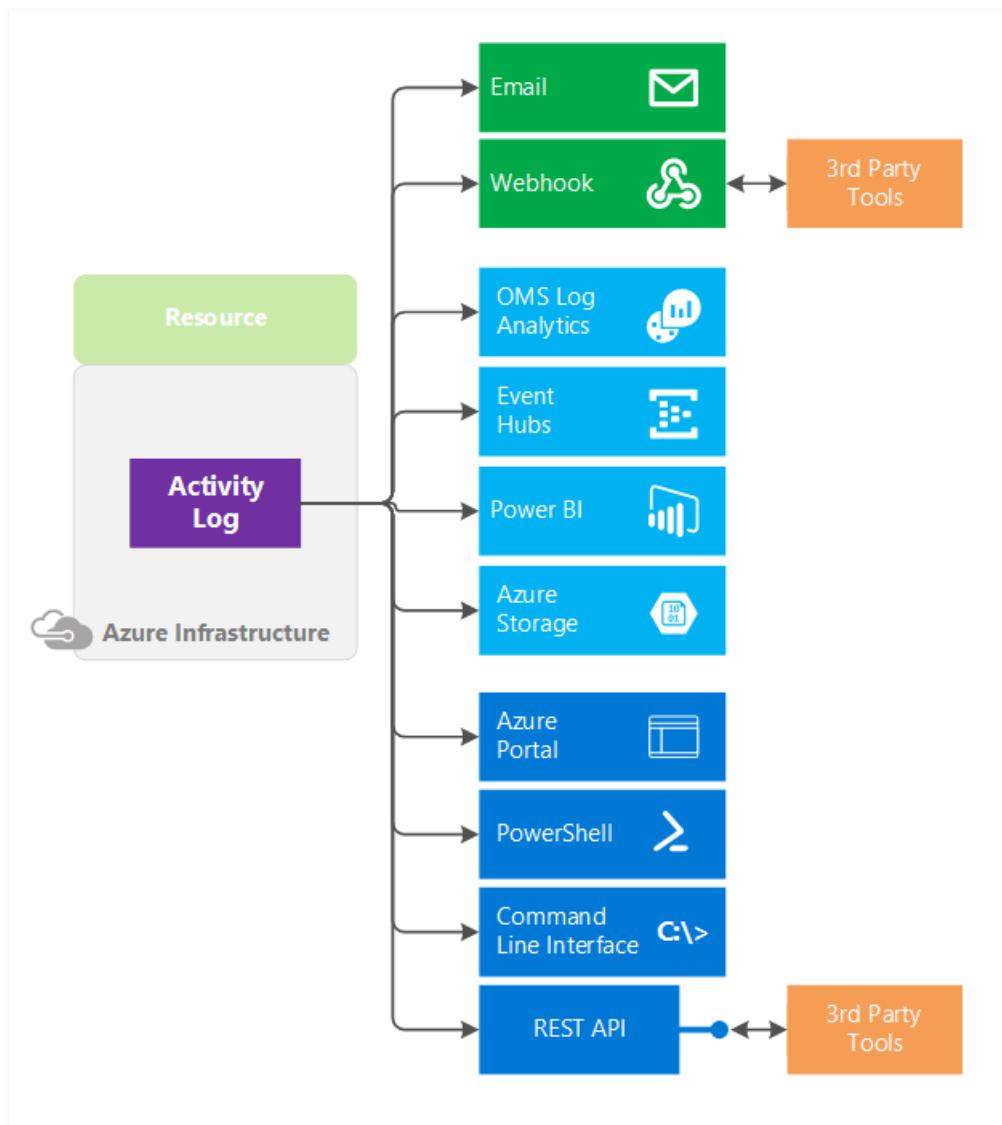
Categories in the Activity Log

The Activity Log contains several categories of data. For full details on the schemata of these categories, [please see this article](#). These include:

- **Administrative** - This category contains the record of all create, update, delete, and action operations performed through Resource Manager. Examples of the types of events you would see in this category include "create virtual machine" and "delete network security group." Every action taken by a user or application using Resource Manager is modeled as an operation on a particular resource type. If the operation type is Write, Delete, or Action, the records of both the start and success or fail of that operation are recorded in the Administrative category. The Administrative category also includes any changes to role-based access control in a subscription.
- **Service Health** - This category contains the record of any service health incidents that have occurred in Azure. An example of the type of event you would see in this category is "SQL Azure in East US is experiencing downtime." Service health events come in five varieties: Action Required, Assisted Recovery, Incident, Maintenance, Information, or Security, and only appear if you have a resource in the subscription that would be impacted by the event.
- **Alert** - This category contains the record of all activations of Azure alerts. An example of the type of event you would see in this category is "CPU % on myVM has been over 80 for the past 5 minutes." A variety of Azure systems have an alerting concept -- you can define a rule of some sort and receive a notification when conditions match that rule. Each time a supported Azure alert type 'activates,' or the conditions are met to generate a notification, a record of the activation is also pushed to this category of the Activity Log.
- **Autoscale** - This category contains the record of any events related to the operation of the autoscale engine based on any autoscale settings you have defined in your subscription. An example of the type of event you would see in this category is "Autoscale scale up action failed." Using autoscale, you can automatically scale out or scale in the number of instances in a supported resource type based on time of day and/or load (metric) data using an autoscale setting. When the conditions are met to scale up or down, the start and succeeded or failed events will be recorded in this category.
- **Recommendation** - This category contains recommendation events from certain resource types, such as web sites and SQL servers. These events offer recommendations for how to better utilize your resources. You will only receive events of this type if you have resources that emit recommendations.
- **Policy, Security, and Resource Health** - These categories do not contain any events; they are reserved for future use.

What you can do with the Activity Log

Here are some of the things you can do with the Activity Log:



- Create an alert that triggers off an Activity Log event.
- Stream it to an **Event Hub** for ingestion by a third-party service or custom analytics solution such as PowerBI.
- Analyze it in PowerBI using the **PowerBI content pack**.
- Save it to a **Storage Account** for archival or manual inspection. You can specify the retention time (in days) using **Log Profiles**.
- Query and view it in the **Azure portal**.
- Query it via PowerShell Cmdlet, CLI, or REST API.

You can use a storage account or event hub namespace that is not in the same subscription as the one emitting logs. The user who configures the setting must have the appropriate RBAC access to both subscriptions.

Export the Activity Log with Log Profiles

A **Log Profile** controls how your Activity Log is exported. Using a Log Profile, you can configure:

- Where the Activity Log should be sent (Storage Account or Event Hubs)
- Which event categories (Write, Delete, Action) should be sent. *The meaning of "category" in Log Profiles and Activity Log events is different. In the Log Profile, "Category" represents the operation type (Write, Delete, Action). In an Activity Log event, the "category" property represents the source or type of event (for example, Administration, ServiceHealth, Alert, and more).*
- Which regions (locations) should be exported
- How long the Activity Log should be retained in a Storage Account.
 - A retention of zero days means logs are kept forever. Otherwise, the value can be any number of days

between 1 and 2147483647.

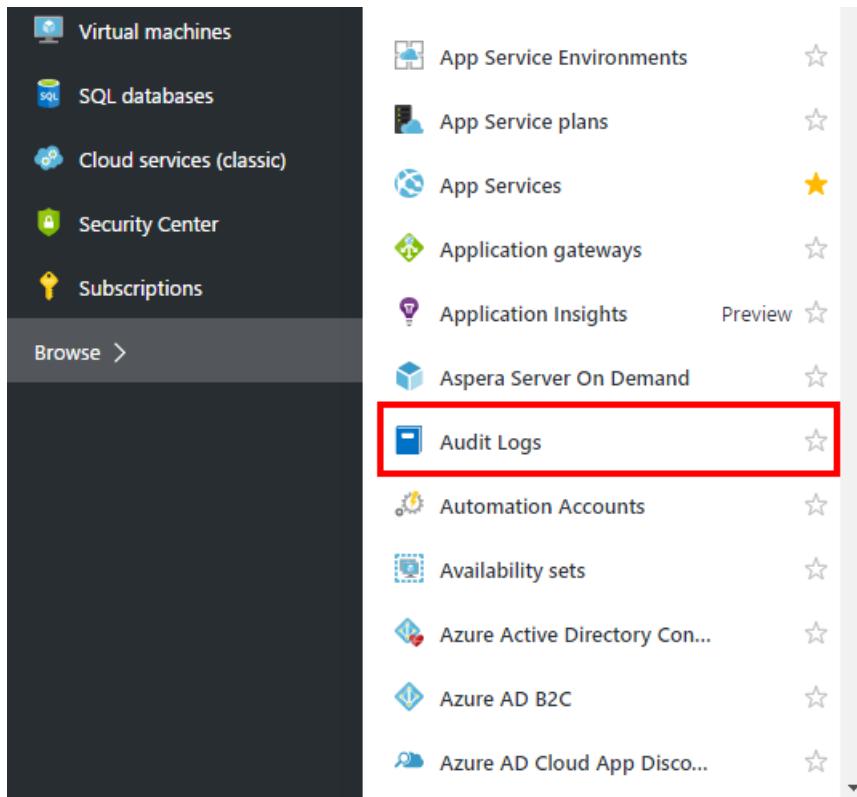
- If retention policies are set but storing logs in a Storage Account is disabled (for example, if only Event Hubs or OMS options are selected), the retention policies have no effect.
- Retention policies are applied per-day, so at the end of a day (UTC), logs from the day that is now beyond the retention policy are deleted. For example, if you had a retention policy of one day, at the beginning of the day today the logs from the day before yesterday would be deleted.

These settings can be configured via the "Export" option in the Activity Log blade in the portal. They can also be configured programmatically [using the Azure Monitor REST API](#), PowerShell cmdlets, or CLI. A subscription can only have one log profile.

Configure log profiles using the Azure portal

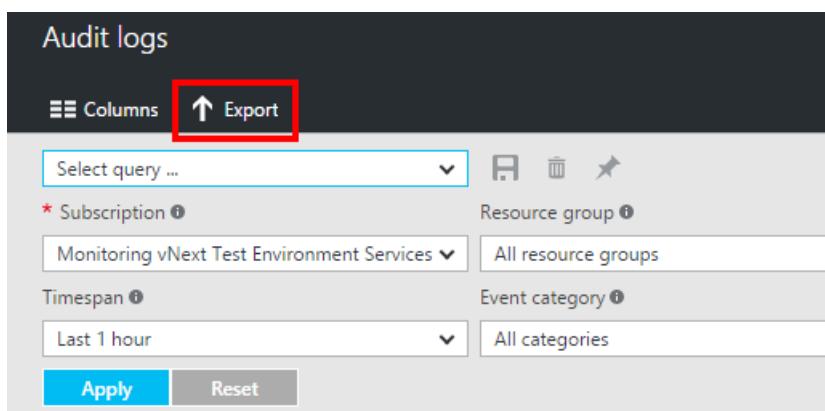
You can stream the Activity Log to an Event Hub or store them in a Storage Account by using the "Export" option in the Azure portal.

1. Navigate to the **Activity Log** blade using the menu on the left side of the portal.



The screenshot shows the left sidebar of the Azure portal. The 'Audit Logs' item is highlighted with a red box. Other items listed include Virtual machines, SQL databases, Cloud services (classic), Security Center, Subscriptions, App Service Environments, App Service plans, App Services, Application gateways, Application Insights (Preview), Aspera Server On Demand, Automation Accounts, Availability sets, Azure Active Directory Con..., Azure AD B2C, and Azure AD Cloud App Disco... . Each item has a star icon next to it.

2. Click the **Export** button at the top of the blade.



The screenshot shows the 'Audit logs' blade. At the top, there is a 'Columns' button and an 'Export' button, which is highlighted with a red box. Below these are dropdown menus for 'Select query ...', 'Subscription' (set to 'Monitoring vNext Test Environment Services'), 'Resource group' (set to 'All resource groups'), 'Timespan' (set to 'Last 1 hour'), 'Event category' (set to 'All categories'), and 'Apply' and 'Reset' buttons.

3. In the blade that appears, you can select:

- regions for which you would like to export events

- the Storage Account to which you would like to save events
- the number of days you want to retain these events in storage. A setting of 0 days retains the logs forever.
- the Service Bus Namespace in which you would like an Event Hub to be created for streaming these events.

Export Audit Logs...

Save **Discard** **Reset**

Archive your Audit logs to a storage account or stream them to an Azure Event Hub. Diagnostic data is billed at normal storage rates.

* Subscription [?](#)
Microsoft Azure Internal Consumption

* Regions [?](#)
0 selected

* STORAGE ACCOUNT [?](#)
Configure required settings

Retention (days) [?](#)
0

AZURE EVENT HUB [?](#)
Optionally configure Event Hub

4. Click **Save** to save these settings. The settings are immediately be applied to your subscription.

Configure log profiles using the Azure PowerShell Cmdlets

Get existing log profile

```
Get-AzureRmLogProfile
```

Add a log profile

```
Add-AzureRmLogProfile -Name my_log_profile -StorageAccountId
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/my_storage -
serviceBusRuleId /subscriptions/s1/resourceGroups/Default-ServiceBus-
EastUS/providers/Microsoft.ServiceBus/namespaces/mytestSB/authorizationrules/RootManageSharedAccessKey -
Locations global,westus,eastus -RetentionInDays 90 -Categories Write,Delete,Action
```

PROPERTY	REQUIRED	DESCRIPTION
Name	Yes	Name of your log profile.
StorageAccountId	No	Resource ID of the Storage Account to which the Activity Log should be saved.

PROPERTY	REQUIRED	DESCRIPTION
serviceBusRuleId	No	<p>Service Bus Rule ID for the Service Bus namespace you would like to have event hubs created in. Is a string with this format:</p> <div style="border: 1px solid black; padding: 5px; display: inline-block;"> {service bus resource ID}/authorizationrules/{key name} </div>
Locations	Yes	Comma-separated list of regions for which you would like to collect Activity Log events.
RetentionInDays	Yes	Number of days for which events should be retained, between 1 and 2147483647. A value of zero stores the logs indefinitely (forever).
Categories	No	Comma-separated list of event categories that should be collected. Possible values are Write, Delete, and Action.

Remove a log profile

```
Remove-AzureRmLogProfile -name my_log_profile
```

Configure log profiles Using the Azure Cross-Platform CLI

Get existing log profile

```
azure insights logprofile list
```

```
azure insights logprofile get --name my_log_profile
```

The `name` property should be the name of your log profile.

Add a log profile

```
azure insights logprofile add --name my_log_profile --storageId /subscriptions/s1/resourceGroups/insights-integration/providers/Microsoft.Storage/storageAccounts/my_storage --serviceBusRuleId /subscriptions/s1/resourceGroups/Default-ServiceBus-EastUS/providers/Microsoft.ServiceBus/namespaces/mytestSB/authorizationrules/RootManageSharedAccessKey --locations global,westus,eastus,northeurope --retentionInDays 90 --categories Write,Delete,Action
```

PROPERTY	REQUIRED	DESCRIPTION
name	Yes	Name of your log profile.
storageId	No	Resource ID of the Storage Account to which the Activity Log should be saved.

PROPERTY	REQUIRED	DESCRIPTION
serviceBusRuleId	No	Service Bus Rule ID for the Service Bus namespace you would like to have event hubs created in. Is a string with this format: <div style="border: 1px solid black; padding: 5px; display: inline-block;">{service bus resource ID}/authorizationrules/{key name}</div>
locations	Yes	Comma-separated list of regions for which you would like to collect Activity Log events.
retentionInDays	Yes	Number of days for which events should be retained, between 1 and 2147483647. A value of zero stores the logs indefinitely (forever).
categories	No	Comma-separated list of event categories that should be collected. Possible values are Write, Delete, and Action.

Remove a log profile

```
azure insights logprofile delete --name my_log_profile
```

Event schema per category

Please see this article to understand the Activity Log event schema per category.

Next Steps

- [Learn more about the Activity Log \(formerly Audit Logs\)](#)
- [Stream the Azure Activity Log to Event Hubs](#)

Create and Manage Action Groups in Azure Portal

7/18/2017 • 2 min to read • [Edit Online](#)

Overview

This article shows you how to create and manage action groups in the Azure portal.

Action groups enable you to configure a list of actions. These groups can then be leveraged when defining activity log alerts; ensuring that a particular action group is invoked when the activity log alert is triggered.

An action group can have up to 10 of each action type. An action is defined by the combination of:

Name: A unique identifier within the action group.

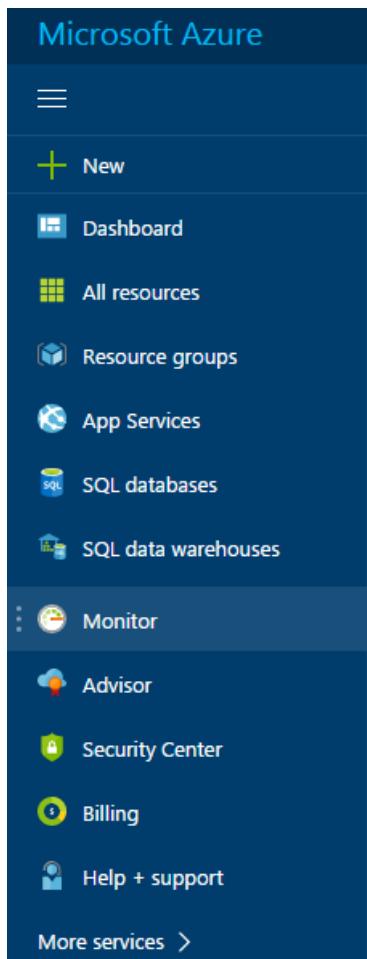
Action Type: This defines the action that will be performed. Options are send SMS, send Email, or call a Webhook.

Details: Based on the action type, the corresponding phone number, email address or webhook URI needs to be provided.

For information on using Azure Resource Manager templates to configure action groups: [Action Group Resource Manager templates](#)

Creating an action group using the Azure portal

1. In the [portal](#), navigate to the **Monitor** service



2. Click the **Monitor** option to open up the Monitor blade. This blade brings together all your monitoring settings and data into one consolidated view. It first opens to the **Activity log** section.

3. Now click on **Action groups** section

The screenshot shows the 'Monitor - Action groups' interface. On the left, there's a sidebar with sections 'EXPLORE' and 'MANAGE'. Under 'EXPLORE', options include 'Activity log', 'Metrics', 'Diagnostics logs', 'Log search', and 'Service notifications'. Under 'MANAGE', options include 'Alerts' and 'Action groups'. The 'Action groups' option is highlighted with a blue background.

4. Click on the **Add** action group command and fill in the fields

The screenshot shows the 'Monitor - Action groups' interface. At the top right, there are buttons for 'Columns' and 'Add action group'. The 'Add action group' button is highlighted with a red box. Below it, there's a note: '* Subscription ⓘ'.

5. Provide a **Name** and **Short Name** for the action group; The Short Name will be referenced in notifications sent to this group

Add action group

* Action group name ⓘ	Sample Action Group	✓
* Short name ⓘ	SampleAG	✓
* Subscription ⓘ	WARP Production	▼
* Resource group ⓘ	Default-ActivityLogAlerts (to be created)	▼

Receivers

NAME	ACTION TYPE	DETAILS
SMS-Receiver	SMS	555-555-5555
Email-Receiver	Email	sample@contoso.com
Webhook-Receiver	Webhook	https://sample.webhook.uri
SMS-Receiver-2	SMS	555-555-5505
Email-Receiver-2	Email	sample-2@contoso.com

The name of the receiver: SMS Details: Depending on the action type...

i Note that only the country code '1' is currently supported for SMS.

[Privacy Statement](#)

[Pricing](#)

OK

6. The **Subscription** is the one the Action group will be saved in. It will be auto filled to the subscription you are currently operating under.
7. Choose the **Resource Group** this alert will be associated with in the **Subscription**.
8. Then, define a list of actions through a combination of:
 - a. **Name:** A unique identifier within the action group.
 - b. **Action Type:** This defines the action that will be performed. Options are send SMS, send Email, or call a Webhook.
 - c. **Details:** Based on the action type, the corresponding phone number, email address or webhook URI needs to be provided.
9. Select **OK** when done to create the action group.

Managing your action groups

Once you have created an action group, it will be visible in the Action groups section of the Monitor service. Select the action group you wish to manage, you will be able to:

- Add, edit, or remove actions.
- Delete the action group.

Next Steps:

Learn more on [SMS alert behavior](#)

Get an [understanding of the activity log alert webhook schema](#)

Learn more about [rate limiting](#) on alerts

Get an overview of activity log alerts and learn how to get alerted

How to configure alerts whenever a service health notification is posted

Collect and consume diagnostic data from your Azure resources

7/18/2017 • 7 min to read • [Edit Online](#)

What are Azure Diagnostic Logs

Azure Diagnostic Logs are logs emitted by a resource that provide rich, frequent data about the operation of that resource. The content of these logs varies by resource type. For example, Windows event system logs are one category of Diagnostic Log for VMs and blob, table, and queue logs are categories of Diagnostic Logs for storage accounts.

Diagnostics Logs differ from the [Activity Log \(formerly known as Audit Log or Operational Log\)](#). The Activity log provides insight into the operations that were performed on resources in your subscription. Diagnostics logs provide insight into operations that your resource performed itself.

Not all resources support the new type of Diagnostic Logs described here. This article contains a section listing which resource types support the new Diagnostic Logs.

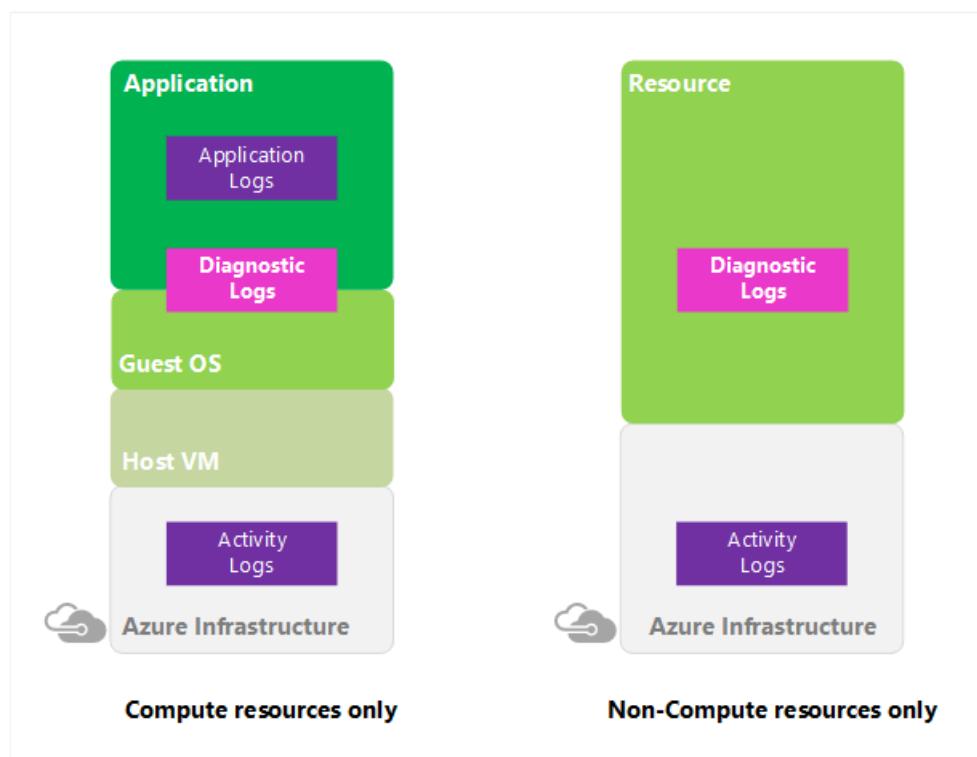
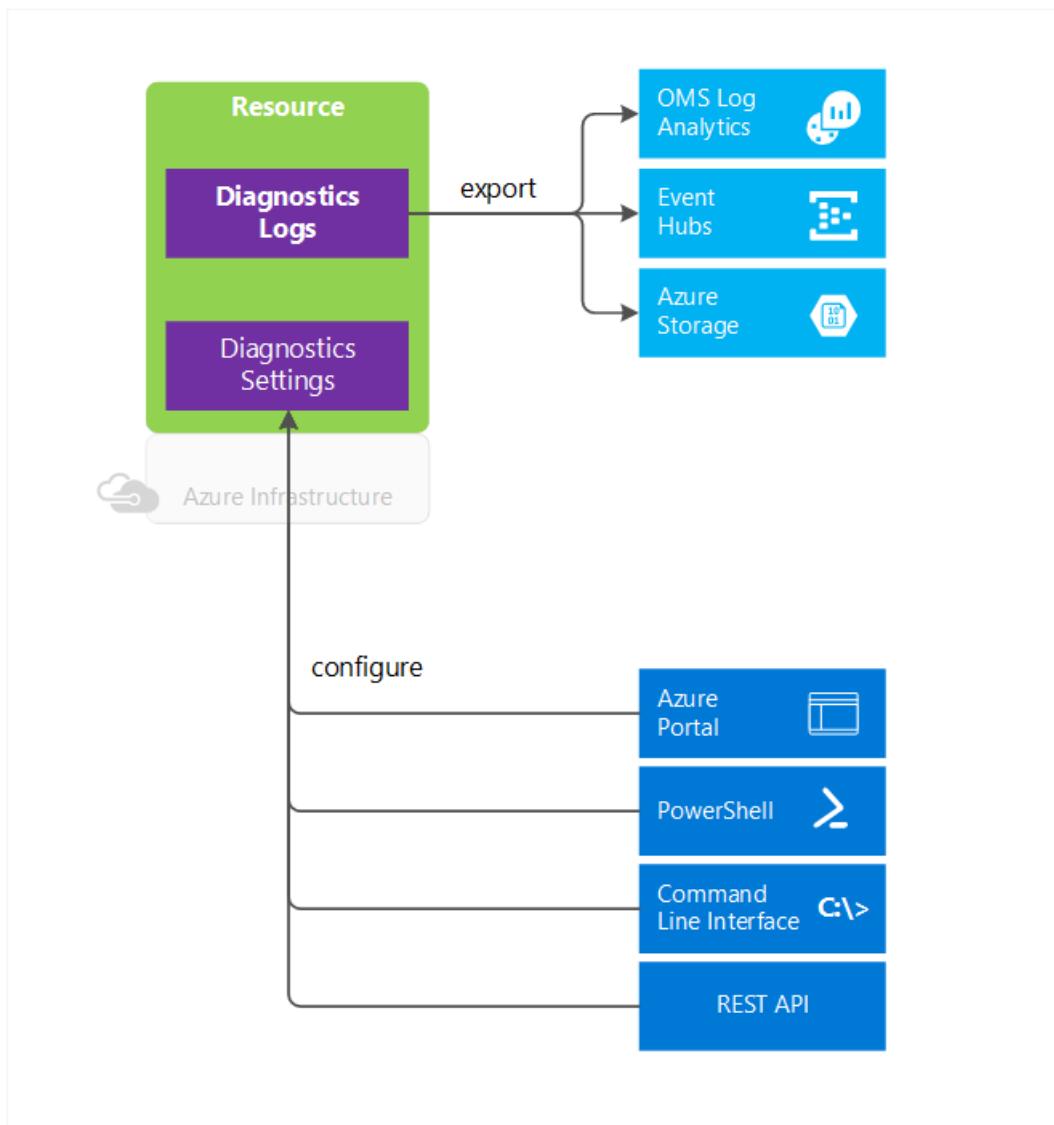


Figure 1: Diagnostics Logs vs other types of logs

What you can do with Diagnostic Logs

Here are some of the things you can do with Diagnostic Logs:



- Save them to a **Storage Account** for auditing or manual inspection. You can specify the retention time (in days) using the **Diagnostic Settings**.
- Stream them to **Event Hubs** for ingestion by a third-party service or custom analytics solution such as PowerBI.
- Analyze them with **OMS Log Analytics**

You can use a storage account or Event Hubs namespace that is not in the same subscription as the one emitting logs. The user who configures the setting must have the appropriate RBAC access to both subscriptions.

Diagnostic Settings

Diagnostic Logs for non-Compute resources are configured using Diagnostic Settings. **Diagnostic Settings** for a resource control:

- Where Diagnostic Logs are sent (Storage Account, Event Hubs, and/or OMS Log Analytics).
- Which Log Categories are sent.
- How long each log category should be retained in a Storage Account
 - A retention of zero days means logs are kept forever. Otherwise, the value can be any number of days between 1 and 2147483647.
 - If retention policies are set but storing logs in a Storage Account is disabled (for example, if only Event Hubs or OMS options are selected), the retention policies have no effect.
 - Retention policies are applied per-day, so at the end of a day (UTC), logs from the day that is now beyond the retention policy are deleted. For example, if you had a retention policy of one day, at the

beginning of the day today the logs from the day before yesterday would be deleted.

These settings are easily configured via the Diagnostics blade for a resource in the Azure portal, via Azure PowerShell and CLI commands, or via the [Azure Monitor REST API](#).

WARNING

Diagnostic logs and metrics for Compute resources (for example, VMs or Service Fabric) use [a separate mechanism for configuration and selection of outputs](#).

How to enable collection of Diagnostic Logs

Collection of Diagnostic Logs can be enabled as part of creating a resource or after a resource is created via the resource's blade in the Portal. You can also enable Diagnostic Logs at any point using Azure PowerShell or CLI commands, or using the Azure Monitor REST API.

TIP

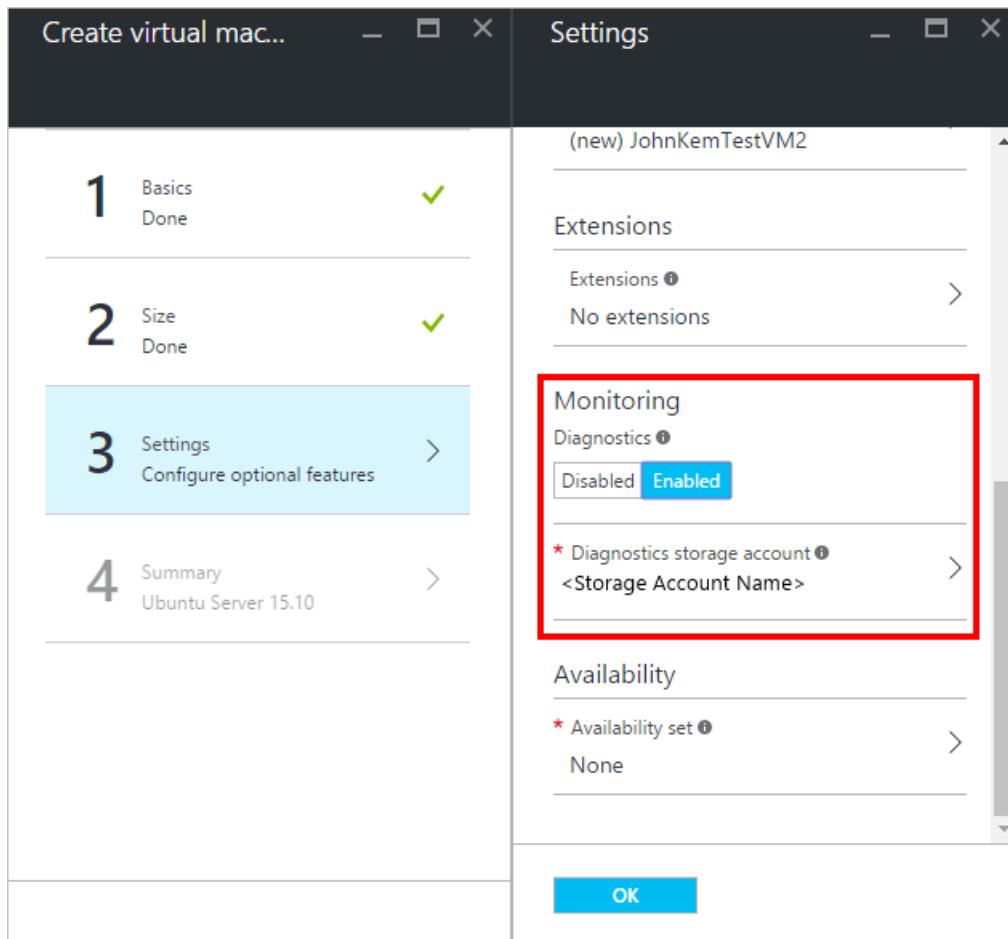
These instructions may not apply directly to every resource. See the schema links at the bottom of this page to understand special steps that may apply to certain resource types.

This article shows how you can use a resource template to enable Diagnostic Settings when creating a resource

Enable Diagnostic Logs in the portal

You can enable Diagnostic Logs in the Azure portal when you create compute resource types by enabling the Windows or Linux Azure Diagnostics extension:

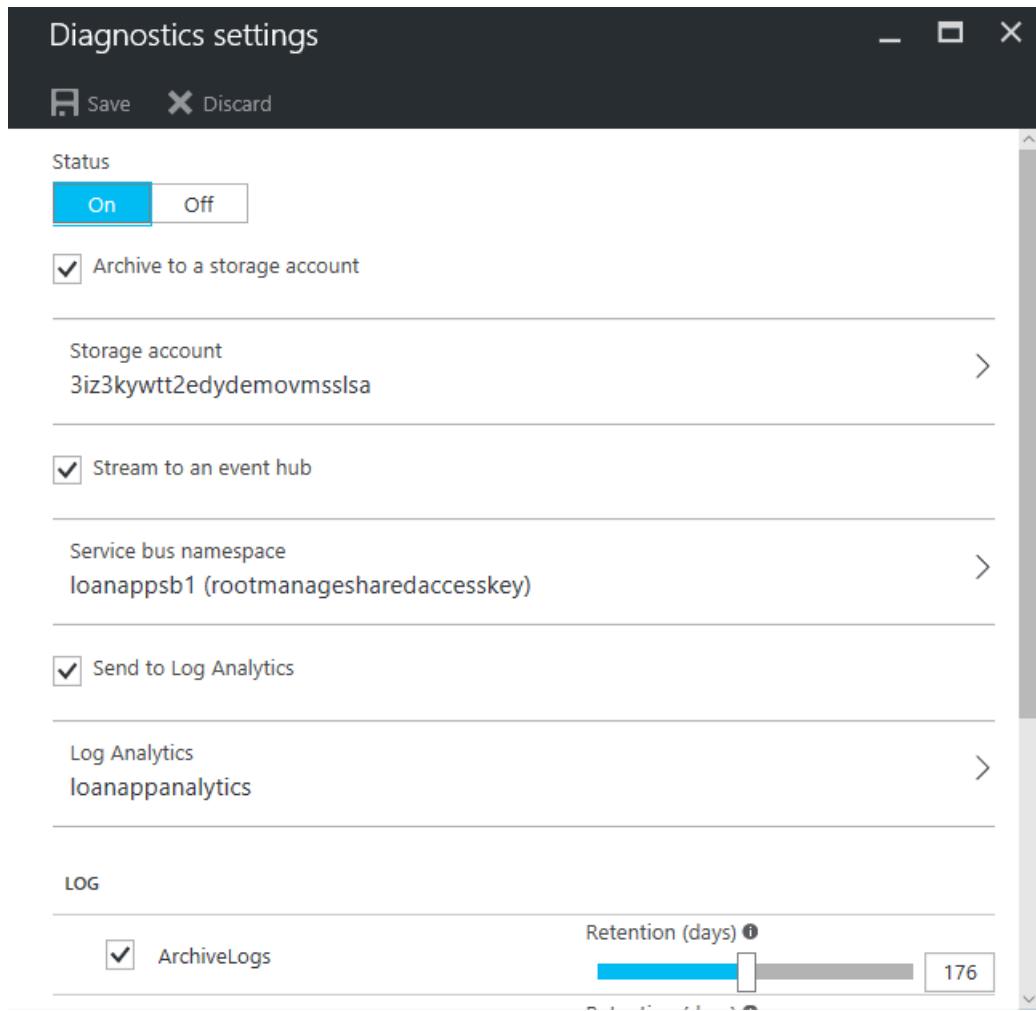
1. Go to **New** and choose the resource you are interested in.
2. After configuring the basic settings and selecting a size, in the **Settings** blade, under **Monitoring**, select **Enabled** and choose a storage account where you would like to store the Diagnostic Logs. You are charged normal data rates for storage and transactions when you send diagnostics to a storage account.



3. Click **OK** and create the resource.

For non-compute resources, you can enable Diagnostic Logs in the Azure portal after a resource has been created by doing the following:

1. Go to the blade for the resource and open the **Diagnostics** blade.
2. Click **On** and pick a Storage Account and/or event hub.



3. Under **Logs**, select which **Log Categories** you would like to collect or stream.

4. Click **Save**.

Enable Diagnostic Logs via PowerShell

To enable Diagnostic Logs via the Azure PowerShell Cmdlets, use the following commands.

To enable storage of Diagnostic Logs in a Storage Account, use this command:

```
Set-AzureRmDiagnosticSetting -ResourceId [your resource id] -StorageAccountId [your storage account id] -Enabled $true
```

The Storage Account ID is the resource id for the storage account to which you want to send the logs.

To enable streaming of Diagnostic Logs to an event hub, use this command:

```
Set-AzureRmDiagnosticSetting -ResourceId [your resource id] -ServiceBusRuleId [your Service Bus rule id] -Enabled $true
```

The Service Bus Rule ID is a string with this format: `{Service Bus resource ID}/authorizationrules/{key name}`.

To enable sending of Diagnostic Logs to a Log Analytics workspace, use this command:

```
Set-AzureRmDiagnosticSetting -ResourceId [your resource id] -WorkspaceId [resource id of the log analytics workspace] -Enabled $true
```

You can obtain the resource id of your Log Analytics workspace using the following command:

```
(Get-AzureRmOperationalInsightsWorkspace).ResourceId
```

You can combine these parameters to enable multiple output options.

Enable Diagnostic Logs via CLI

To enable Diagnostic Logs via the Azure CLI, use the following commands:

To enable storage of Diagnostic Logs in a Storage Account, use this command:

```
azure insights diagnostic set --resourceId <resourceId> --storageId <storageAccountId> --enabled true
```

The Storage Account ID is the resource id for the storage account to which you want to send the logs.

To enable streaming of Diagnostic Logs to an event hub, use this command:

```
azure insights diagnostic set --resourceId <resourceId> --serviceBusRuleId <serviceBusRuleId> --enabled true
```

The Service Bus Rule ID is a string with this format: `{Service Bus resource ID}/authorizationrules/{key name}`.

To enable sending of Diagnostic Logs to a Log Analytics workspace, use this command:

```
azure insights diagnostic set --resourceId <resourceId> --workspaceId <resource id of the log analytics workspace> --enabled true
```

You can combine these parameters to enable multiple output options.

Enable Diagnostic Logs via REST API

To change Diagnostic Settings using the Azure Monitor REST API, see [this document](#).

Manage Diagnostic Settings in the portal

Ensure that all of your resources are set up with diagnostic settings. Navigate to the **Monitoring** blade in the portal and open the **Diagnostic Logs** blade.

You may have to click "More services" to find the Monitoring blade.

In this blade, you can view and filter all resources that support Diagnostic Logs to see if they have diagnostics enabled. You can also check which storage account, event hub, and/or Log Analytics workspace those logs are flowing to.

Name	Resource Type	Resource...	Diagnostic...	Storage Account	Service Bus Name...	Log Analytics
johnkembatch	Batch Account	JohnKe...	Disabled			
JohnKemTestLA	Logic App	JohnKe...	Enabled	storageAccount1		
johnkemtestlan...	Logic App	JohnKe...	Disabled			
webhookTest	Logic App	JohnKe...	Disabled			
alertTesdlfj867...	Network sec...	JohnKe...	Disabled			
johnkemtestVM	Network sec...	JohnKe...	Enabled	storageAccount1	/subscriptions/df...	
JohnKemSA	Stream Anal...	JohnKe...	Enabled	storageAccount1		

Clicking on a resource shows all logs that have been stored in the storage account and give you the option to turn off or modify the diagnostic settings. Click the download icon to download logs for a particular time period.

Turn off diagnostics

* Log categories	Timespan	
2 selected	Last 24 hours	
<hr/>		
LOG CATEGORIES	LAST UPDATED	SIZE
Network Security Group Event	23 h ago	100.130 KB
Network Security Group Event	22 h ago	100.130 KB
Network Security Group Event	21 h ago	100.130 KB
Network Security Group Event	20 h ago	100.130 KB
Network Security Group Event	19 h ago	100.130 KB
Network Security Group Event	18 h ago	100.130 KB
Network Security Group Event	17 h ago	100.130 KB
Network Security Group Event	16 h ago	100.130 KB

NOTE

Diagnostic logs only appear in this view and be available for download if you have configured diagnostic settings to save them to a storage account.

Clicking on the link for **Diagnostic Settings** shows the Diagnostic Settings blade, where you can enable, disable, or modify your diagnostic settings for the selected resource.

Supported services and schema for Diagnostic Logs

The schema for Diagnostic Logs varies depending on the resource and log category.

SERVICE	SCHEMA & DOCS
API Management	Schema not available.
Application Gateways	Diagnostics Logging for Application Gateway
Azure Automation	Log analytics for Azure Automation
Azure Batch	Azure Batch diagnostic logging
Customer Insights	Schema not available.
Content Delivery Network	Schema not available.
CosmosDB	Schema not available.
Data Lake Analytics	Accessing diagnostic logs for Azure Data Lake Analytics
Data Lake Store	Accessing diagnostic logs for Azure Data Lake Store

SERVICE	SCHEMA & DOCS
Event Hubs	Azure Event Hubs diagnostic logs
Key Vault	Azure Key Vault Logging
Load Balancer	Log analytics for Azure Load Balancer
Logic Apps	Logic Apps B2B custom tracking schema
Network Security Groups	Log analytics for network security groups (NSGs)
Recovery Services	Schema not available.
Search	Enabling and using Search Traffic Analytics
Server Management	Schema not available.
Service Bus	Azure Service Bus diagnostic logs
Stream Analytics	Job diagnostic logs

Supported log categories per resource type

RESOURCE TYPE	CATEGORY	CATEGORY DISPLAY NAME
Microsoft.ApiManagement/service	GatewayLogs	Logs related to ApiManagement Gateway
Microsoft.Automation/automationAccounts	JobLogs	Job Logs
Microsoft.Automation/automationAccounts	JobStreams	Job Streams
Microsoft.Automation/automationAccounts	DscNodeStatus	Dsc Node Status
Microsoft.Batch/batchAccounts	ServiceLog	Service Logs
Microsoft.Cdn/profiles/endpoints	CoreAnalytics	Gets the metrics of the endpoint, e.g., bandwidth, egress, etc.
Microsoft.CustomerInsights/hubs	AuditEvents	AuditEvents
Microsoft.DataLakeAnalytics/accounts	Audit	Audit Logs
Microsoft.DataLakeAnalytics/accounts	Requests	Request Logs
Microsoft.DataLakeStore/accounts	Audit	Audit Logs
Microsoft.DataLakeStore/accounts	Requests	Request Logs

RESOURCE TYPE	CATEGORY	CATEGORY DISPLAY NAME
Microsoft.DocumentDB/databaseAccounts	DataPlaneRequests	DataPlaneRequests
Microsoft.EventHub/namespaces	ArchiveLogs	Archive Logs
Microsoft.EventHub/namespaces	OperationalLogs	Operational Logs
Microsoft.EventHub/namespaces	AutoScaleLogs	Auto Scale Logs
Microsoft.KeyVault/vaults	AuditEvent	Audit Logs
Microsoft.Logic/workflows	WorkflowRuntime	Workflow runtime diagnostic events
Microsoft.Logic/integrationAccounts	IntegrationAccountTrackingEvents	Integration Account track events
Microsoft.Network/networksecuritygroups	NetworkSecurityGroupEvent	Network Security Group Event
Microsoft.Network/networksecuritygroups	NetworkSecurityGroupRuleCounter	Network Security Group Rule Counter
Microsoft.Network/loadBalancers	LoadBalancerAlertEvent	Load Balancer Alert Events
Microsoft.Network/loadBalancers	LoadBalancerProbeHealthStatus	Load Balancer Probe Health Status
Microsoft.Network/applicationGateways	ApplicationGatewayAccessLog	Application Gateway Access Log
Microsoft.Network/applicationGateways	ApplicationGatewayPerformanceLog	Application Gateway Performance Log
Microsoft.Network/applicationGateways	ApplicationGatewayFirewallLog	Application Gateway Firewall Log
Microsoft.RecoveryServices/Vaults	AzureBackupReport	Azure Backup Reporting Data
Microsoft.RecoveryServices/Vaults	AzureSiteRecoveryJobs	Azure Site Recovery Jobs
Microsoft.RecoveryServices/Vaults	AzureSiteRecoveryEvents	Azure Site Recovery Events
Microsoft.RecoveryServices/Vaults	AzureSiteRecoveryReplicatedItems	Azure Site Recovery Replicated Items
Microsoft.Search/searchServices	OperationLogs	Operation Logs
Microsoft.ServiceBus/namespaces	OperationalLogs	Operational Logs
Microsoft.StreamAnalytics/streamingjobs	Execution	Execution
Microsoft.StreamAnalytics/streamingjobs	Authoring	Authoring

Next Steps

- Stream Diagnostic Logs to [Event Hubs](#)
- Change Diagnostic Settings using the Azure Monitor REST API
- Analyze logs from Azure storage with Log Analytics

Azure Monitor partner integrations

7/18/2017 • 4 min to read • [Edit Online](#)

PARTNERS		
 ALERT LOGIC® Security. Compliance. Cloud. AlertLogic	APPDYNAMICS AppDynamics	 Atlassian Atlassian
 CloudHealth	 CloudMonix	 Cloudyn
 DATADOG Datadog	 dynatrace Dynatrace	 New Relic® NewRelic
 OpsGenie OpsGenie	 pagerduty PagerDuty	 ScienceLogic ScienceLogic
 splunk® Splunk	 sumologic Sumo Logic	

AlertLogic Log Manager

Alert Logic Log Manager collects VM, Application, and Azure platform logs for security analysis and retention. This includes Azure Audit Logs via the Azure Monitor API. This information is used to detect malfeasance and meet compliance requirements.

[Go to the documentation.](#)

AppDynamics

AppDynamics Application Performance Management (APM) enables application owners to rapidly troubleshoot performance bottlenecks and optimize the performance of their applications running in Azure environment. AppDynamics APM is seamlessly integrated with Azure Marketplace and available for monitor Azure Cloud Services (PaaS) (Including web & worker roles), Virtual Machines (IaaS), Remote Service Detection (Microsoft Azure Service Bus), Microsoft Azure Queue Microsoft Azure Remote Services (Azure Blob), Azure Queue (Microsoft Service Bus), Data Storage, Microsoft Azure Blob Storage.

[Go to the documentation.](#)

Atlassian JIRA

You can create JIRA tickets on Azure Monitor alerts.

[Go to the documentation.](#)

CloudHealth

Unite and automate your cloud with a platform built to save serious time and money. With unparalleled visibility, intuitive optimization and rock-solid governance practices, CloudHealth is redefining cloud management. The CloudHealth platform enables enterprises and MSPs to maximize return on cloud investments and make confident decisions around cost, usage, performance and security.

[Learn More.](#)

CloudMonix

CloudMonix offers monitoring, automation and self-healing services for Microsoft Azure platform.

[Go to the documentation.](#)

Cloudyn

Cloudyn manages and optimizes multi-platform, hybrid cloud deployments to help enterprises fully realize their cloud potential. The SaaS solution delivers visibility into usage, performance and cost, coupled with insights and actionable recommendations for smart optimization and cloud governance. Cloudyn enables accountability through accurate chargeback and hierarchical cost allocation management. Cloudyn is integrated with Azure Monitoring in order to provide insights and actionable recommendations in order to optimize your Azure deployment.

[Go to the documentation.](#)

Datadog

Datadog is the world's leading monitoring service for cloud-scale applications, bringing together data from servers, databases, tools and services to present a unified view of your entire stack. These capabilities are provided on a SaaS-based data analytics platform that enables Dev and Ops teams to work collaboratively to avoid downtime, resolve performance problems, and ensure that development and deployment cycles finish on time. By integrating Datadog and Azure, you can collect and view metrics from across your infrastructure, correlate VM metrics with application-level metrics, and slice and dice your metrics using any combination of properties and custom tags.

[Go to the documentation.](#)

Dynatrace

The Dynatrace OneAgent integrates with Azure VMs and App Services via the according Azure extension mechanisms. This way Dynatrace OneAgent can gather performance metrics about hosts, network and services. Besides just displaying metrics Dynatrace visualizes environments end-to-end, showing transactions from the client side to the database layer. AI-based correlation of problems and fully integrated root-cause-analysis, including method level insights into code and database, make troubleshooting and performance optimizations much easier.

[Go to the documentation.](#)

NewRelic

[Learn more.](#)

OpsGenie

OpsGenie acts as a dispatcher for the alerts generated by Azure. OpsGenie determines the right people to notify based on on-call schedules and escalations, by notifying them using email, text messages (SMS), phone calls, push notifications. Simply, Azure generates alerts for detected problems, and OpsGenie ensures the right people are working on them.

[Go to the documentation.](#)

PagerDuty

PagerDuty, the leading incident management solution, has provided first-class support for Azure Alerts on metrics. Today, PagerDuty now supports notifications on Azure Monitor Alerts, Autoscale Notifications, and Audit Log Events, in addition to notifications on platform-level metrics for Azure services. These enhancements give users increased visibility into the core Azure Platform while enabling them to take full advantage of PagerDuty's incident management capabilities for real-time response. Our expanded Azure integration is made possible via webhooks, allowing for quick and easy set-up and customization.

[Go to the documentation.](#)

ScienceLogic

ScienceLogic delivers the next generation IT service assurance platform for managing any technology, anywhere. In one platform, ScienceLogic delivers the scale, security, automation, and resiliency necessary to simplify the ever-expanding task of managing IT resources, services, and applications that are in constant motion. The ScienceLogic platform uses Azure APIs to interface with Microsoft Azure. ScienceLogic gives you real-time visibility into your Azure services and resources so you know when something's not working and can fix it faster. You can also manage Azure alongside your other clouds and data center systems and services.

[Learn more.](#)

Azure Monitor Add-on for Splunk

The Azure Monitor Add-on for Splunk is [available in the Splunkbase here](#).

[Go to the documentation.](#)

Sumo Logic

Sumo Logic is a secure, cloud-native, machine data analytics service, delivering real-time, continuous intelligence from structured, semi-structured and unstructured data across the entire application lifecycle and stack. More than 1,000 customers around the globe rely on Sumo Logic for the analytics and insights to build, run and secure their modern applications and cloud infrastructures. With Sumo Logic, customers gain a multi-tenant, service-model

advantage to accelerate their shift to continuous innovation, increasing competitive advantage, business value and growth.

[Learn more.](#)

Next Steps

- [Learn more about Azure Monitor](#)
- [Access metrics using the REST API](#)
- [Stream the Activity Log to a 3rd party service](#)
- [Stream Diagnostic Logs to a 3rd party service](#)

What is Azure Diagnostics

6/27/2017 • 2 min to read • [Edit Online](#)

Azure Diagnostics is the capability within Azure that enables the collection of diagnostic data on a deployed application. You can use the diagnostics extension from a number of different sources. Currently supported are Azure Cloud Service Web and Worker Roles, Azure Virtual Machines running Microsoft Windows and Service Fabric. Other Azure services have their own separate diagnostics.

Data you can collect

Azure Diagnostics can collect the following types of data:

DATA SOURCE	DESCRIPTION
Performance counters	Operating System and custom performance counters
Application Logs	Trace messages written by your application
Windows Event logs	Information sent to the Windows event logging system
.NET Event Source	Code writing events using the .NET EventSource class
IIS Logs	Information about IIS web sites
Manifest based ETW	Event Tracing for Windows events generated by any process
Crash dumps	Information about the state of the process in the event of an application crash
Custom error logs	Logs created by your application or service
Azure Diagnostic infrastructure logs	Information about Diagnostics itself

The Azure diagnostics extension can transfer this data to an Azure storage account or send it to services like [Application Insights](#). You can use the data for debugging and troubleshooting, measuring performance, monitoring resource usage, traffic analysis and capacity planning, and auditing.

Versioning

See [Azure Diagnostics Versioning History](#).

Next steps

Choose which service you are trying to collect diagnostics on and use the following articles to get started. Use the general Azure diagnostics links for reference for specific tasks.

Web Apps

Note that Web Apps do not use Azure Diagnostics. Find the equivalent information at [Web Apps](#)

Cloud Services using Azure Diagnostics

- If using Visual Studio, see [Use Visual Studio to trace a Cloud Services application](#) to get started. Otherwise, see
- [How to monitor Cloud services using Azure Diagnostics](#)
- [Set up Azure Diagnostics in a Cloud Services Application](#)

For more advanced topics, see

- [Using Azure Diagnostics with Application Insights for Cloud Services](#)
- [Trace the flow of a Cloud Services application with Azure Diagnostics](#)
- [Use PowerShell to set up diagnostics on Cloud Services](#)

Virtual Machines using Azure Diagnostics

- If using Visual Studio, see [Use Visual Studio to trace Azure Virtual Machines](#) to get started. Otherwise, see
- [Set up Azure Diagnostics on an Azure Virtual Machine](#)

For more advanced topics, see

- [Use PowerShell to set up diagnostics on Azure Virtual Machines](#)
- [Create a Windows Virtual machine with monitoring and diagnostics using Azure Resource Manager Template](#)

Service Fabric using Azure Diagnostics

Get started at [Monitor a Service Fabric application](#). Many other Service Fabric diagnostics articles are available in the navigation tree on the left once you get to this article.

General Azure Diagnostics articles

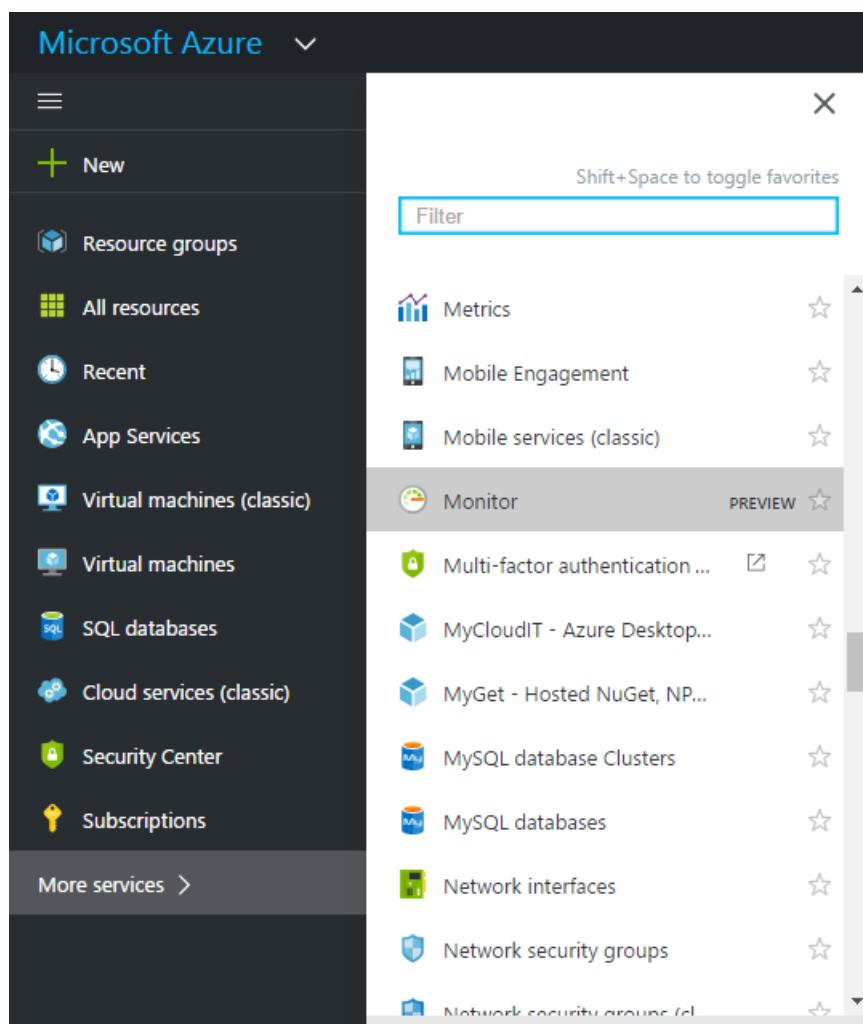
- [Azure Diagnostics Schema Configuration](#) - Learn how to change the schema file to collect and route diagnostics data. Note that you can also use Visual Studio to change the schema file.
- [How Azure Diagnostics data is stored in Azure Storage](#) - Know the names of the tables and blobs where the diagnostic data is written.
- Learn to [use Performance Counters in Azure Diagnostics](#).
- Learn to [Route Azure diagnostics information to Application Insights](#)
- If you have trouble with diagnostics starting or finding your data in Azure Storage tables, see [TroubleShooting Azure Diagnostics](#)

Get started with Azure Monitor

7/18/2017 • 4 min to read • [Edit Online](#)

Azure Monitor is the platform service that provides a single source for monitoring Azure resources. With Azure Monitor, you can visualize, query, route, archive, and take action on the metrics and logs coming from resources in Azure. You can work with this data using the Monitor portal blade, [Monitor PowerShell Cmdlets](#), [Cross-Platform CLI](#), or [Azure Monitor REST APIs](#). In this article, we walk through a few of the key components of Azure Monitor, using the portal for demonstration.

1. In the portal, navigate to **More services** and find the **Monitor** option. Click the star icon to add this option to your favorites list so that it is always easily accessible from the left-hand navigation bar.



2. Click the **Monitor** option to open up the **Monitor** blade. This blade brings together all your monitoring settings and data into one consolidated view. It first opens to the **Activity log** section.

OPERATION NAME	STATUS	TIME
ListKeys	Succeeded	Just now
ListKeys	Succeeded	3 min ago
ListKeys	Succeeded	3 min ago
ListKeys	Succeeded	6 min ago
ListKeys	Succeeded	8 min ago
ListKeys	Succeeded	8 min ago

Azure Monitor has three basic categories of monitoring data: The **activity log**, **metrics**, and **diagnostic logs**.

- Click **Activity log** to ensure that the activity log section is displayed.

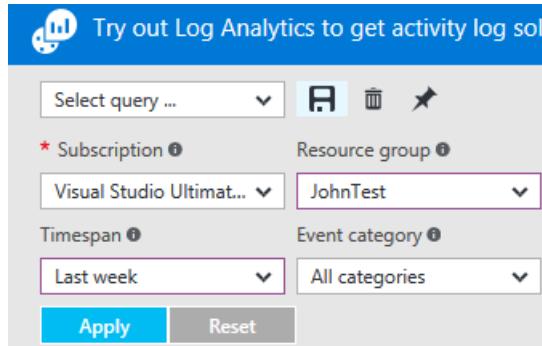
OPERATION NAME	STATUS	TIME	TIME STAMP	SUBSCRIPTION	EVENT INITIATED BY
ListKeys	Succeeded	2 min ago	Thu Sep 22 2...	Microsoft Azure Internal Consu...	AzureApplicationInsights
ListKeys	Succeeded	4 min ago	Thu Sep 22 2...	Microsoft Azure Internal Consu...	AzureApplicationInsights
ListKeys	Succeeded	4 min ago	Thu Sep 22 2...	Microsoft Azure Internal Consu...	AzureApplicationInsights
ListKeys	Succeeded	7 min ago	Thu Sep 22 2...	Microsoft Azure Internal Consu...	AzureApplicationInsights
ListKeys	Succeeded	9 min ago	Thu Sep 22 2...	Microsoft Azure Internal Consu...	AzureApplicationInsights
ListKeys	Succeeded	9 min ago	Thu Sep 22 2...	Microsoft Azure Internal Consu...	AzureApplicationInsights

The **activity log** describes all operations performed on resources in your subscription. Using the Activity Log, you can determine the 'what, who, and when' for any create, update, or delete operations on resources in your subscription. For example, the Activity Log tells you when a web app was stopped and who stopped

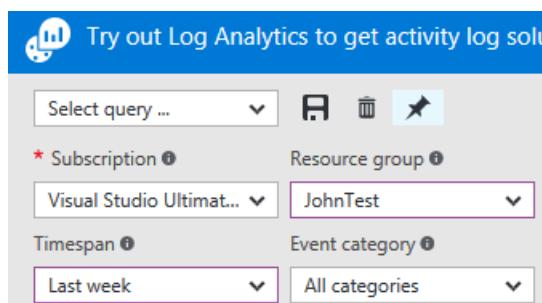
it. Activity Log events are stored in the platform and available to query for 90 days.

You can create and save queries for common filters, then pin the most important queries to a portal dashboard so you'll always know if events that meet your criteria have occurred.

4. Filter the view to a particular resource group over the last week, then click the **Save** button.

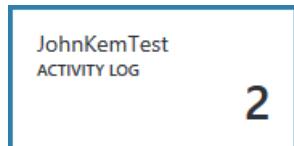


5. Now, click the **Pin** button.

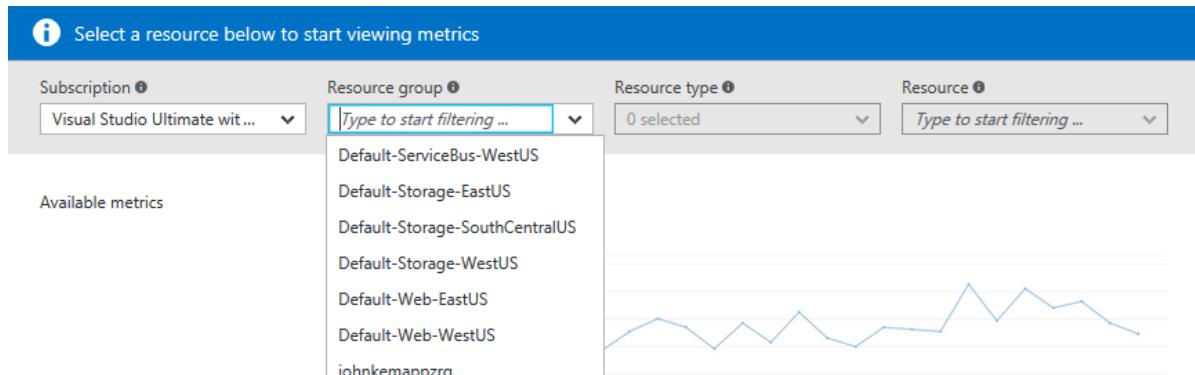


Most of the views in this walkthrough can be pinned to a dashboard. This helps you create a single source of information for operational data on your services.

6. Return to your dashboard. You can now see that the query (and number of results) is displayed in your dashboard. This is useful if you want to quickly see any high-profile actions that have occurred recently in your subscription, eg. a new role was assigned or a VM was deleted.



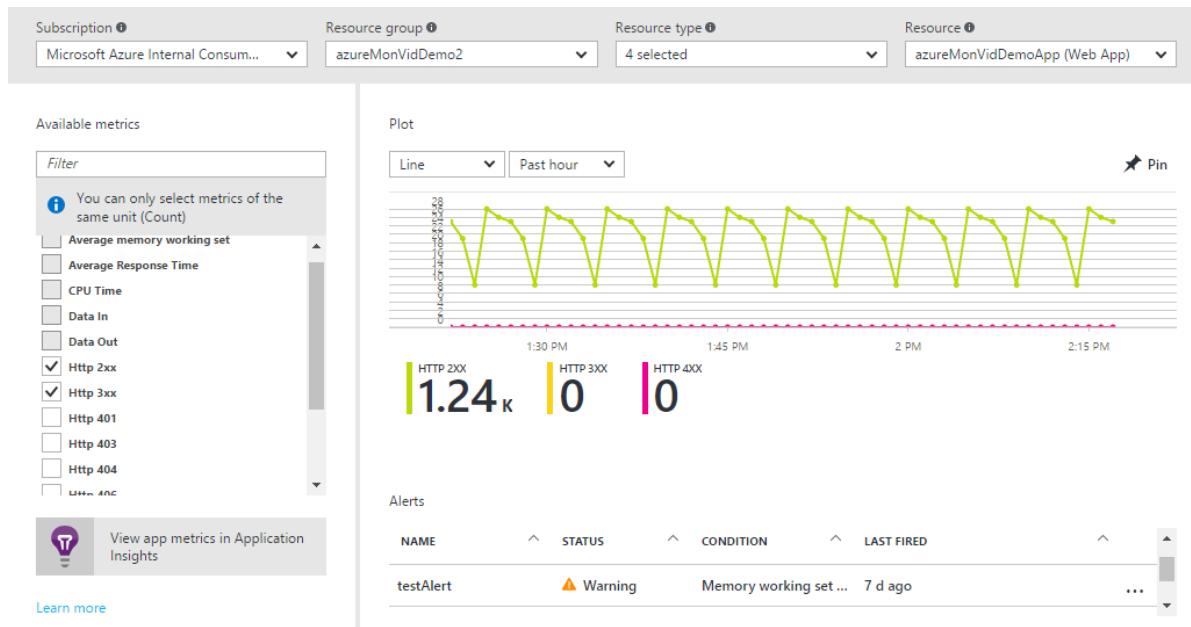
7. Return to the **Monitor** tile and click the **Metrics** section. You first need to select a resource by filtering and selecting using the drop down options at the top of the blade.



All Azure resources emit **metrics**. This view brings together all metrics in a single pane of glass so you can easily understand how your resources are performing.

8. Once you have selected a resource, all available metrics appear on the left side of the blade. You can chart

multiple metrics at once by selecting metrics and modify the graph type and time range. You can also view all metric alerts set on this resource.



NOTE

Some metrics are only available by enabling [Application Insights](#) and/or Windows or Linux Azure Diagnostics on your resource.

9. When you are happy with your chart, you can use the **Pin** button to pin it to your dashboard.

10. Return to the **Monitor** blade and click **Diagnostic logs**.

NAME	RESOURCE TYPE	RESOURCE GROUP	DIAGNOSTIC LOGS STATUS	STORAGE ACCOUNT	SERVICE BUS	LOG ANALYTICS
loanprocessingEH	Event Hubs	ContosoLoanApp1	Disabled			
LoanAppLA	Logic App	ContosoLoanApp1	Enabled	/subscription.../storag...		
simplealerttest1	Logic App	ContosoLoanApp1	Disabled			
simpletest2	Logic App	ContosoLoanApp1	Disabled			
loandatabasevm2-nsg	Network security group	ContosoLoanApp1	Enabled	/subscription.../storag...		
loandatabasevm1-nsg	Network security group	ContosoLoanApp1	Disabled			
demoDoWorkLA	Logic App	demoLogicApp	Disabled			
LoanAppASA	Stream Analytics job	LoanASA	Disabled			
batchtutorial	Batch Account	MDMTestBatch	Disabled			
loanprocessingEH	Event Hubs	ContosoLoanApp1	Disabled			

Diagnostic logs are logs emitted by a resource that provide data about the operation of that particular resource. For example, Network Security Group Rule Counters and Logic App Workflow Logs are both types of diagnostic logs. These logs can be stored in a storage account, streamed to an Event Hub, and/or sent to [Log Analytics](#). Log Analytics is Microsoft's operational intelligence product for advanced searching and alerting.

In the portal you can view and filter a list of all resources in your subscription to identify if they have diagnostic logs enabled.

11. Click a resource in the diagnostic logs blade. If diagnostic logs are being stored in a storage account, you will see a list of hourly logs that you can directly download.

Diagnostics settings

Storage account
[n4l6pxi0q07kcimdmlinuxusa](#)

Service bus namespace
[loanappsb1](#)

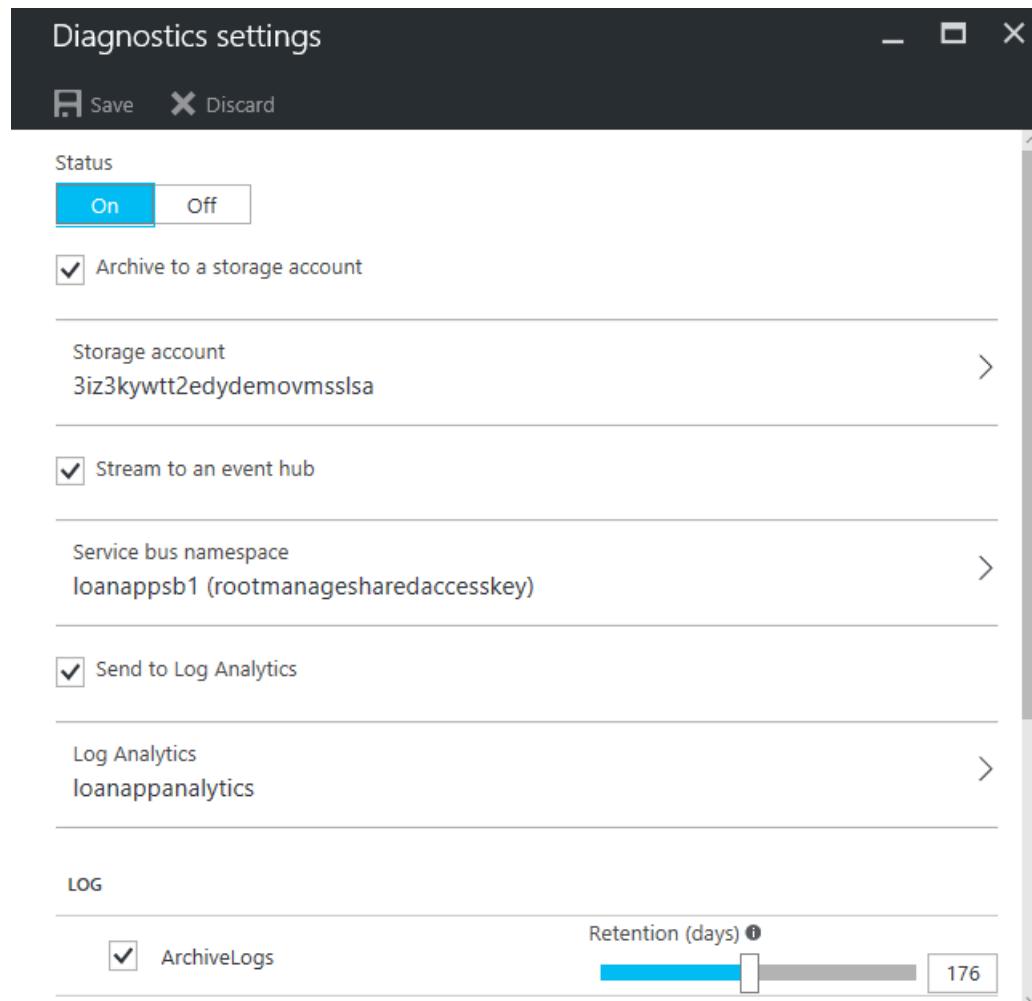
Log Analytics
[loanappanalytics](#)

* Log categories Timespan

2 selected Last 24 hours

LOG CATEGORIES	LAST UPDATED	SIZE
Network Security Group Event	22 h ago	146.615 KB
Network Security Group Event	21 h ago	146.615 KB
Network Security Group Event	20 h ago	146.615 KB
Network Security Group Event	19 h ago	146.615 KB

You can also click **Diagnostic Settings**, which allows you to set up or modify your settings for archival to a storage account, streaming to Event Hubs, or sending to a Log Analytics workspace.



If you have set up diagnostic logs to Log Analytics, you can then search them in the **Log search** section of

the portal.

12. Navigate to the **Alerts** section of the Monitor blade.

The screenshot shows the 'Alerts' blade in the Azure Monitor. At the top, there are filters for Subscription (Visual Studio Ultimate with MSDN), Source (All sources), Resource group (Type to start fil...), Resource type (0 selected), and Resource (Type to start fil...). Below the filters is a search bar labeled 'Filter alerts...'. A table lists two alerts:

NAME	STATUS	CONDITION	RESOURCE GR...	RESOURCE	LAST FIRED	...
Scale Up on CPU	⚠ Warning	CPU percentage...	JohnTest	johnkemtest	2 wk ago	...
Scale Up on CPU	⚠ Warning	CPU percentage...	JohnTest	johnkemtest	2 wk ago	...

Here you can manage all **alerts** on your Azure resources. This includes alerts on metrics, activity log events, Application Insights web tests (Locations), and Application Insights proactive diagnostics. Alerts can trigger an email to be sent or an HTTP POST to a webhook URL.

13. Click **Add metric alert** to create an alert.

The screenshot shows the 'Add rule' dialog box. It has several sections:

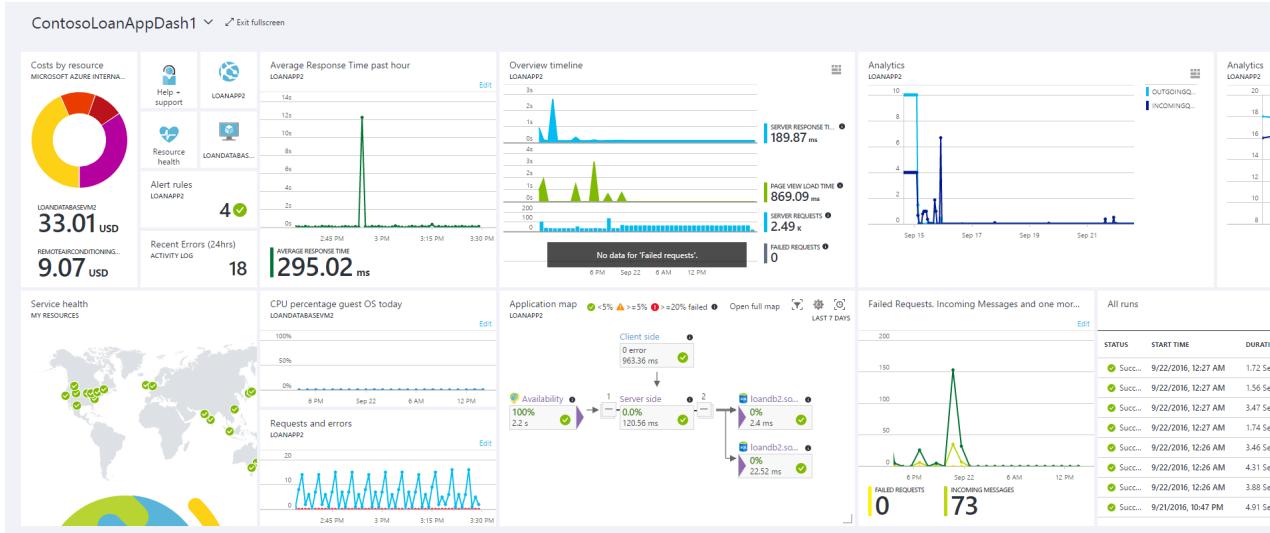
- Name:** New Metric Rule
- Description:** Alert on high CPU %
- Source:** Alert on Metrics
- Criteria:**
 - Subscription:** Visual Studio Ultimate with MSDN
 - Resource group:** JohnTest
 - Resource:** Filter
 - Metric:** CPU percentage guest OS
- OK** button at the bottom

You can then pin an alert to your dashboard to easily see its state at any time.

14. The Monitor section also includes links to [Application Insights](#) applications and [Log Analytics](#) management solutions. These other Microsoft products have deep integration with Azure Monitor.
15. If you are not using Application Insights or Log Analytics, chances are that Azure Monitor has a partnership

with your current monitoring, logging, and alerting products. See our [partners page](#) for a full list and instructions for how to integrate.

By following these steps and pinning all relevant tiles to a dashboard, you can create comprehensive views of your application and infrastructure like this one:



Next Steps

- Read the [Overview of Azure Monitor](#)

Get started with Autoscale in Azure

7/28/2017 • 3 min to read • [Edit Online](#)

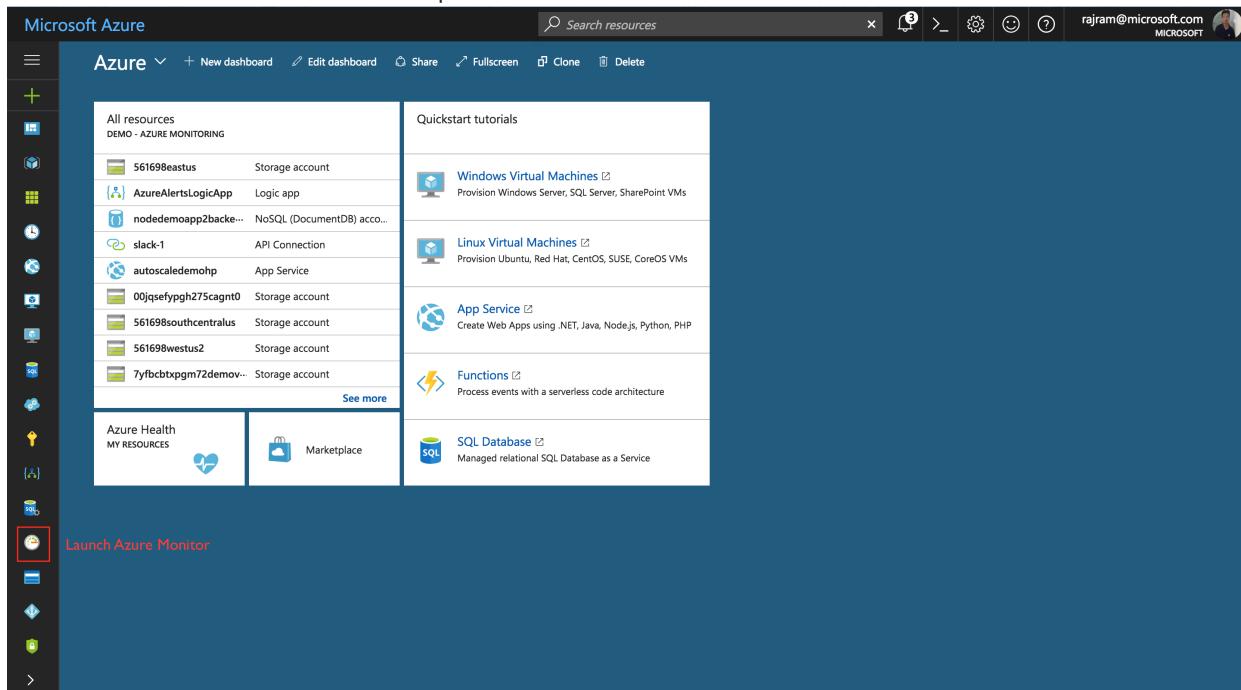
This article describes how to set up your Autoscale settings for your resource in the Microsoft Azure portal.

Azure Monitor Autoscale applies only to virtual machine scale sets, cloud services, Azure App Service plans, and App Service environments.

Discover the Autoscale settings in your subscription

You can discover all the resources for which Autoscale is applicable in Azure Monitor. Use the following steps for a step-by-step walkthrough:

1. Open the [Azure portal](#).
2. Click the Azure Monitor icon in the left pane.



3. Click **Autoscale** to view all the resources for which Autoscale is applicable, along with their current Autoscale status.

The screenshot shows the Microsoft Azure Monitor - Autoscale blade. On the left, there's a navigation menu with options like Explore, Manage, and Health. The 'Autoscale' option is selected and highlighted in blue. The main area displays a table of resources. The columns are: NAME, RESOURCE TYPE, RESOURCE GROUP, LOCATION, INSTANCE COUNT, and AUTOSCALE STATUS. The table lists various resources such as WebWorkerDemo, Production/WebRole1, demovmss, CPUBasedScaleAsp, HolidaySpikeAsp, staticscaleasp, WeekdayTrafficAsp, BrazilSouthPlan, CanadaCentralPlan, SouthCentralUSPlan, WestUS2Plan, contoso-mvc-app-asp, contoso-web-api-asp, contoso-web-react-a..., and nodedemoapp2sp. Most resources have an instance count of 1, except for CPUBasedScaleAsp which has 2, staticscaleasp which has 2, and contoso-mvc-app-asp which has 10. The 'AUTOSCALE STATUS' column indicates whether Autoscale is 'Not configured' (grey dot), 'Enabled' (green dot), or 'Disabled' (red dot). For example, 'demovmss' and 'CPUBasedScaleAsp' are marked as 'Enabled'.

You can use the filter pane at the top to scope down the list to select resources in a specific resource group, specific resource types, or a specific resource.

For each resource, you will find the current instance count and the Autoscale status. The Autoscale status can be:

- **Not configured:** You have not enabled Autoscale yet for this resource.
- **Enabled:** You have enabled Autoscale for this resource.
- **Disabled:** You have disabled Autoscale for this resource.

Create your first Autoscale setting

Let's now go through a simple step-by-step walkthrough to create your first Autoscale setting.

1. Open the **Autoscale** blade in Azure Monitor and select a resource that you want to scale. (The following steps use an App Service plan associated with a web app. You can [create your first ASP.NET web app in Azure in 5 minutes.](#))
2. Note that the current instance count is 1. Click **Enable autoscale**.

The screenshot shows the 'Autoscale setting' blade for the 'nodedemoapp2sp' App Service plan. At the top, there are buttons for Save, Discard, and Disable autoscale. Below that, there are tabs for Configure, Run history, JSON, and Notify, with 'Configure' being the active tab. Under 'Configure', there's a section for 'Override condition' with a slider for 'Instance count' set to 1. A note below says 'Your autoscale configuration is disabled. To reinstate your configuration, enable autoscale.' At the bottom, there's a prominent blue 'Enable autoscale' button.

3. Provide a name for the scale setting, and then click **Add a rule**. Notice the scale rule options that open as a context pane on the right side. By default, this sets the option to scale your instance count by 1 if the CPU percentage of the resource exceeds 70 percent. Leave it at its default values and click **Add**.

The screenshot shows the Azure portal interface for managing autoscale settings. On the left, a sidebar lists various monitoring and diagnostic tools. The main area is titled 'Autoscale setting' under 'nodeapp2sp (App Service plan)'. It includes tabs for 'Configure', 'Run history', 'JSON', and 'Notify'. The 'Configure' tab is active, showing a form to set up a scale rule. The 'Default' section contains a 'Scale mode' (set to 'Scale based on a metric') and a 'Rules' section with a note about scaling out based on CPU usage. Below this is a 'Schedule' section stating 'This scale condition is executed when none of the other scale condition(s) match'. To the right, a detailed configuration panel for a 'Scale rule' is open, showing fields for 'Metric source' (Current resource), 'Resource type' (App Service plans), 'Resource' (nodeapp2sp), 'Criteria' (Time aggregation: Average, Metric name: CPU Percentage, Operator: Greater than, Threshold: 70, Duration: 1 minute), and 'Action' (Operation: Increase count by 1). A blue 'Add' button is at the bottom.

4. You've now created your first scale rule. Note that the UX recommends best practices and states that "It is recommended to have at least one scale in rule." To do so:
 - a. Click **Add a rule**.
 - b. Set **Operator** to **Less than**.
 - c. Set **Threshold** to **20**.
 - d. Set **Operation** to **Decrease count by**.

You should now have a scale setting that scales out/scales in based on CPU usage.

The screenshot shows the same Azure portal interface after the scale rule has been configured. The 'Rules' section now displays two rules: 'Scale out' (When average CPU percentage is greater than 70, increase instance count by 1) and 'Scale in' (When average CPU percentage is less than 20, decrease instance count by 1). The rest of the interface remains consistent with the previous screenshot, showing the 'Default' section and the detailed 'Scale rule' configuration panel on the right.

5. Click **Save**.

Congratulations! You've now successfully created your first scale setting to autoscale your web app based on CPU usage.

NOTE

The same steps are applicable to get started with a virtual machine scale set or cloud service role.

Other considerations

Scale based on a schedule

In addition to scale based on CPU, you can set your scale differently for specific days of the week.

1. Click **Add a scale condition**.
2. Setting the scale mode and the rules is the same as the default condition.
3. Select **Repeat specific days** for the schedule.
4. Select the days and the start/end time for when the scale condition should be applied.

The screenshot shows the Microsoft Azure portal interface for managing autoscale settings. At the top, it says "Autoscale setting" and "nodedemoapp2sp (App Service plan)". Below this, there's a "Rules" section with a "Scale in" rule for "nodedemoapp2sp" when "Average CpuPercentage < 20", decreasing the instance count by 1. There are fields for "Instance limits" (Minimum 2, Maximum 5, Default 2). A "Schedule" section notes "This scale condition is executed when none of the other scale condition(s) match". A red box highlights the "Auto created scale condition" section, which includes fields for "Scale mode" (set to "Scale to a specific instance count"), "Instance count" (1), "Schedule" (set to "Repeat specific days" with Saturday and Sunday checked), "Repeat every" (checkboxes for Monday through Friday), "Timezone" (UTC-08:00 Pacific Time (US & Canada)), "Start time" (00:00), and "End time" (11:59). A "+ Add a scale condition" button is also present.

Scale differently on specific dates

In addition to scale based on CPU, you can set your scale differently for specific dates.

1. Click **Add a scale condition**.
2. Setting the scale mode and the rules is the same as the default condition.
3. Select **Specify start/end dates** for the schedule.
4. Select the start/end dates and the start/end time for when the scale condition should be applied.

View the scale history of your resource

Whenever your resource is scaled up or down, an event is logged in the activity log. You can view the scale history of your resource for the past 24 hours by switching to the **Run history** tab.

OPERATION NAME	STATUS	EVENT CATEG...	TIME	TIME STAMP	SUBSCRIPTION	EVENT INITIATED BY	RESOURCE TYPE	RESOURCE
Scaleup	Succeeded	Autoscale	14 h ago	Sun May 07 2...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaleup	Succeeded	Autoscale	14 h ago	Sat May 06 2...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaledown	Succeeded	Autoscale	2 d ago	Sat May 06 2...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaledown	Succeeded	Autoscale	2 d ago	Fri May 05 20...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaledown	Succeeded	Autoscale	2 d ago	Fri May 05 20...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaleup	Succeeded	Autoscale	2 d ago	Fri May 05 20...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaleup	Succeeded	Autoscale	2 d ago	Fri May 05 20...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaledown	Succeeded	Autoscale	2 d ago	Fri May 05 20...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaleup	Succeeded	Autoscale	2 d ago	Fri May 05 20...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaleup	Succeeded	Autoscale	2 d ago	Fri May 05 20...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaledown	Succeeded	Autoscale	4 d ago	Wed May 03 ...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaledown	Succeeded	Autoscale	4 d ago	Wed May 03 ...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp
Scaleup	Surrounded	Autoscale	4 d ago	Wed May 03 ...	Demo - Azure Monitoring	Microsoft.Insights/autoscale...	Microsoft.Web/serverFarms	serverFarms/contoso-mvc-app-asp

If you want to view the complete scale history (for up to 90 days), select [Click here to see more details](#). The activity log opens, with Autoscale pre-selected for your resource and category.

View the scale definition of your resource

Autoscale is an Azure Resource Manager resource. You can view the scale definition in JSON by switching to the **JSON** tab.

The screenshot shows the Microsoft Azure portal interface for managing an Autoscale setting. The top navigation bar includes 'Microsoft Azure', 'Monitor - Autoscale > Autoscale setting', a search bar, and user information. The main area is titled 'Autoscale setting' for 'contoso-mvc-app-asp (App Service plan)'. It has tabs for 'Configure', 'Run history', 'JSON', and 'Notify'. The 'JSON' tab is selected, displaying a large block of JSON code. The code defines an 'autoscaleSettings' object with properties like 'location', 'tags', and 'profiles' for different scaling conditions (e.g., 'Build conference scale condition', 'Weekend scale condition').

```
4 "type": "Microsoft.Insights autoscaleSettings",
5 "location": "West US 2",
6 "tags": {
7   "$type": "Microsoft.WindowsAzure.Management.Common.Storage.CasePreservedDictionary, Microsoft.WindowsAzure.Management.Common.Storage"
8 },
9 "properties": {
10   "profiles": [
11     {
12       "name": "Build conference scale condition",
13       "capacity": {
14         "minimum": "10",
15         "maximum": "10",
16         "default": "10"
17       },
18       "rules": [],
19       "fixedDate": {
20         "timeZone": "Pacific Standard Time",
21         "start": "2017-05-07T00:00:00Z",
22         "end": "2017-05-13T23:59:00Z"
23       },
24     },
25     {
26       "name": "Weekend scale condition",
27       "capacity": {
28         "minimum": "1",
29         "maximum": "1",
30         "default": "1"
31       },
32       "rules": []
33     }
34   ]
35 }
```

You can make changes in JSON directly, if required. These changes will be reflected after you save them.

Disable Autoscale and manually scale your instances

There might be times when you want to disable your current scale setting and manually scale your resource.

Click the **Disable autoscale** button at the top.

The screenshot shows the Microsoft Azure portal interface for managing an Autoscale setting. The top navigation bar includes 'Microsoft Azure', 'Monitor - Autoscale > Autoscale setting', a search bar, and user information. The main area is titled 'Autoscale setting' for 'contoso-mvc-app-asp (App Service plan)'. It has tabs for 'Configure', 'Run history', 'JSON', and 'Notify'. The 'Configure' tab is selected. A prominent button labeled 'Disable autoscale' is highlighted in blue. Below it, a message states 'Disable your current autoscale configuration? You can reinstate your configuration anytime.' There are 'Yes' and 'No' buttons at the bottom. At the bottom of the page, there is a note: 'Default Normal scale behavior1' and 'Scale mode: Scale based on a metric'.

NOTE

This option disables your configuration. However, you can get back to it after you enable Autoscale again.

You can now set the number of instances that you want to scale to manually.

The screenshot shows the Microsoft Azure portal interface for managing an Autoscale setting. The top navigation bar includes 'Microsoft Azure', 'Monitor - Autoscale > Autoscale setting', a search bar, and user information. The main area is titled 'Autoscale setting' for 'contoso-mvc-app-asp (App Service plan)'. It has tabs for 'Configure', 'Run history', 'JSON', and 'Notify'. The 'Configure' tab is selected. A section titled 'Override condition' contains a slider for 'Instance count' set to 4. Below the slider, a message says 'Your autoscale configuration is disabled. To reinstate your configuration, enable autoscale.' A blue 'Enable autoscale' button is visible at the bottom.

You can always return to Autoscale by clicking **Enable autoscale** and then **Save**.

Next steps

- [Create an Activity Log Alert to monitor all Autoscale engine operations on your subscription](#)
- [Create an Activity Log Alert to monitor all failed Autoscale scale-in/scale-out operations on your subscription](#)

Get started with roles, permissions, and security with Azure Monitor

7/18/2017 • 7 min to read • [Edit Online](#)

Many teams need to strictly regulate access to monitoring data and settings. For example, if you have team members who work exclusively on monitoring (support engineers, devops engineers) or if you use a managed service provider, you may want to grant them access to only monitoring data while restricting their ability to create, modify, or delete resources. This article shows how to quickly apply a built-in monitoring RBAC role to a user in Azure or build your own custom role for a user who needs limited monitoring permissions. It then discusses security considerations for your Azure Monitor-related resources and how you can limit access to the data they contain.

Built-in monitoring roles

Azure Monitor's built-in roles are designed to help limit access to resources in a subscription while still enabling those responsible for monitoring infrastructure to obtain and configure the data they need. Azure Monitor provides two out-of-the-box roles: A Monitoring Reader and a Monitoring Contributor.

Monitoring Reader

People assigned the Monitoring Reader role can view all monitoring data in a subscription but cannot modify any resource or edit any settings related to monitoring resources. This role is appropriate for users in an organization, such as support or operations engineers, who need to be able to:

- View monitoring dashboards in the portal and create their own private monitoring dashboards.
- Query for metrics using the [Azure Monitor REST API](#), [PowerShell cmdlets](#), or [cross-platform CLI](#).
- Query the Activity Log using the portal, Azure Monitor REST API, PowerShell cmdlets, or cross-platform CLI.
- View the [diagnostic settings](#) for a resource.
- View the [log profile](#) for a subscription.
- View autoscale settings.
- View alert activity and settings.
- Access Application Insights data and view data in AI Analytics.
- Search Log Analytics (OMS) workspace data including usage data for the workspace.
- View Log Analytics (OMS) management groups.
- Retrieve the Log Analytics (OMS) search schema.
- List Log Analytics (OMS) intelligence packs.
- Retrieve and execute Log Analytics (OMS) saved searches.
- Retrieve the Log Analytics (OMS) storage configuration.

NOTE

This role does not give read access to log data that has been streamed to an event hub or stored in a storage account. [See below](#) for information on configuring access to these resources.

Monitoring Contributor

People assigned the Monitoring Contributor role can view all monitoring data in a subscription and create or modify monitoring settings, but cannot modify any other resources. This role is a superset of the Monitoring Reader role, and is appropriate for members of an organization's monitoring team or managed service providers

who, in addition to the permissions above, also need to be able to:

- Publish monitoring dashboards as a shared dashboard.
- Set [diagnostic settings](#) for a resource.*
- Set the [log profile](#) for a subscription.*
- Set alert activity and settings.
- Create Application Insights web tests and components.
- List Log Analytics (OMS) workspace shared keys.
- Enable or disable Log Analytics (OMS) intelligence packs.
- Create and delete and execute Log Analytics (OMS) saved searches.
- Create and delete the Log Analytics (OMS) storage configuration.

*user must also separately be granted ListKeys permission on the target resource (storage account or event hub namespace) to set a log profile or diagnostic setting.

NOTE

This role does not give read access to log data that has been streamed to an event hub or stored in a storage account. [See below](#) for information on configuring access to these resources.

Monitoring permissions and custom RBAC roles

If the above built-in roles don't meet the exact needs of your team, you can [create a custom RBAC role](#) with more granular permissions. Below are the common Azure Monitor RBAC operations with their descriptions.

OPERATION	DESCRIPTION
Microsoft.Insights/AlertRules/[Read, Write, Delete]	Read/write/delete alert rules.
Microsoft.Insights/AlertRules/Incidents/Read	List incidents (history of the alert rule being triggered) for alert rules. This only applies to the portal.
Microsoft.Insights/AutoscaleSettings/[Read, Write, Delete]	Read/write/delete autoscale settings.
Microsoft.Insights/DiagnosticSettings/[Read, Write, Delete]	Read/write/delete diagnostic settings.
Microsoft.Insights/eventtypes/digestevents/Read	This permission is necessary for users who need access to Activity Logs via the portal.
Microsoft.Insights/eventtypes/values/Read	List Activity Log events (management events) in a subscription. This permission is applicable to both programmatic and portal access to the Activity Log.
Microsoft.Insights/LogDefinitions/Read	This permission is necessary for users who need access to Activity Logs via the portal.
Microsoft.Insights/MetricDefinitions/Read	Read metric definitions (list of available metric types for a resource).
Microsoft.Insights/Metrics/Read	Read metrics for a resource.

NOTE

Access to alerts, diagnostic settings, and metrics for a resource requires that the user has Read access to the resource type and scope of that resource. Creating ("write") a diagnostic setting or log profile that archives to a storage account or streams to event hubs requires the user to also have ListKeys permission on the target resource.

For example, using the above table you could create a custom RBAC role for an "Activity Log Reader" like this:

```
$role = Get-AzureRmRoleDefinition "Reader"
$role.Id = $null
$role.Name = "Activity Log Reader"
$role.Description = "Can view activity logs."
$role.Actions.Clear()
$role.Actions.Add("Microsoft.Insights/eventtypes/*")
$roleAssignableScopes.Clear()
$role.AssignableScopes.Add("/subscriptions/mySubscription")
New-AzureRmRoleDefinition -Role $role
```

Security considerations for monitoring data

Monitoring data—particularly log files—can contain sensitive information, such as IP addresses or user names.

Monitoring data from Azure comes in three basic forms:

1. The Activity Log, which describes all control-plane actions on your Azure subscription.
2. Diagnostic Logs, which are logs emitted by a resource.
3. Metrics, which are emitted by resources.

All three of these data types can be stored in a storage account or streamed to Event Hub, both of which are general-purpose Azure resources. Because these are general-purpose resources, creating, deleting, and accessing them is a privileged operation usually reserved for an administrator. We suggest that you use the following practices for monitoring-related resources to prevent misuse:

- Use a single, dedicated storage account for monitoring data. If you need to separate monitoring data into multiple storage accounts, never share usage of a storage account between monitoring and non-monitoring data, as this may inadvertently give those who only need access to monitoring data (eg. a third-party SIEM) access to non-monitoring data.
- Use a single, dedicated Service Bus or Event Hub namespace across all diagnostic settings for the same reason as above.
- Limit access to monitoring-related storage accounts or event hubs by keeping them in a separate resource group, and [use scope](#) on your monitoring roles to limit access to only that resource group.
- Never grant the ListKeys permission for either storage accounts or event hubs at subscription scope when a user only needs access to monitoring data. Instead, give these permissions to the user at a resource or resource group (if you have a dedicated monitoring resource group) scope.

Limiting access to monitoring-related storage accounts

When a user or application needs access to monitoring data in a storage account, you should [generate an Account SAS](#) on the storage account that contains monitoring data with service-level read-only access to blob storage. In PowerShell, this might look like:

```
$context = New-AzureStorageContext -ConnectionString "[connection string for your monitoring Storage Account]"
$token = New-AzureStorageAccountSASToken -ResourceType Service -Service Blob -Permission "r" -Context $context
```

You can then give the token to the entity that needs to read from that storage account, and it can list and read from

all blobs in that storage account.

Alternatively, if you need to control this permission with RBAC, you can grant that entity the Microsoft.Storage/storageAccounts/listkeys/action permission on that particular storage account. This is necessary for users who need to be able to set a diagnostic setting or log profile to archive to a storage account. For example, you could create the following custom RBAC role for a user or application that only needs to read from one storage account:

```
$role = Get-AzureRmRoleDefinition "Reader"
$role.Id = $null
$role.Name = "Monitoring Storage Account Reader"
$role.Description = "Can get the storage account keys for a monitoring storage account."
$role.Actions.Clear()
$role.Actions.Add("Microsoft.Storage/storageAccounts/listkeys/action")
$role.Actions.Add("Microsoft.Storage/storageAccounts/Read")
$roleAssignableScopes.Clear()
$role.AssignableScopes.Add("/subscriptions/mySubscription/resourceGroups/myResourceGroup/providers/Microsoft.Storage/storageAccounts/myMonitoringStorageAccount")
New-AzureRmRoleDefinition -Role $role
```

WARNING

The ListKeys permission enables the user to list the primary and secondary storage account keys. These keys grant the user all signed permissions (read, write, create blobs, delete blobs, etc.) across all signed services (blob, queue, table, file) in that storage account. We recommend using an Account SAS described above when possible.

Limiting access to monitoring-related event hubs

A similar pattern can be followed with event hubs, but first you need to create a dedicated Listen authorization rule. If you want to grant access to an application that only needs to listen to monitoring-related event hubs, do the following:

1. Create a shared access policy on the event hub(s) that were created for streaming monitoring data with only Listen claims. This can be done in the portal. For example, you might call it "monitoringReadOnly." If possible, you will want to give that key directly to the consumer and skip the next step.
2. If the consumer needs to be able to get the key ad-hoc, grant the user the ListKeys action for that event hub. This is also necessary for users who need to be able to set a diagnostic setting or log profile to stream to event hubs. For example, you might create an RBAC rule:

```
$role = Get-AzureRmRoleDefinition "Reader"
$role.Id = $null
$role.Name = "Monitoring Event Hub Listener"
$role.Description = "Can get the key to listen to an event hub streaming monitoring data."
$role.Actions.Clear()
$role.Actions.Add("Microsoft.ServiceBus/namespaces/authorizationrules/listkeys/action")
$role.Actions.Add("Microsoft.ServiceBus/namespaces/Read")
$roleAssignableScopes.Clear()
$role.AssignableScopes.Add("/subscriptions/mySubscription/resourceGroups/myResourceGroup/providers/Microsoft.ServiceBus/namespaces/mySBNamespace")
New-AzureRmRoleDefinition -Role $role
```

Next steps

- [Read about RBAC and permissions in Resource Manager](#)
- [Read the overview of monitoring in Azure](#)

Create metric alerts in Azure Monitor for Azure services - Azure portal

6/27/2017 • 2 min to read • [Edit Online](#)

Overview

This article shows you how to set up Azure metric alerts using the Azure portal.

You can receive an alert based on monitoring metrics for, or events on, your Azure services.

- **Metric values** - The alert triggers when the value of a specified metric crosses a threshold you assign in either direction. That is, it triggers both when the condition is first met and then afterwards when that condition is no longer being met.
- **Activity log events** - An alert can trigger on *every* event, or, only when a certain events occurs. To learn more about activity log alerts [click here](#)

You can configure a metric alert to do the following when it triggers:

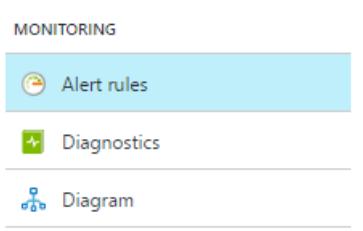
- send email notifications to the service administrator and co-administrators
- send email to additional emails that you specify.
- call a webhook
- start execution of an Azure runbook (only from the Azure portal)

You can configure and get information about metric alert rules using

- [Azure portal](#)
- [PowerShell](#)
- [command-line interface \(CLI\)](#)
- [Azure Monitor REST API](#)

Create an alert rule on a metric with the Azure portal

1. In the [portal](#), locate the resource you are interested in monitoring and select it.
2. Select **Alerts** or **Alert rules** under the MONITORING section. The text and icon may vary slightly for different resources.



3. Select the **Add alert** command and fill in the fields.

Add an alert rule

* Resource *?*
SampleVM (virtualMachines)

* Name *?*
MyNewVMArt

Description
Description

* Metric *?*
Memory percentage

* Condition
greater than

* Threshold *?*
90 %

* Period *?*
Over the last 5 minutes

Email owners, contributors, and readers

Additional administrator email(s)
admin@contoso.com

Webhook *?*
http://www.contoso.com/dowork?param

Learn more about configuring webhooks

Take Action *?* >
Run a runbook from this alert

OK

4. **Name** your alert rule, and choose a **Description**, which also shows in notification emails.
5. Select the **Metric** you want to monitor, then choose a **Condition** and **Threshold** value for the metric. Also chose the **Period** of time that the metric rule must be satisfied before the alert triggers. So for example, if you use the period "PT5M" and your alert looks for CPU above 80%, the alert triggers when the CPU has been consistently above 80% for 5 minutes. Once the first trigger occurs, it again triggers when the CPU stays below 80% for 5 minutes. The CPU measurement occurs every 1 minute.
6. Check **Email owners...** if you want administrators and co-administrators to be emailed when the alert fires.

7. If you want additional emails to receive a notification when the alert fires, add them in the **Additional Administrator email(s)** field. Separate multiple emails with semi-colons -
email@contoso.com;email2@contoso.com
8. Put in a valid URI in the **Webhook** field if you want it called when the alert fires.
9. If you use Azure Automation, you can select a Runbook to be run when the alert fires.
10. Select **OK** when done to create the alert.

Within a few minutes, the alert is active and triggers as previously described.

Managing your alerts

Once you have created an alert, you can select it and:

- View a graph showing the metric threshold and the actual values from the previous day.
- Edit or delete it.
- **Disable** or **Enable** it if you want to temporarily stop or resume receiving notifications for that alert.

Next steps

- Get an [overview of Azure monitoring](#) including the types of information you can collect and monitor.
- Learn more about [configuring webhooks in alerts](#).
- Learn more about [configuring alerts on Activity log events](#).
- Learn more about [Azure Automation Runbooks](#).
- Get an [overview of diagnostic logs](#) and collect detailed high-frequency metrics on your service.
- Get an [overview of metrics collection](#) to make sure your service is available and responsive.

Create metric alerts in Azure Monitor for Azure services - Cross-platform CLI

6/27/2017 • 3 min to read • [Edit Online](#)

Overview

This article shows you how to set up Azure metric alerts using the cross-platform Command Line Interface (CLI).

NOTE

Azure Monitor is the new name for what was called "Azure Insights" until Sept 25th, 2016. However, the namespaces and thus the commands below still contain the "insights".

You can receive an alert based on monitoring metrics for, or events on, your Azure services.

- **Metric values** - The alert triggers when the value of a specified metric crosses a threshold you assign in either direction. That is, it triggers both when the condition is first met and then afterwards when that condition is no longer being met.
- **Activity log events** - An alert can trigger on *every* event, or, only when a certain events occurs. To learn more about activity log alerts [click here](#)

You can configure a metric alert to do the following when it triggers:

- send email notifications to the service administrator and co-administrators
- send email to additional emails that you specify.
- call a webhook
- start execution of an Azure runbook (only from the Azure portal at this time)

You can configure and get information about metric alert rules using

- [Azure portal](#)
- [PowerShell](#)
- [command-line interface \(CLI\)](#)
- [Azure Monitor REST API](#)

You can always receive help for commands by typing a command and putting -help at the end. For example:

```
```console
azure insights alerts -help
azure insights alerts actions email create -help
```
```

Create alert rules using the CLI

1. Perform the Prerequisites and login to Azure. See [Azure Monitor CLI samples](#). In short, install the CLI and run these commands. They get you logged in, show what subscription you are using, and prepare you to run Azure Monitor commands.

```
azure login  
azure account show  
azure config mode arm
```

2. To list existing rules on a resource group, use the following form **azure insights alerts rule list [options] <resourceGroup>**

```
azure insights alerts rule list myresourcegroupname
```

3. To create a rule, you need to have several important pieces of information first.

- The **Resource ID** for the resource you want to set an alert for
- The **metric definitions** available for that resource

One way to get the Resource ID is to use the Azure portal. Assuming the resource is already created, select it in the portal. Then in the next blade, select *Properties* under the *Settings* section. The *RESOURCE ID* is a field in the next blade. Another way is to use the [Azure Resource Explorer](#).

An example resource id for a web app is

```
/subscriptions/dededede-7aa0-407d-a6fb-  
eb20c8bd1192/resourceGroups/myresourcegroupname/providers/Microsoft.Web/sites/mywebsitename
```

To get a list of the available metrics and units for those metrics for the previous resource example, use the following CLI command:

```
azure insights metrics list /subscriptions/dededede-7aa0-407d-a6fb-  
eb20c8bd1192/resourceGroups/myresourcegroupname/providers/Microsoft.Web/sites/mywebsitename PT1M
```

PT1M is the granularity of the available measurement (1-minute intervals). Using different granularities gives you different metric options.

4. To create a metric-based alert rule, use a command of the following form:

```
azure insights alerts rule metric set [options] <ruleName> <location> <resourceGroup>  
<windowSize> <operator> <threshold> <targetResourceId> <metricName>  
<timeAggregationOperator>
```

The following example sets up an alert on a web site resource. The alert triggers whenever it consistently receives any traffic for 5 minutes and again when it receives no traffic for 5 minutes.

```
azure insights alerts rule metric set myrule eastus myresourcegroup PT5M GreaterThan 2  
/subscriptions/dededede-7aa0-407d-a6fb-  
eb20c8bd1192/resourceGroups/myresourcegroupname/providers/Microsoft.Web/sites/mywebsitename  
BytesReceived Total
```

5. To create webhook or send email when a metric alert fires, first create the email and/or webhooks. Then create the rule immediately afterwards. You cannot associate webhook or emails with already created rules using the CLI.

```
azure insights alerts actions email create --customEmails myemail@contoso.com

azure insights alerts actions webhook create https://www.contoso.com

azure insights alerts rule metric set myrulewithwebhookandemail eastus myresourcegroup PT5M
GreaterThan 2 /subscriptions/dedede-7aa0-407d-a6fb-
eb20c8bd1192/resourceGroups/myresourcegroupname/providers/Microsoft.Web/sites/mywebsitename
BytesReceived Total
```

6. You can verify that your alerts have been created properly by looking at an individual rule.

```
azure insights alerts rule list myresourcegroup --ruleName myrule
```

7. To delete rules, use a command of the form:

insights alerts rule delete [options] <resourceGroup> <ruleName>

These commands delete the rules previously created in this article.

```
azure insights alerts rule delete myresourcegroup myrule
azure insights alerts rule delete myresourcegroup myrulewithwebhookandemail
azure insights alerts rule delete myresourcegroup myActivityLogRule
```

Next steps

- Get an [overview of Azure monitoring](#) including the types of information you can collect and monitor.
- Learn more about [configuring webhooks in alerts](#).
- Learn more about [configuring alerts on Activity log events](#).
- Learn more about [Azure Automation Runbooks](#).
- Get an [overview of collecting diagnostic logs](#) to collect detailed high-frequency metrics on your service.
- Get an [overview of metrics collection](#) to make sure your service is available and responsive.

Create metric alerts in Azure Monitor for Azure services - PowerShell

6/27/2017 • 3 min to read • [Edit Online](#)

Overview

This article shows you how to set up Azure metric alerts using PowerShell.

You can receive an alert based on monitoring metrics for, or events on, your Azure services.

- **Metric values** - The alert triggers when the value of a specified metric crosses a threshold you assign in either direction. That is, it triggers both when the condition is first met and then afterwards when that condition is no longer being met.
- **Activity log events** - An alert can trigger on *every* event, or, only when a certain events occurs. To learn more about activity log alerts [click here](#)

You can configure a metric alert to do the following when it triggers:

- send email notifications to the service administrator and co-administrators
- send email to additional emails that you specify.
- call a webhook
- start execution of an Azure runbook (only from the Azure portal)

You can configure and get information about alert rules using

- [Azure portal](#)
- [PowerShell](#)
- [command-line interface \(CLI\)](#)
- [Azure Monitor REST API](#)

For additional information, you can always type `Get-Help` and then the PowerShell command you want help on.

Create alert rules in PowerShell

1. Log in to Azure.

```
Login-AzureRmAccount
```

2. Get a list of the subscriptions you have available. Verify that you are working with the right subscription. If not, set it to the right one using the output from `Get-AzureRmSubscription`.

```
Get-AzureRmSubscription  
Get-AzureRmContext  
Set-AzureRmContext -SubscriptionId <subscriptionid>
```

3. To list existing rules on a resource group, use the following command:

```
Get-AzureRmAlertRule -ResourceGroup <myresourcegroup> -DetailedOutput
```

4. To create a rule, you need to have several important pieces of information first.

- The **Resource ID** for the resource you want to set an alert for
- The **metric definitions** available for that resource

One way to get the Resource ID is to use the Azure portal. Assuming the resource is already created, select it in the portal. Then in the next blade, select *Properties* under the *Settings* section. **RESOURCE ID** is a field in the next blade. Another way is to use the [Azure Resource Explorer](#).

An example Resource ID for a web app is

```
/subscriptions/dedede-7aa0-407d-a6fb-  
eb20c8bd1192/resourceGroups/myresourcegroupname/providers/Microsoft.Web/sites/mywebsitename
```

You can use `Get-AzureRmMetricDefinition` to view the list of all metric definitions for a specific resource.

```
Get-AzureRmMetricDefinition -ResourceId <resource_id>
```

The following example generates a table with the metric Name and the Unit for that metric.

```
Get-AzureRmMetricDefinition -ResourceId <resource_id> | Format-Table -Property Name,Unit
```

A full list of available options for `Get-AzureRmMetricDefinition` is available by running `Get-MetricDefinitions`.

5. The following example sets up an alert on a web site resource. The alert triggers whenever it consistently receives any traffic for 5 minutes and again when it receives no traffic for 5 minutes.

```
Add-AzureRmMetricAlertRule -Name myMetricRuleWithWebhookAndEmail -Location "East US" -ResourceGroup  
myresourcegroup -TargetResourceId /subscriptions/dedede-7aa0-407d-a6fb-  
eb20c8bd1192/resourceGroups/myresourcegroupname/providers/Microsoft.Web/sites/mywebsitename -  
MetricName "BytesReceived" -Operator GreaterThan -Threshold 2 -WindowSize 00:05:00 -  
TimeAggregationOperator Total -Description "alert on any website activity"
```

6. To create webhook or send email when an alert triggers, first create the email and/or webhooks. Then immediately create the rule afterwards with the `-Actions` tag and as shown in the following example. You cannot associate webhook or emails with already created rules via PowerShell.

```
$actionEmail = New-AzureRmAlertRuleEmail -CustomEmail myname@company.com  
$actionWebhook = New-AzureRmAlertRuleWebhook -ServiceUri https://www.contoso.com?token=mytoken  
  
Add-AzureRmMetricAlertRule -Name myMetricRuleWithWebhookAndEmail -Location "East US" -ResourceGroup  
myresourcegroup -TargetResourceId /subscriptions/dedede-7aa0-407d-a6fb-  
eb20c8bd1192/resourceGroups/myresourcegroupname/providers/Microsoft.Web/sites/mywebsitename -  
MetricName "BytesReceived" -Operator GreaterThan -Threshold 2 -WindowSize 00:05:00 -  
TimeAggregationOperator Total -Actions $actionEmail, $actionWebhook -Description "alert on any website  
activity"
```

7. To verify that your alerts have been created properly by looking at the individual rules.

```
Get-AzureRmAlertRule -Name myMetricRuleWithWebhookAndEmail -ResourceGroup myresourcegroup -DetailedOutput
```

```
Get-AzureRmAlertRule -Name myLogAlertRule -ResourceGroup myresourcegroup -DetailedOutput
```

8. Delete your alerts. These commands delete the rules created previously in this article.

```
Remove-AzureRmAlertRule -ResourceGroup myresourcegroup -Name myrule  
Remove-AzureRmAlertRule -ResourceGroup myresourcegroup -Name myMetricRuleWithWebhookAndEmail  
Remove-AzureRmAlertRule -ResourceGroup myresourcegroup -Name myLogAlertRule
```

Next steps

- [Get an overview of Azure monitoring](#) including the types of information you can collect and monitor.
- Learn more about [configuring webhooks in alerts](#).
- Learn more about [configuring alerts on Activity log events](#).
- Learn more about [Azure Automation Runbooks](#).
- Get an [overview of collecting diagnostic logs](#) to collect detailed high-frequency metrics on your service.
- Get an [overview of metrics collection](#) to make sure your service is available and responsive.

Configure a webhook on an Azure metric alert

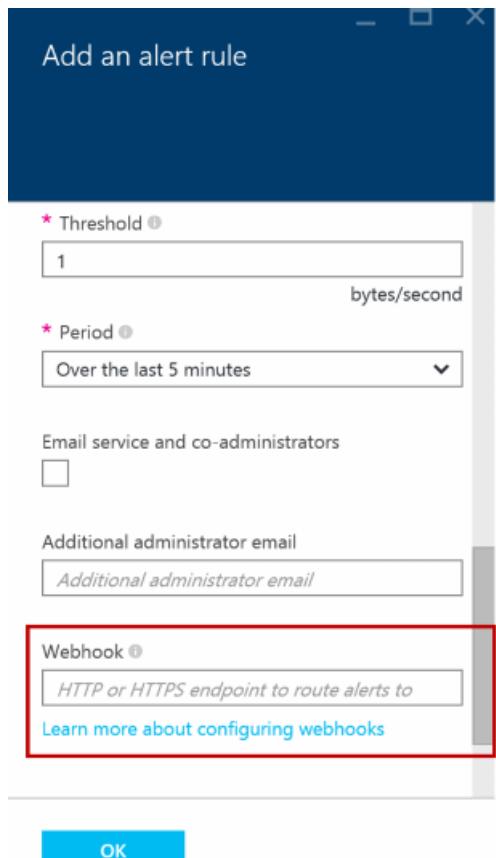
7/18/2017 • 3 min to read • [Edit Online](#)

Webhooks allow you to route an Azure alert notification to other systems for post-processing or custom actions. You can use a webhook on an alert to route it to services that send SMS, log bugs, notify a team via chat/messaging services, or do any number of other actions. This article describes how to set a webhook on an Azure metric alert and what the payload for the HTTP POST to a webhook looks like. For information on the setup and schema for an Azure Activity Log alert (alert on events), [see this page instead](#).

Azure alerts HTTP POST the alert contents in JSON format, schema defined below, to a webhook URI that you provide when creating the alert. This URI must be a valid HTTP or HTTPS endpoint. Azure posts one entry per request when an alert is activated.

Configuring webhooks via the portal

You can add or update the webhook URI in the Create/Update Alerts screen in the [portal](#).



You can also configure an alert to post to a webhook URI using the [Azure PowerShell Cmdlets](#), [Cross-Platform CLI](#), or [Azure Monitor REST API](#).

Authenticating the webhook

The webhook can authenticate using token-based authorization. The webhook URI is saved with a token ID, eg.

```
https://mysamplealert/webcallback?tokenid=sometokenid&someparameter=somevalue
```

Payload schema

The POST operation contains the following JSON payload and schema for all metric-based alerts.

```
{
  "status": "Activated",
  "context": {
    "timestamp": "2015-08-14T22:26:41.9975398Z",
    "id": "/subscriptions/s1/resourceGroups/useast/providers/microsoft.insights/alertrules/ruleName1",
    "name": "ruleName1",
    "description": "some description",
    "conditionType": "Metric",
    "condition": {
      "metricName": "Requests",
      "metricUnit": "Count",
      "metricValue": "10",
      "threshold": "10",
      "windowSize": "15",
      "timeAggregation": "Average",
      "operator": "GreaterThanOrEqual"
    },
    "subscriptionId": "s1",
    "resourceGroupName": "useast",
    "resourceName": "mysite1",
    "resourceType": "microsoft.foo/sites",
    "resourceId": "/subscriptions/s1/resourceGroups/useast/providers/microsoft.foo/sites/mysite1",
    "resourceRegion": "centralus",
    "portalLink": "https://portal.azure.com/#resource/subscriptions/s1/resourceGroups/useast/providers/microsoft.foo/sites/mysite1"
  },
  "properties": {
    "key1": "value1",
    "key2": "value2"
  }
}
```

| FIELD | MANDATORY | FIXED SET OF VALUES | NOTES |
|---------------|-----------|-------------------------|---|
| status | Y | "Activated", "Resolved" | Status for the alert based off of the conditions you have set. |
| context | Y | | The alert context. |
| timestamp | Y | | The time at which the alert was triggered. |
| id | Y | | Every alert rule has a unique id. |
| name | Y | | The alert name. |
| description | Y | | Description of the alert. |
| conditionType | Y | "Metric", "Event" | Two types of alerts are supported. One based on a metric condition and the other based on an event in the Activity Log. Use this value to check if the alert is based on metric or event. |

| FIELD | MANDATORY | FIXED SET OF VALUES | NOTES |
|-------------------|-------------------|--|--|
| condition | Y | | The specific fields to check for based on the conditionType. |
| metricName | for Metric alerts | | The name of the metric that defines what the rule monitors. |
| metricUnit | for Metric alerts | "Bytes", "BytesPerSecond", "Count", "CountPerSecond", "Percent", "Seconds" | The unit allowed in the metric. Allowed values are listed here . |
| metricValue | for Metric alerts | | The actual value of the metric that caused the alert. |
| threshold | for Metric alerts | | The threshold value at which the alert is activated. |
| windowSize | for Metric alerts | | The period of time that is used to monitor alert activity based on the threshold. Must be between 5 minutes and 1 day. ISO 8601 duration format. |
| timeAggregation | for Metric alerts | "Average", "Last", "Maximum", "Minimum", "None", "Total" | How the data that is collected should be combined over time. The default value is Average. Allowed values are listed here . |
| operator | for Metric alerts | | The operator used to compare the current metric data to the set threshold. |
| subscriptionId | Y | | Azure subscription ID. |
| resourceGroupName | Y | | Name of the resource group for the impacted resource. |
| resourceName | Y | | Resource name of the impacted resource. |
| resourceType | Y | | Resource type of the impacted resource. |
| resourceId | Y | | Resource ID of the impacted resource. |
| resourceRegion | Y | | Region or location of the impacted resource. |
| portalLink | Y | | Direct link to the portal resource summary page. |

| FIELD | MANDATORY | FIXED SET OF VALUES | NOTES |
|------------|-----------|---------------------|--|
| properties | N | Optional | <p>Set of <code><Key, Value></code> pairs (i.e. <code>Dictionary<String, String></code>) that includes details about the event. The properties field is optional. In a custom UI or Logic app-based workflow, users can enter key/values that can be passed via the payload. The alternate way to pass custom properties back to the webhook is via the webhook uri itself (as query parameters)</p> |

NOTE

The properties field can only be set using the [Azure Monitor REST API](#).

Next steps

- Learn more about Azure alerts and webhooks in the video [Integrate Azure Alerts with PagerDuty](#)
- [Execute Azure Automation scripts \(Runbooks\) on Azure alerts](#)
- [Use Logic App to send an SMS via Twilio from an Azure alert](#)
- [Use Logic App to send a Slack message from an Azure alert](#)
- [Use Logic App to send a message to an Azure Queue from an Azure alert](#)

Create a metric alert with a Resource Manager template

7/18/2017 • 4 min to read • [Edit Online](#)

This article shows how you can use an [Azure Resource Manager template](#) to configure Azure metric alerts. This enables you to automatically set up alerts on your resources when they are created to ensure that all resources are monitored correctly.

The basic steps are as follows:

1. Create a template as a JSON file that describes how to create the alert.
2. [Deploy the template using any deployment method](#).

Below we describe how to create a Resource Manager template first for an alert alone, then for an alert during the creation of another resource.

Resource Manager template for a metric alert

To create an alert using a Resource Manager template, you create a resource of type

`Microsoft.Insights/alertRules` and fill in all related properties. Below is a template that creates an alert rule.

```
{  
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",  
    "contentVersion": "1.0.0.0",  
    "parameters": {  
        "alertName": {  
            "type": "string",  
            "metadata": {  
                "description": "Name of alert"  
            }  
        },  
        "alertDescription": {  
            "type": "string",  
            "defaultValue": "",  
            "metadata": {  
                "description": "Description of alert"  
            }  
        },  
        "isEnabled": {  
            "type": "bool",  
            "defaultValue": true,  
            "metadata": {  
                "description": "Specifies whether alerts are enabled"  
            }  
        },  
        "resourceId": {  
            "type": "string",  
            "defaultValue": "",  
            "metadata": {  
                "description": "Resource ID of the resource emitting the metric that will be used for the  
comparison."  
            }  
        },  
        "metricName": {  
            "type": "string",  
            "defaultValue": "",  
            "metadata": {  
                "description": "Name of the metric used in the comparison to activate the alert."  
            }  
        }  
    }  
}
```

```
        }
    },
    "operator": {
        "type": "string",
        "defaultValue": "GreaterThan",
        "allowedValues": [
            "GreaterThan",
            "GreaterThanOrEqual",
            "LessThan",
            "LessThanOrEqual"
        ],
        "metadata": {
            "description": "Operator comparing the current value with the threshold value."
        }
    },
    "threshold": {
        "type": "string",
        "defaultValue": "",
        "metadata": {
            "description": "The threshold value at which the alert is activated."
        }
    },
    "aggregation": {
        "type": "string",
        "defaultValue": "Average",
        "allowedValues": [
            "Average",
            "Last",
            "Maximum",
            "Minimum",
            "Total"
        ],
        "metadata": {
            "description": "How the data that is collected should be combined over time."
        }
    },
    "windowSize": {
        "type": "string",
        "defaultValue": "PT5M",
        "metadata": {
            "description": "Period of time used to monitor alert activity based on the threshold. Must be between five minutes and one day. ISO 8601 duration format."
        }
    },
    "sendToServiceOwners": {
        "type": "bool",
        "defaultValue": true,
        "metadata": {
            "description": "Specifies whether alerts are sent to service owners"
        }
    },
    "customEmailAddresses": {
        "type": "string",
        "defaultValue": "",
        "metadata": {
            "description": "Comma-delimited email addresses where the alerts are also sent"
        }
    },
    "webhookUrl": {
        "type": "string",
        "defaultValue": "",
        "metadata": {
            "description": "URL of a webhook that will receive an HTTP POST when the alert activates."
        }
    },
    "variables": {
        "customEmails": "[split(parameters('customEmailAddresses'), ',')]"
    }
}
```

```

        ],
        "resources": [
            {
                "type": "Microsoft.Insights/alertRules",
                "name": "[parameters('alertName')]",
                "location": "[resourceGroup().location]",
                "apiVersion": "2016-03-01",
                "properties": {
                    "name": "[parameters('alertName')]",
                    "description": "[parameters('alertDescription')]",
                    "isEnabled": "[parameters('isEnabled')]",
                    "condition": {
                        "odata.type": "Microsoft.Azure.Management.Insights.Models.ThresholdRuleCondition",
                        "dataSource": {
                            "odata.type": "Microsoft.Azure.Management.Insights.Models.RuleMetricDataSource",
                            "resourceUri": "[parameters('resourceId')]",
                            "metricName": "[parameters('metricName')]"
                        },
                        "operator": "[parameters('operator')]",
                        "threshold": "[parameters('threshold')]",
                        "windowSize": "[parameters('windowSize')]",
                        "timeAggregation": "[parameters('aggregation')]"
                    },
                    "actions": [
                        {
                            "odata.type": "Microsoft.Azure.Management.Insights.Models.RuleEmailAction",
                            "sendToServiceOwners": "[parameters('sendToServiceOwners')]",
                            "customEmails": "[variables('customEmails')]"
                        },
                        {
                            "odata.type": "Microsoft.Azure.Management.Insights.Models.RuleWebhookAction",
                            "serviceUri": "[parameters('webhookUrl')]",
                            "properties": {}
                        }
                    ]
                }
            }
        ]
    }
}

```

An explanation of the schema and properties for an alert rule [is available here](#).

Resource Manager template for a resource with an alert

An alert on a Resource Manager template is most often useful when creating an alert while creating a resource. For example, you may want to ensure that a “CPU % > 80” rule is set up every time you deploy a Virtual Machine. To do this, you add the alert rule as a resource in the resource array for your VM template and add a dependency using the `dependsOn` property to the VM resource ID. Here’s a full example that creates a Windows VM and adds an alert that notifies subscription admins when the CPU utilization goes above 80%.

```
{
    "$schema": "http://schema.management.azure.com/schemas/2014-04-01-preview/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "newStorageAccountName": {
            "type": "string",
            "metadata": {
                "Description": "The name of the storage account where the VM disk is stored."
            }
        },
        "adminUsername": {
            "type": "string",
            "metadata": {
                "Description": "The name of the administrator account on the VM."
            }
        }
    }
}
```

```

        },
        "adminPassword": {
            "type": "securestring",
            "metadata": {
                "Description": "The administrator account password on the VM."
            }
        },
        "dnsNameForPublicIP": {
            "type": "string",
            "metadata": {
                "Description": "The name of the public IP address used to access the VM."
            }
        }
    },
    "variables": {
        "location": "Central US",
        "imagePublisher": "MicrosoftWindowsServer",
        "imageOffer": "WindowsServer",
        "windowsOSVersion": "2012-R2-Datacenter",
        "OSDiskName": "osdisk1",
        "nicName": "nc1",
        "addressPrefix": "10.0.0.0/16",
        "subnetName": "sn1",
        "subnetPrefix": "10.0.0.0/24",
        "storageAccountType": "Standard_LRS",
        "publicIPAddressName": "ip1",
        "publicIPAddressType": "Dynamic",
        "vmStorageAccountContainerName": "vhds",
        "vmName": "vm1",
        "vmSize": "Standard_A0",
        "virtualNetworkName": "vn1",
        "vnetID": "[resourceId('Microsoft.Network/virtualNetworks',variables('virtualNetworkName'))]",
        "subnetRef": "[concat(variables('vnetID'), '/subnets/', variables('subnetName'))]",
        "vmID": "[resourceId('Microsoft.Compute/virtualMachines',variables('vmName'))]",
        "alertName": "highCPUOnVM",
        "alertDescription": "CPU is over 80%",
        "alertisEnabled": true,
        "resourceId": "",
        "metricName": "Percentage CPU",
        "operator": "Greater Than",
        "threshold": "80",
        "windowSize": "PT5M",
        "aggregation": "Average",
        "customEmails": "",
        "sendToServiceOwners": true,
        "webhookUrl": "http://testwebhook.test"
    },
    "resources": [
        {
            "type": "Microsoft.Storage/storageAccounts",
            "name": "[parameters('newStorageAccountName')]",
            "apiVersion": "2015-06-15",
            "location": "[variables('location')]",
            "properties": {
                "accountType": "[variables('storageAccountType')]"
            }
        },
        {
            "apiVersion": "2016-03-30",
            "type": "Microsoft.Network/publicIPAddresses",
            "name": "[variables('publicIPAddressName')]",
            "location": "[variables('location')]",
            "properties": {
                "publicIPAllocationMethod": "[variables('publicIPAddressType')]",
                "dnsSettings": {
                    "domainNameLabel": "[parameters('dnsNameForPublicIP')]"
                }
            }
        }
    ],
}

```

```
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/virtualNetworks",
    "name": "[variables('virtualNetworkName')]",
    "location": "[variables('location')]",
    "properties": {
        "addressSpace": {
            "addressPrefixes": [
                "[variables('addressPrefix')]"
            ]
        },
        "subnets": [
            {
                "name": "[variables('subnetName')]",
                "properties": {
                    "addressPrefix": "[variables('subnetPrefix')]"
                }
            }
        ]
    }
},
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Network/networkInterfaces",
    "name": "[variables('nicName')]",
    "location": "[variables('location')]",
    "dependsOn": [
        "[concat('Microsoft.Network/publicIPAddresses/', variables('publicIPAddressName'))]",
        "[concat('Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'))]"
    ],
    "properties": {
        "ipConfigurations": [
            {
                "name": "ipconfig1",
                "properties": {
                    "privateIPAllocationMethod": "Dynamic",
                    "publicIPAddress": {
                        "id": "[resourceId('Microsoft.Network/publicIPAddresses', variables('publicIPAddressName'))]"
                    },
                    "subnet": {
                        "id": "[variables('subnetRef')]"
                    }
                }
            }
        ]
    }
},
{
    "apiVersion": "2016-03-30",
    "type": "Microsoft.Compute/virtualMachines",
    "name": "[variables('vmName')]",
    "location": "[variables('location')]",
    "dependsOn": [
        "[concat('Microsoft.Storage/storageAccounts/', parameters('newStorageAccountName'))]",
        "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
    ],
    "properties": {
        "hardwareProfile": {
            "vmSize": "[variables('vmSize')]"
        },
        "osProfile": {
            "computerName": "[variables('vmName')]",
            "adminUsername": "[parameters('adminUsername')]",
            "adminPassword": "[parameters('adminPassword')]"
        },
        "storageProfile": {
            "imageReference": {
                "publisher": "[variables('imagePublisher')]"
            }
        }
    }
}
```

```

        "offer": "[variables('imageOffer')]",
        "sku": "[variables('windowsOSVersion')]",
        "version": "latest"
    },
    "osDisk": {
        "name": "osdisk",
        "vhd": {
            "uri": "
[concat('http://',parameters('newStorageAccountName'),'.blob.core.windows.net/',variables('vmStorageAccountContainerName'),'/',variables('OSDiskName'),'.vhf')]"
        },
        "caching": "ReadWrite",
        "createOption": "FromImage"
    }
},
"networkProfile": {
    "networkInterfaces": [
        {
            "id": "[resourceId('Microsoft.Network/networkInterfaces',variables('nicName'))]"
        }
    ]
}
},
{
    "type": "Microsoft.Insights/alertRules",
    "name": "[variables('alertName')]",
    "dependsOn": [
        "[variables('vmID')]"
    ],
    "location": "[variables('location')]",
    "apiVersion": "2016-03-01",
    "properties": {
        "name": "[variables('alertName')]",
        "description": "variables('alertDescription')",
        "isEnabled": "[variables('alertEnabled')]",
        "condition": {
            "odata.type": "Microsoft.Azure.Management.Insights.Models.ThresholdRuleCondition",
            "dataSource": {
                "odata.type": "Microsoft.Azure.Management.Insights.Models.RuleMetricDataSource",
                "resourceUri": "[variables('vmID')]",
                "metricName": "[variables('metricName')]"
            },
            "operator": "[variables('operator')]",
            "threshold": "[variables('threshold')]",
            "windowSize": "[variables('windowSize')]",
            "timeAggregation": "[variables('aggregation')]"
        },
        "actions": [
            {
                "odata.type": "Microsoft.Azure.Management.Insights.Models.RuleEmailAction",
                "sendToServiceOwners": "[variables('sendToServiceOwners')]",
                "customEmails": "[variables('customEmails')]"
            },
            {
                "odata.type": "Microsoft.Azure.Management.Insights.Models.RuleWebhookAction",
                "serviceUri": "[variables('webhookUrl')]",
                "properties": {}
            }
        ]
    }
}
]
}
}

```

Next Steps

- [Read more about Alerts](#)
- [Add Diagnostic Settings](#) to your Resource Manager template

Best practices for Autoscale

7/18/2017 • 9 min to read • [Edit Online](#)

This article teaches best practices to autoscale in Azure. Azure Monitor autoscale applies only to [Virtual Machine Scale Sets](#), [Cloud Services](#), and [App Service - Web Apps](#). Other Azure services use different scaling methods.

Autoscale concepts

- A resource can have only *one* autoscale setting
- An autoscale setting can have one or more profiles and each profile can have one or more autoscale rules.
- An autoscale setting scales instances horizontally, which is *out* by increasing the instances and *in* by decreasing the number of instances. An autoscale setting has a maximum, minimum, and default value of instances.
- An autoscale job always reads the associated metric to scale by, checking if it has crossed the configured threshold for scale-out or scale-in. You can view a list of metrics that autoscale can scale by at [Azure Monitor autoscaling common metrics](#).
- All thresholds are calculated at an instance level. For example, "scale out by 1 instance when average CPU > 80% when instance count is 2", means scale-out when the average CPU across all instances is greater than 80%.
- You always receive failure notifications via email. Specifically, the owner, contributor, and readers of the target resource receive email. You also always receive a *recovery* email when autoscale recovers from a failure and starts functioning normally.
- You can opt-in to receive a successful scale action notification via email and webhooks.

Autoscale best practices

Use the following best practices as you use autoscale.

Ensure the maximum and minimum values are different and have an adequate margin between them

If you have a setting that has minimum=2, maximum=2 and the current instance count is 2, no scale action can occur. Keep an adequate margin between the maximum and minimum instance counts, which are inclusive. Autoscale always scales between these limits.

Manual scaling is reset by autoscale min and max

If you manually update the instance count to a value above or below the maximum, the autoscale engine automatically scales back to the minimum (if below) or the maximum (if above). For example, you set the range between 3 and 6. If you have one running instance, the autoscale engine scales to 3 instances on its next run. Likewise, it would scale-in 8 instances back to 6 on its next run. Manual scaling is very temporary unless you reset the autoscale rules as well.

Always use a scale-out and scale-in rule combination that performs an increase and decrease

If you use only one part of the combination, autoscale scale-in that single out, or in, until the maximum, or minimum, is reached.

Do not switch between the Azure portal and the Azure classic portal when managing Autoscale

For Cloud Services and App Services (Web Apps), use the Azure portal (portal.azure.com) to create and manage autoscale settings. For Virtual Machine Scale Sets use PowerShell, CLI or REST API to create and manage autoscale setting. Do not switch between the Azure classic portal (manage.windowsazure.com) and the Azure portal (portal.azure.com) when managing autoscale configurations. The Azure classic portal and its underlying backend has limitations. Move to the Azure portal to manage autoscale using a graphical user interface. The options are to use the autoscale PowerShell, CLI or REST API (via Azure Resource Explorer).

Choose the appropriate statistic for your diagnostics metric

For diagnostics metrics, you can choose among *Average*, *Minimum*, *Maximum* and *Total* as a metric to scale by. The most common statistic is *Average*.

Choose the thresholds carefully for all metric types

We recommend carefully choosing different thresholds for scale-out and scale-in based on practical situations.

We *do not recommend* autoscale settings like the examples below with the same or very similar threshold values for out and in conditions:

- Increase instances by 1 count when Thread Count \leq 600
- Decrease instances by 1 count when Thread Count \geq 600

Let's look at an example of what can lead to a behavior that may seem confusing. Consider the following sequence.

1. Assume there are 2 instances to begin with and then the average number of threads per instance grows to 625.
2. Autoscale scales out adding a 3rd instance.
3. Next, assume that the average thread count across instance falls to 575.
4. Before scaling down, autoscale tries to estimate what the final state will be if it scaled in. For example, 575×3 (current instance count) = $1,725 / 2$ (final number of instances when scaled down) = 862.5 threads. This means autoscale would have to immediately scale-out again even after it scaled in, if the average thread count remains the same or even falls only a small amount. However, if it scaled up again, the whole process would repeat, leading to an infinite loop.
5. To avoid this situation (termed "flapping"), autoscale does not scale down at all. Instead, it skips and reevaluates the condition again the next time the service's job executes. This could confuse many people because autoscale wouldn't appear to work when the average thread count was 575.

Estimation during a scale-in is intended to avoid "flapping" situations, where scale-in and scale-out actions continually go back and forth. Keep this behavior in mind when you choose the same thresholds for scale-out and in.

We recommend choosing an adequate margin between the scale-out and in thresholds. As an example, consider the following better rule combination.

- Increase instances by 1 count when CPU% \geq 80
- Decrease instances by 1 count when CPU% \leq 60

In this case

1. Assume there are 2 instances to start with.
2. If the average CPU% across instances goes to 80, autoscale scales out adding a third instance.
3. Now assume that over time the CPU% falls to 60.
4. Autoscale's scale-in rule estimates the final state if it were to scale-in. For example, 60×3 (current instance count) = $180 / 2$ (final number of instances when scaled down) = 90. So autoscale does not scale-in because it would have to scale-out again immediately. Instead, it skips scaling down.
5. The next time autoscale checks, the CPU continues to fall to 50. It estimates again - 50×3 instance = $150 / 2$ instances = 75, which is below the scale-out threshold of 80, so it scales in successfully to 2 instances.

Considerations for scaling threshold values for special metrics

For special metrics such as Storage or Service Bus Queue length metric, the threshold is the average number of messages available per current number of instances. Carefully choose the choose the threshold value for this metric.

Let's illustrate it with an example to ensure you understand the behavior better.

- Increase instances by 1 count when Storage Queue message count \geq 50

- Decrease instances by 1 count when Storage Queue message count <= 10

Consider the following sequence:

- There are 2 storage queue instances.
- Messages keep coming and when you review the storage queue, the total count reads 50. You might assume that autoscale should start a scale-out action. However, note that it is still $50/2 = 25$ messages per instance. So, scale-out does not occur. For the first scale-out to happen, the total message count in the storage queue should be 100.
- Next, assume that the total message count reaches 100.
- A 3rd storage queue instance is added due to a scale-out action. The next scale-out action will not happen until the total message count in the queue reaches 150 because $150/3 = 50$.
- Now the number of messages in the queue gets smaller. With 3 instances, the first scale-in action happens when the total messages in all queues add up to 30 because $30/3 = 10$ messages per instance, which is the scale-in threshold.

Considerations for scaling when multiple profiles are configured in an autoscale setting

In an autoscale setting, you can choose a default profile, which is always applied without any dependency on schedule or time, or you can choose a recurring profile or a profile for a fixed period with a date and time range.

When autoscale service processes them, it always checks in the following order:

- Fixed Date profile
- Recurring profile
- Default ("Always") profile

If a profile condition is met, autoscale does not check the next profile condition below it. Autoscale only processes one profile at a time. This means if you want to also include a processing condition from a lower-tier profile, you must include those rules as well in the current profile.

Let's review this using an example:

The image below shows an autoscale setting with a default profile of minimum instances = 2 and maximum instances = 10. In this example, rules are configured to scale-out when the message count in the queue is greater than 10 and scale-in when the message count in the queue is less than 3. So now the resource can scale between 2 and 10 instances.

In addition, there is a recurring profile set for Monday. It is set for minimum instances = 2 and maximum instances = 12. This means on Monday, the first time autoscale checks for this condition, if the instance count is 2, it scales to the new minimum of 3. As long as autoscale continues to find this profile condition matched (Monday), it only processes the CPU-based scale-out/in rules configured for this profile. At this time, it does not check for the queue length. However, if you also want the queue length condition to be checked, you should include those rules from the default profile as well in your Monday profile.

Similarly, when autoscale switches back to the default profile, it first checks if the minimum and maximum conditions are met. If the number of instances at the time is 12, it scales in to 10, the maximum allowed for the default profile.

Autoscale setting
rrtest123241/Production/WorkerRole1 (Cloud services deployment slot role)

Save Discard Disable autoscale

Configure Run history JSON Notify

Autoscale setting name rrtest123241-Production-WorkerRole1-rrtest123241

Resource group rrtest123241

Instance count 2

Default rrtest123241-Production-WorkerRole1-rrtest123241

Scale mode Scale based on a metric Scale to a specific instance count

Rules

| Scale out | | | |
|-----------|---------------|-------------------|------------------------------|
| When | approvalqueue | MessageCount > 10 | Increase instance count by 1 |
| Scale in | | | |
| When | approvalqueue | MessageCount < 3 | Decrease instance count by 1 |

[+ Add a rule](#)

Instance limits Minimum Maximum Default

Schedule This scale condition is executed when none of the other scale condition(s) match

Auto created scale condition

Scale mode Scale based on a metric Scale to a specific instance count

Rules

| Scale out | | | |
|-----------|-------------|------------------------------|------------------------------|
| When | WorkerRole1 | (Average) Percentage CPU > 5 | Increase instance count by 1 |
| Scale in | | | |
| When | WorkerRole1 | (Average) Percentage CPU < 2 | Decrease instance count by 1 |

[+ Add a rule](#)

Instance limits Minimum Maximum Default

Schedule Specify start/end dates Repeat specific days

Repeat every Monday Tuesday Wednesday Thursday Friday
 Saturday Sunday

Timezone

Start time

End time

Considerations for scaling when multiple rules are configured in a profile

There are cases where you may have to set multiple rules in a profile. The following set of autoscale rules are used by services use when multiple rules are set.

On *scale out*, autoscale runs if any rule is met. On *scale-in*, autoscale require all rules to be met.

To illustrate, assume that you have the following 4 autoscale rules:

- If CPU < 30 %, scale-in by 1

- If Memory < 50%, scale-in by 1
- If CPU > 75%, scale-out by 1
- If Memory > 75%, scale-out by 1

Then the follow occurs:

- If CPU is 76% and Memory is 50%, we scale-out.
- If CPU is 50% and Memory is 76% we scale-out.

On the other hand, if CPU is 25% and memory is 51% autoscale does **not** scale-in. In order to scale-in, CPU must be 29% and Memory 49%.

Always select a safe default instance count

The default instance count is important autoscale scales your service to that count when metrics are not available. Therefore, select a default instance count that's safe for your workloads.

Configure autoscale notifications

Autoscale notifies the administrators and contributors of the resource by email if any of the following conditions occur:

- autoscale service fails to take an action.
- Metrics are not available for autoscale service to make a scale decision.
- Metrics are available (recovery) again to make a scale decision. In addition to the conditions above, you can configure email or webhook notifications to get notified for successful scale actions.

You can also use an Activity Log alert to monitor the health of the autoscale engine. Here are examples to [create an Activity Log Alert to monitor all autoscale engine operations on your subscription](#) or to [create an Activity Log Alert to monitor all failed autoscale scale in/scale out operations on your subscription](#).

Next Steps

- [Create an Activity Log Alert to monitor all autoscale engine operations on your subscription](#).
- [Create an Activity Log Alert to monitor all failed autoscale scale in/scale out operations on your subscription](#)

Azure Monitor autoscaling common metrics

7/18/2017 • 5 min to read • [Edit Online](#)

Azure Monitor autoscaling allows you to scale the number of running instances up or down, based on telemetry data (metrics). This document describes common metrics that you might want to use. In the Azure portal for Cloud Services and Server Farms, you can choose the metric of the resource to scale by. However, you can also choose any metric from a different resource to scale by.

Azure Monitor autoscale applies only to [Virtual Machine Scale Sets](#), [Cloud Services](#), and [App Service - Web Apps](#). Other Azure services use different scaling methods.

Compute metrics for Resource Manager-based VMs

By default, Resource Manager-based Virtual Machines and Virtual Machine Scale Sets emit basic (host-level) metrics. In addition, when you configure diagnostics data collection for an Azure VM and VMSS, the Azure diagnostic extension also emits guest-OS performance counters (commonly known as "guest-OS metrics"). You use all these metrics in autoscale rules.

You can use the [Get MetricDefinitions](#) API/PoSH/CLI to view the metrics available for your VMSS resource.

If you're using VM scale sets and you don't see a particular metric listed, then it is likely *disabled* in your diagnostics extension.

If a particular metric is not being sampled or transferred at the frequency you want, you can update the diagnostics configuration.

If either preceding case is true, then review [Use PowerShell to enable Azure Diagnostics in a virtual machine running Windows](#) about PowerShell to configure and update your Azure VM Diagnostics extension to enable the metric. That article also includes a sample diagnostics configuration file.

Host metrics for Resource Manager-based Windows and Linux VMs

The following host-level metrics are emitted by default for Azure VM and VMSS in both Windows and Linux instances. These metrics describe your Azure VM, but are collected from the Azure VM host rather than via agent installed on the guest VM. You may use these metrics in autoscaling rules.

- [Host metrics for Resource Manager-based Windows and Linux VMs](#)
- [Host metrics for Resource Manager-based Windows and Linux VM Scale Sets](#)

Guest OS metrics Resource Manager-based Windows VMs

When you create a VM in Azure, diagnostics is enabled by using the Diagnostics extension. The diagnostics extension emits a set of metrics taken from inside of the VM. This means you can autoscale off of metrics that are not emitted by default.

You can generate a list of the metrics by using the following command in PowerShell.

```
Get-AzureRmMetricDefinition -ResourceId <resource_id> | Format-Table -Property Name,Unit
```

You can create an alert for the following metrics:

| METRIC NAME | UNIT |
|--|----------------|
| \Processor(_Total)\% Processor Time | Percent |
| \Processor(_Total)\% Privileged Time | Percent |
| \Processor(_Total)\% User Time | Percent |
| \Processor Information(_Total)\Processor Frequency | Count |
| \System\Processes | Count |
| \Process(_Total)\Thread Count | Count |
| \Process(_Total)\Handle Count | Count |
| \Memory\% Committed Bytes In Use | Percent |
| \Memory\Available Bytes | Bytes |
| \Memory\Committed Bytes | Bytes |
| \Memory\Commit Limit | Bytes |
| \Memory\Pool Paged Bytes | Bytes |
| \Memory\Pool Nonpaged Bytes | Bytes |
| \PhysicalDisk(_Total)\% Disk Time | Percent |
| \PhysicalDisk(_Total)\% Disk Read Time | Percent |
| \PhysicalDisk(_Total)\% Disk Write Time | Percent |
| \PhysicalDisk(_Total)\Disk Transfers/sec | CountPerSecond |
| \PhysicalDisk(_Total)\Disk Reads/sec | CountPerSecond |
| \PhysicalDisk(_Total)\Disk Writes/sec | CountPerSecond |
| \PhysicalDisk(_Total)\Disk Bytes/sec | BytesPerSecond |
| \PhysicalDisk(_Total)\Disk Read Bytes/sec | BytesPerSecond |
| \PhysicalDisk(_Total)\Disk Write Bytes/sec | BytesPerSecond |
| \PhysicalDisk(_Total)\Avg. Disk Queue Length | Count |
| \PhysicalDisk(_Total)\Avg. Disk Read Queue Length | Count |
| \PhysicalDisk(_Total)\Avg. Disk Write Queue Length | Count |

| METRIC NAME | UNIT |
|-------------------------------------|---------|
| \LogicalDisk(_Total)\% Free Space | Percent |
| \LogicalDisk(_Total)\Free Megabytes | Count |

Guest OS metrics Linux VMs

When you create a VM in Azure, diagnostics is enabled by default by using Diagnostics extension.

You can generate a list of the metrics by using the following command in PowerShell.

```
Get-AzureRmMetricDefinition -ResourceId <resource_id> | Format-Table -Property Name,Unit
```

You can create an alert for the following metrics:

| METRIC NAME | UNIT |
|----------------------------------|----------------|
| \Memory\AvailableMemory | Bytes |
| \Memory\PercentAvailableMemory | Percent |
| \Memory\UsedMemory | Bytes |
| \Memory\PercentUsedMemory | Percent |
| \Memory\PercentUsedByCache | Percent |
| \Memory\PagesPerSec | CountPerSecond |
| \Memory\PagesReadPerSec | CountPerSecond |
| \Memory\PagesWrittenPerSec | CountPerSecond |
| \Memory\AvailableSwap | Bytes |
| \Memory\PercentAvailableSwap | Percent |
| \Memory\UsedSwap | Bytes |
| \Memory\PercentUsedSwap | Percent |
| \Processor\PercentIdleTime | Percent |
| \Processor\PercentUserTime | Percent |
| \Processor\PercentNiceTime | Percent |
| \Processor\PercentPrivilegedTime | Percent |
| \Processor\PercentInterruptTime | Percent |
| \Processor\PercentDPCTime | Percent |

| METRIC NAME | UNIT |
|--------------------------------------|----------------|
| \Processor\PercentProcessorTime | Percent |
| \Processor\PercentIOWaitTime | Percent |
| \PhysicalDisk\BytesPerSecond | BytesPerSecond |
| \PhysicalDisk\ReadBytesPerSecond | BytesPerSecond |
| \PhysicalDisk\WriteBytesPerSecond | BytesPerSecond |
| \PhysicalDisk\TransfersPerSecond | CountPerSecond |
| \PhysicalDisk\ReadsPerSecond | CountPerSecond |
| \PhysicalDisk\WritesPerSecond | CountPerSecond |
| \PhysicalDisk\AverageReadTime | Seconds |
| \PhysicalDisk\AverageWriteTime | Seconds |
| \PhysicalDisk\AverageTransferTime | Seconds |
| \PhysicalDisk\AverageDiskQueueLength | Count |
| \NetworkInterface\BytesTransmitted | Bytes |
| \NetworkInterface\BytesReceived | Bytes |
| \NetworkInterface\PacketsTransmitted | Count |
| \NetworkInterface\PacketsReceived | Count |
| \NetworkInterface\BytesTotal | Bytes |
| \NetworkInterface>TotalRxErrors | Count |
| \NetworkInterface>TotalTxErrors | Count |
| \NetworkInterface\TotalCollisions | Count |

Commonly used Web (Server Farm) metrics

You can also perform autoscale based on common web server metrics such as the Http queue length. Its metric name is **HttpQueueLength**. The following section lists available server farm (Web Apps) metrics.

Web Apps metrics

You can generate a list of the Web Apps metrics by using the following command in PowerShell.

```
Get-AzureRmMetricDefinition -ResourceId <resource_id> | Format-Table -Property Name,Unit
```

You can alert on or scale by these metrics.

| METRIC NAME | UNIT |
|------------------|---------|
| CpuPercentage | Percent |
| MemoryPercentage | Percent |
| DiskQueueLength | Count |
| HttpQueueLength | Count |
| BytesReceived | Bytes |
| BytesSent | Bytes |

Commonly used Storage metrics

You can scale by Storage queue length, which is the number of messages in the storage queue. Storage queue length is a special metric and the threshold is the number of messages per instance. For example, if there are two instances and if the threshold is set to 100, scaling occurs when the total number of messages in the queue is 200. That can be 100 messages per instance, 120 and 80, or any other combination that adds up to 200 or more.

Configure this setting in the Azure portal in the **Settings** blade. For VM scale sets, you can update the Autoscale setting in the Resource Manager template to use *metricName* as *ApproximateMessageCount* and pass the ID of the storage queue as *metricResourceUri*.

For example, with a Classic Storage Account the autoscale setting metricTrigger would include:

```
"metricName": "ApproximateMessageCount",
"metricNamespace": "",
"metricResourceUri":
"/subscriptions/SUBSCRIPTION_ID/resourceGroups/RES_GROUP_NAME/providers/Microsoft.ClassicStorage/storageAccounts/STORAGE_ACCOUNT_NAME/services/queue/queues/QUEUE_NAME"
```

For a (non-classic) storage account, the metricTrigger would include:

```
"metricName": "ApproximateMessageCount",
"metricNamespace": "",
"metricResourceUri":
"/subscriptions/SUBSCRIPTION_ID/resourceGroups/RES_GROUP_NAME/providers/Microsoft.Storage/storageAccounts/STORAGE_ACCOUNT_NAME/services/queue/queues/QUEUE_NAME"
```

Commonly used Service Bus metrics

You can scale by Service Bus queue length, which is the number of messages in the Service Bus queue. Service Bus queue length is a special metric and the threshold is the number of messages per instance. For example, if there are two instances and if the threshold is set to 100, scaling occurs when the total number of messages in the queue is 200. That can be 100 messages per instance, 120 and 80, or any other combination that adds up to 200 or more.

For VM scale sets, you can update the Autoscale setting in the Resource Manager template to use *metricName* as *ApproximateMessageCount* and pass the ID of the storage queue as *metricResourceUri*.

```
"metricName": "MessageCount",
"metricNamespace": "",
"metricResourceUri":
"/subscriptions/SUBSCRIPTION_ID/resourceGroups/RES_GROUP_NAME/providers/Microsoft.ServiceBus/namespaces/SB_NAM
ESPACE/queues/QUEUE_NAME"
```

NOTE

For Service Bus, the resource group concept does not exist but Azure Resource Manager creates a default resource group per region. The resource group is usually in the 'Default-ServiceBus-[region]' format. For example, 'Default-ServiceBus-EastUS', 'Default-ServiceBus-WestUS', 'Default-ServiceBus-AustraliaEast' etc.

Overview of common autoscale patterns

7/18/2017 • 1 min to read • [Edit Online](#)

This article describes some of the common patterns to scale your resource in Azure.

Azure Monitor auto scale applies only to Virtual Machine Scale Sets (VMSS), cloud services, app service plans and app service environments.

Lets get started

This article assumes that you are familiar with auto scale. You can [get started here to scale your resource](#). The following are some of the common scale patterns.

Scale based on CPU

You have a web app (/VMSS/cloud service role) and

- You want to scale out/scale in based on CPU.
- Additionally, you want to ensure there is a minimum number of instances.
- Also, you want to ensure that you set a maximum limit to the number of instances you can scale to.

Microsoft Azure Monitor - Autoscale > Autoscale setting

Autoscale setting
demovmss (Virtual machine scale set)

Save Discard Disable autoscale

Configure Run history JSON Notify

Autoscale setting name: cpuautoscale
Resource group: demovmss
Instance count: 1

Default Profile1

Scale mode: Scale based on a metric Scale to a specific instance count

Rules

Scale out: When demovmss (Average) Percentage CPU > 75 Increase instance count by 1

Scale in: When demovmss (Average) Percentage CPU < 25 Decrease instance count by 1

+ Add a rule

Instance limits: Minimum 1, Maximum 10, Default 1

Schedule: This scale condition is executed when none of the other scale condition(s) match

+ Add a scale condition

Scale differently on weekdays vs weekends

You have a web app (/VMSS/cloud service role) and

- You want 3 instances by default (on weekdays)
- You don't expect traffic on weekends and hence you want to scale down to 1 instance on weekends.

Autoscale setting
WebdayTrafficAsp (App Service plan)

Save Discard Disable autoscale

Configure Run history JSON Notify

Autoscale setting name: WeekdayTrafficAsp
Resource group: autoscaledemo
Instance count: 1

Default Auto created scale condition

Scale mode: Scale based on a metric Scale to a specific instance count
Instance count: 3

Schedule: This scale condition is executed when none of the other scale condition(s) match

WeekendTraffic

Scale mode: Scale based on a metric Scale to a specific instance count
Instance count: 1

Schedule: Specify start/end dates Repeat specific days
Repeat every: Monday, Tuesday, Wednesday, Thursday, Friday
Saturday, Sunday

Timezone: (UTC-08:00) Pacific Time (US & Canada)
Start time: 12:00
End time: 11:59

+ Add a scale condition

<https://portal.azure.com/?feature.customportal=false#>

Scale differently during holidays

You have a web app (/VMSS/cloud service role) and

- You want to scale up/down based on CPU usage by default
- However, during holiday season (or specific days that are important for your business) you want to override the defaults and have more capacity at your disposal.

Autoscale setting
HolidaySpikeAsp (App Service plan)

Save Discard Disable autoscale

Configure Run history JSON Notify

Autoscale setting name: HolidaySpikeAsp
Resource group: autoscaledemo
Instance count: 2

Default NormalScale

Scale mode: Scale based on a metric Scale to a specific instance count

Rules

| When | Condition | Action |
|-----------|--|------------------------------|
| Scale out | HolidaySpikeAsp (Average) CpuPercentage > 70 | Increase instance count by 2 |
| Scale in | HolidaySpikeAsp (Average) CpuPercentage < 30 | Decrease instance count by 1 |

+ Add a rule

Instance limits: Minimum 2, Maximum 5, Default 2

Schedule: This scale condition is executed when none of the other scale condition(s) match

HolidayScale

Scale mode: Scale based on a metric Scale to a specific instance count
Instance count: 8

Schedule: Specify start/end dates Repeat specific days
Timezone: (UTC-08:00) Pacific Time (US & Canada)
Start date: 2017-11-30 23:00
End date: 2017-12-31 22:59:00

Scale based on custom metric

You have a web front end and a API tier that communicates with the backend.

- You want to scale the API tier based on custom events in the front end (example: You want to scale your checkout process based on the number of items in the shopping cart)

Microsoft Azure Monitor - Autoscale > Autoscale setting

Autoscale setting
contoso-web-api-app (App Service plan)

Save Discard Disable autoscale

Configure Run history JSON Notify

Autoscale setting name: Web api autoscale

Resource group: contoso-web

Instance count: 1

Default Auto created scale condition

Scale mode: Scale based on a metric Scale to a specific instance count

Rules

Scale out

| When | loans-app-ai | (Total) customMetrics/LoanSubmissions > 100 | Increase instance count by 1 |
|------|--------------|---|------------------------------|
|------|--------------|---|------------------------------|

Scale in

| When | loans-app-ai | (Total) customMetrics/LoanSubmissions < 25 | Decrease instance count by 1 |
|------|--------------|--|------------------------------|
|------|--------------|--|------------------------------|

+ Add a rule

Instance limits: Minimum 2, Maximum 5, Default 2

Schedule: This scale condition is executed when none of the other scale condition(s) match

+ Add a scale condition

The screenshot shows the Azure portal interface for managing an App Service plan's autoscale settings. The main pane displays basic information like the setting name, resource group, and current instance count. Below this, the 'Default' scale condition is detailed, showing two rules: one to increase instances if loan submissions exceed 100, and another to decrease instances if submissions drop below 25. The 'Schedule' section indicates this condition runs when no other conditions match. A sidebar on the left provides navigation links for various Azure services.

Get started with auto scale by custom metric in Azure

7/18/2017 • 1 min to read • [Edit Online](#)

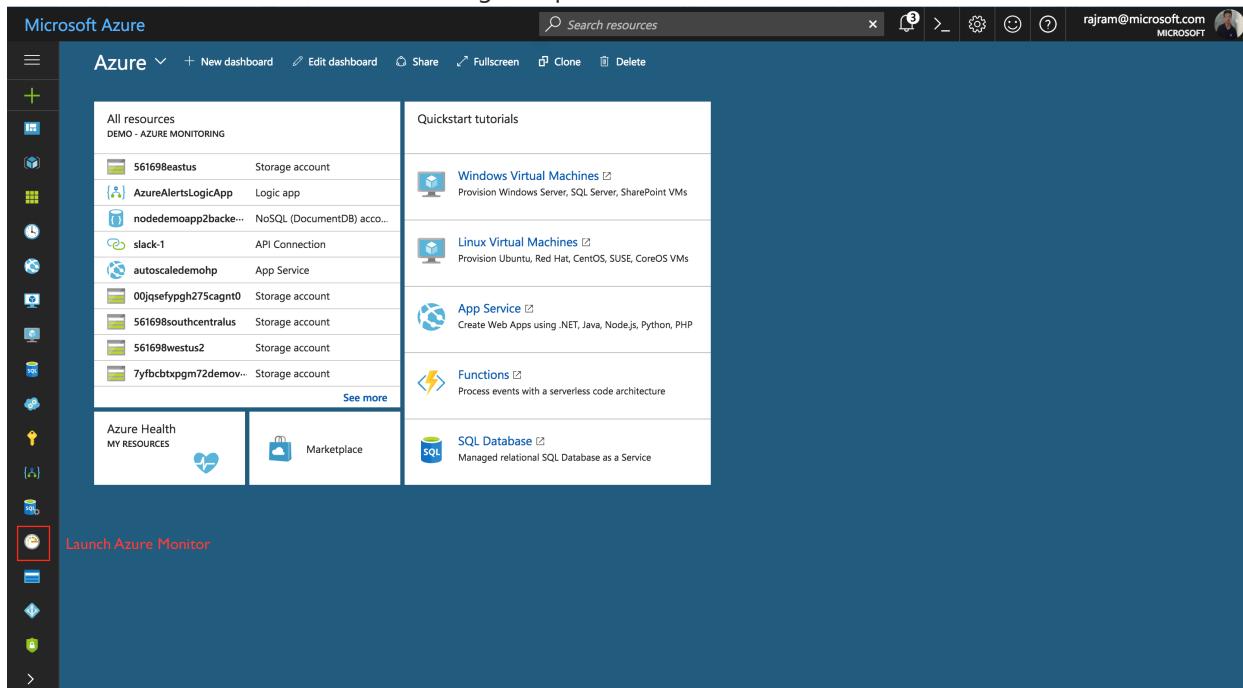
This article describes how to scale your resource by a custom metric in Azure portal.

Azure Monitor auto scale applies only to Virtual Machine Scale Sets (VMSS), cloud services, app service plans and app service environments.

Lets get started

This article assumes that you have a web app with application insights configured. If you don't have one already, you can [set up Application Insights for your ASP.NET website](#)

- Open [Azure portal](#)
- Click on Azure Monitor icon in the left navigation pane.



- Click on Autoscale setting to view all the resources for which auto scale is applicable, along with its current autoscale status

The screenshot shows the Microsoft Azure Monitor - Autoscale blade. On the left, there's a navigation sidebar with sections like EXPLORE, MANAGE, and ADVANCED. Under EXPLORE, 'Autoscale' is selected. The main area displays a table of resources:

| NAME | RESOURCE TYPE | RESOURCE GROUP | LOCATION | INSTANCE COUNT | AUTOSCALE STATUS |
|------------------------|---------------------------|----------------|------------------|----------------|------------------|
| WebWorkerDemo | Cloud service (classic) | autoscaledemo | Southeast Asia | | Not configured |
| Production/WebRole1 | Role | autoscaledemo | | 1 | Not configured |
| Production/WorkerRo... | Role | autoscaledemo | | 1 | Not configured |
| demovmss | Virtual machine scale set | demovmss | West US 2 | 1 | Enabled |
| CPUBasedScaleAsp | App Service plan | autoscaledemo | West US 2 | 2 | Enabled |
| HolidaySpikeAsp | App Service plan | autoscaledemo | West US 2 | 2 | Enabled |
| staticscaleasp | App Service plan | autoscaledemo | West US 2 | 2 | Enabled |
| WeekdayTrafficAsp | App Service plan | autoscaledemo | West US 2 | 1 | Enabled |
| BrazilSouthPlan | App Service plan | contoso-common | Brazil South | 0 | Not configured |
| CanadaCentralPlan | App Service plan | contoso-common | Canada Central | 0 | Not configured |
| SouthCentralUSPlan | App Service plan | contoso-common | South Central US | 0 | Not configured |
| WestUS2Plan | App Service plan | contoso-common | West US 2 | 0 | Not configured |
| contoso-mvc-app-asp | App Service plan | contoso-web | West US 2 | 10 | Enabled |
| contoso-web-api-asp | App Service plan | contoso-web | West US | 5 | Enabled |
| contoso-web-react-a... | App Service plan | contoso-web | West Europe | 1 | Not configured |
| nodeappn2sn | App Service plan | nodedemom | West US | 1 | Not configured |

- Open 'Autoscale' blade in Azure Monitor and select a resource you want to scale > Note: The steps below use an app service plan associated with a web app that has app insights configured.
- In the scale setting blade for the resource, notice that the current instance count is 1. Click on 'Enable autoscale'.

The screenshot shows the 'Autoscale setting' blade for the 'contoso-web-api-asp' App Service plan. It includes tabs for 'Configure', 'Run history', 'JSON', and 'Notify'. The 'Configure' tab is active. The 'Override condition' section shows an 'Instance count' slider set to 1. Below it, a message says 'Your autoscale configuration is disabled. To reinstate your configuration, enable autoscale.' A prominent blue 'Enable autoscale' button is at the bottom.

- Provide a name for the scale setting, and click on "Add a rule". Notice the scale rule options that open as a context pane in the right hand side. By default, it sets the option to scale your instance count by 1 if the CPU percentage of the resource exceeds 70%. Change the metric source at the top to "Application Insights", select the app insights resource in the 'Resource' dropdown and then select the custom metric based on which you want to scale.

The screenshot shows the 'Autoscale setting' blade for the 'contoso-web-api-asp' App Service plan. The 'Scale rule' context pane is open on the right. It includes fields for 'Metric source' (set to 'Application Insights'), 'Resource type' (set to 'Application Insights'), 'Resource' (set to 'loans-app-ai'), and 'Criteria' (which are detailed below). The 'Criteria' section contains settings for 'Time aggregation' (set to 'Total'), 'Metric name' (set to 'LoanSubmissions'), 'Time grain (in mins)' (set to '1'), 'Operator' (set to 'Greater than'), 'Threshold' (set to '100'), and 'Duration (in minutes)' (set to '10'). At the bottom of the pane is an 'Action' section with a 'Save' button.

- Similar to the step above, add a scale rule that will scale in and decrease the scale count by 1 if the custom metric is below a threshold.

Autoscale setting name: Web api autoscale
Resource group: contoso-web
Instance count: 1

Default Auto created scale condition

Scale mode: Scale based on a metric Scale to a specific instance count

Rules

| When | loans-app-ai | (Total) customMetrics/LoanSubmissions > 100 | Increase instance count by 1 |
|------|--------------|---|------------------------------|
| When | loans-app-ai | (Total) customMetrics/LoanSubmissions < 25 | Decrease instance count by 1 |

Instance limits: Minimum 2, Maximum 5, Default 2

Schedule: This scale condition is executed when none of the other scale condition(s) match

- Set the instance limits. For example, if you want to scale between 2-5 instances depending on the custom metric fluctuations, set 'minimum' to '2', 'maximum' to '5' and 'default' to '2' > Note: In case there is a problem reading the resource metrics and the current capacity is below the default capacity, then to ensure the availability of the resource, Autoscale will scale out to the default value. If the current capacity is already higher than default capacity, Autoscale will not scale in.
- Click on 'Save'

Congratulations. You now successfully created your scale setting to auto scale your web app based on a custom metric.

Note: The same steps are applicable to get started with a VMSS or cloud service role.

Advanced autoscale configuration using Resource Manager templates for VM Scale Sets

7/18/2017 • 5 min to read • [Edit Online](#)

You can scale-in and scale-out in Virtual Machine Scale Sets based on performance metric thresholds, by a recurring schedule, or by a particular date. You can also configure email and webhook notifications for scale actions. This walkthrough shows an example of configuring all these objects using a Resource Manager template on a VM Scale Set.

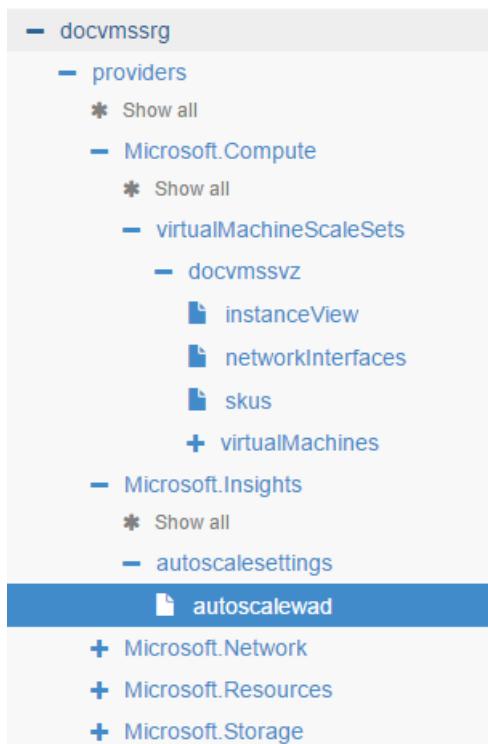
NOTE

While this walkthrough explains the steps for VM Scale Sets, the same information applies to autoscaling [Cloud Services](#), and [App Service - Web Apps](#). For a simple scale in/out setting on a VM Scale Set based on a simple performance metric such as CPU, refer to the [Linux](#) and [Windows](#) documents

Walkthrough

In this walkthrough, we use [Azure Resource Explorer](#) to configure and update the autoscale setting for a scale set. Azure Resource Explorer is an easy way to manage Azure resources via Resource Manager templates. If you are new to Azure Resource Explorer tool, read [this introduction](#).

1. Deploy a new scale set with a basic autoscale setting. This article uses the one from the Azure QuickStart Gallery, which has a Windows scale set with a basic autoscale template. Linux scale sets work the same way.
2. After the scale set is created, navigate to the scale set resource from Azure Resource Explorer. You see the following under Microsoft.Insights node.



The template execution has created a default autoscale setting with the name '**autoscalewad**'. On the right-hand side, you can view the full definition of this autoscale setting. In this case, the default autoscale setting comes with a CPU% based scale-out and scale-in rule.

3. You can now add more profiles and rules based on the schedule or specific requirements. We create an autoscale setting with three profiles. To understand profiles and rules in autoscale, review [Autoscale Best Practices](#).

| PROFILES & RULES | DESCRIPTION |
|------------------|---|
| Profile | Performance/metric based |
| Rule | Service Bus Queue Message Count > x |
| Rule | Service Bus Queue Message Count < y |
| Rule | CPU% > n |
| Rule | CPU% < p |
| Profile | Weekday morning hours (no rules) |
| Profile | Product Launch day (no rules) |

4. Here is a hypothetical scaling scenario that we use for this walk-through.

- **Load based** - I'd like to scale out or in based on the load on my application hosted on my scale set.*
- **Message Queue size** - I use a Service Bus Queue for the incoming messages to my application. I use the queue's message count and CPU% and configure a default profile to trigger a scale action if either of message count or CPU hits the threshold.*
- **Time of week and day** - I want a weekly recurring 'time of the day' based profile called 'Weekday Morning Hours'. Based on historical data, I know it is better to have certain number of VM instances to handle my application's load during this time.*
- **Special Dates** - I added a 'Product Launch Day' profile. I plan ahead for specific dates so my application is ready to handle the load due marketing announcements and when we put a new product in the application.*
- *The last two profiles can also have other performance metric based rules within them. In this case, I decided not to have one and instead to rely on the default performance metric based rules. Rules are optional for the recurring and date-based profiles.*

Autoscale engine's prioritization of the profiles and rules is also captured in the [autoscaling best practices](#) article. For a list of common metrics for autoscale, refer [Common metrics for Autoscale](#)

5. Make sure you are on the **Read/Write** mode in Resource Explorer

6. Click Edit. Replace the 'profiles' element in autoscale setting with the following configuration:

```
"profiles": [
  {
    "name": "Perf_Based_Scale",
    "capacity": {
      "minimum": "2",
      "maximum": "12",
      "default": "2"
    },
    "rules": [
      {
        "metricTrigger": {
          "metricName": "MessageCount",
          "metricNamespace": "",
          "metricResourceUri": "/subscriptions/s1/resourceGroups/rg1/providers/Microsoft.ServiceBus/namespaces/mySB/queues/myqueue",
          "timeGrain": "PT5M",
          "statistic": "Average",
          "timeWindow": "PT5M",
          "timeAggregation": "Average",
          "operator": "GreaterThan",
          "threshold": 10
        },
        "scaleAction": {
          "direction": "Increase",
          "type": "ChangeCount",
          "value": "1",
          "cooldown": "PT5M"
        }
      },
      {
        "metricTrigger": {
          "metricName": "MessageCount",
          "metricNamespace": "",
          "metricResourceUri": "/subscriptions/s1/resourceGroups/rg1/providers/Microsoft.ServiceBus/namespaces/mySB/queues/myqueue",
          "timeGrain": "PT5M",
          "statistic": "Average",
          "timeWindow": "PT5M",
          "timeAggregation": "Average",
          "operator": "LessThan",
          "threshold": 3
        },
        "scaleAction": {
          "direction": "Decrease",
          "type": "ChangeCount",
          "value": "1",
          "cooldown": "PT5M"
        }
      },
      {
        "metricTrigger": {
          "metricName": "Percentage CPU",
          "metricNamespace": "",
          "metricResourceUri": "/subscriptions/s1/resourceGroups/rg1/providers/Microsoft.Compute/virtualMachineScaleSets/<this_vmss_name>",
          "timeGrain": "PT5M",
          "statistic": "Average",
          "timeWindow": "PT30M",
          "timeAggregation": "Average",
          "operator": "GreaterThan",
          "threshold": 85
        },
        "scaleAction": {
          "direction": "Increase",
          "type": "ChangeCount"
        }
      }
    ]
  }
]
```

```

        "type": "ChangeCount",
        "value": "1",
        "cooldown": "PT5M"
    }
},
{
    "metricTrigger": {
        "metricName": "Percentage CPU",
        "metricNamespace": "",
        "metricResourceUri":
        "/subscriptions/s1/resourceGroups/rg1/providers/Microsoft.Compute/virtualMachineScaleSets/<this_vmss_name>",
        "timeGrain": "PT5M",
        "statistic": "Average",
        "timeWindow": "PT30M",
        "timeAggregation": "Average",
        "operator": "LessThan",
        "threshold": 60
    },
    "scaleAction": {
        "direction": "Decrease",
        "type": "ChangeCount",
        "value": "1",
        "cooldown": "PT5M"
    }
}
],
},
{
    "name": "Weekday_Morning_Hours_Scale",
    "capacity": {
        "minimum": "4",
        "maximum": "12",
        "default": "4"
    },
    "rules": [],
    "recurrence": {
        "frequency": "Week",
        "schedule": {
            "timeZone": "Pacific Standard Time",
            "days": [
                "Monday",
                "Tuesday",
                "Wednesday",
                "Thursday",
                "Friday"
            ],
            "hours": [
                6
            ],
            "minutes": [
                0
            ]
        }
    }
},
{
    "name": "Product_Launch_Day",
    "capacity": {
        "minimum": "6",
        "maximum": "20",
        "default": "6"
    },
    "rules": [],
    "fixedDate": {
        "timeZone": "Pacific Standard Time",
        "start": "2016-06-20T00:06:00Z",
        "end": "2016-06-21T23:59:00Z"
    }
}
]
}

```

For supported fields and their values, see [Autoscale REST API documentation](#). Now your autoscale setting contains the three profiles explained previously.

- Finally, look at the Autoscale **notification** section. Autoscale notifications allow you to do three things when a scale-out or in action is successfully triggered.

- Notify the admin and co-admins of your subscription
- Email a set of users
- Trigger a webhook call. When fired, this webhook sends metadata about the autoscaling condition and the scale set resource. To learn more about the payload of autoscale webhook, see [Configure Webhook & Email Notifications for Autoscale](#).

Add the following to the Autoscale setting replacing your **notification** element whose value is null

```
"notifications": [
  {
    "operation": "Scale",
    "email": {
      "sendToSubscriptionAdministrator": true,
      "sendToSubscriptionCoAdministrators": false,
      "customEmails": [
        "user1@mycompany.com",
        "user2@mycompany.com"
      ]
    },
    "webhooks": [
      {
        "serviceUri": "https://foo.webhook.example.com?token=abcd1234",
        "properties": {
          "optional_key1": "optional_value1",
          "optional_key2": "optional_value2"
        }
      }
    ]
  }
]
```

Hit **Put** button in Resource Explorer to update the autoscale setting.

You have updated an autoscale setting on a VM Scale set to include multiple scale profiles and scale notifications.

Next Steps

Use these links to learn more about autoscaling.

[Troubleshoot Autoscale with Virtual Machine Scale Sets](#)

[Common Metrics for Autoscale](#)

[Best Practices for Azure Autoscale](#)

[Manage Autoscale using PowerShell](#)

[Manage Autoscale using CLI](#)

[Configure Webhook & Email Notifications for Autoscale](#)

Automatically scale machines in a virtual machine scale set

6/27/2017 • 11 min to read • [Edit Online](#)

Virtual machine scale sets make it easy for you to deploy and manage identical virtual machines as a set. Scale sets provide a highly scalable and customizable compute layer for hyperscale applications, and they support Windows platform images, Linux platform images, custom images, and extensions. For more information about scale sets, see [Virtual Machine Scale Sets](#).

This tutorial shows you how to create a scale set of Windows virtual machines and automatically scale the machines in the set. You create the scale set and set up scaling by creating an Azure Resource Manager template and deploying it using Azure PowerShell. For more information about templates, see [Authoring Azure Resource Manager templates](#). To learn more about automatic scaling of scale sets, see [Automatic scaling and Virtual Machine Scale Sets](#).

In this article, you deploy the following resources and extensions:

- Microsoft.Storage/storageAccounts
- Microsoft.Network/virtualNetworks
- Microsoft.Network/publicIPAddresses
- Microsoft.Network/loadBalancers
- Microsoft.Network/networkInterfaces
- Microsoft.Compute/virtualMachines
- Microsoft.Compute/virtualMachineScaleSets
- Microsoft.Insights.VMDiagnosticsSettings
- Microsoft.Insights/autoscaleSettings

For more information about Resource Manager resources, see [Azure Resource Manager vs. classic deployment](#).

Step 1: Install Azure PowerShell

See [How to install and configure Azure PowerShell](#) for information about installing the latest version of Azure PowerShell, selecting your subscription, and signing in to Azure.

Step 2: Create a resource group and a storage account

1. **Create a resource group** – All resources must be deployed to a resource group. Use [New-AzureRmResourceGroup](#) to create a resource group named **vmsstestrg1**.
2. **Create a storage account** – This storage account is where the template is stored. Use [New-AzureRmStorageAccount](#) to create a storage account named **vmsstestsa**.

Step 3: Create the template

An Azure Resource Manager template makes it possible for you to deploy and manage Azure resources together by using a JSON description of the resources and associated deployment parameters.

1. In your favorite editor, create the file C:\VMSSTemplate.json and add the initial JSON structure to support the template.

```
{  
    "$schema": "http://schema.management.azure.com/schemas/2014-04-01-preview/VM.json",  
    "contentVersion": "1.0.0.0",  
    "parameters": {  
    },  
    "variables": {  
    },  
    "resources": [  
    ]  
}
```

2. Parameters are not always required, but they provide a way to input values when the template is deployed.
Add these parameters under the parameters parent element that you added to the template.

```
"vmName": { "type": "string" },  
"vmSSName": { "type": "string" },  
"instanceCount": { "type": "string" },  
"adminUsername": { "type": "string" },  
"adminPassword": { "type": "securestring" },  
"resourcePrefix": { "type": "string" }
```

- A name for the separate virtual machine that is used to access the machines in the scale set.
 - The name of the storage account where the template is stored.
 - The number of virtual machines to initially create in the scale set.
 - The name and password of the administrator account on the virtual machines.
 - A name prefix for the resources that are created to support the scale set.
3. Variables can be used in a template to specify values that may change frequently or values that need to be created from a combination of parameter values. Add these variables under the variables parent element that you added to the template.

```

"dnsName1": "[concat(parameters('resourcePrefix'), 'dn1')]",
"dnsName2": "[concat(parameters('resourcePrefix'), 'dn2')]",
"publicIP1": "[concat(parameters('resourcePrefix'), 'ip1')]",
"publicIP2": "[concat(parameters('resourcePrefix'), 'ip2')]",
"loadBalancerName": "[concat(parameters('resourcePrefix'), 'lb1')]",
"virtualNetworkName": "[concat(parameters('resourcePrefix'), 'vn1')]",
"nicName": "[concat(parameters('resourcePrefix'), 'nc1')]",
"lbID": "[resourceId('Microsoft.Network/loadBalancers', variables('loadBalancerName'))]",
"frontEndIPConfigID": "[concat(variables('lbID'), '/frontendIPConfigurations/loadBalancerFrontEnd')]",
"storageAccountSuffix": [ "a", "g", "m", "s", "y" ],
"diagnosticsStorageAccountName": "[concat(parameters('resourcePrefix'), 'a')]",
"accountid": "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/', 'Microsoft.Storage/storageAccounts/',
variables('diagnosticsStorageAccountName'))]",
"wadlogs": "<WadCfg> <DiagnosticMonitorConfiguration overallQuotaInMB=\"4096\""
xmlNs=\"http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration\">
<DiagnosticInfrastructureLogs scheduledTransferLogLevelFilter=\"Error\"/> <WindowsEventLog
scheduledTransferPeriod=\"PT1M\"> <DataSource name=\"Application!*[System[(Level = 1 or Level = 2)]]\"/>
<DataSource name=\"Security!*[System[(Level = 1 or Level = 2)]]\" /> <DataSource name=\"System!*
[System[(Level = 1 or Level = 2)]]\" /></WindowsEventLog>",
"wadperfcounter": "<PerformanceCounters scheduledTransferPeriod=\"PT1M\">
<PerformanceCounterConfiguration counterSpecifier=\"\\Processor(_Total)\\% Processor Time\"
sampleRate=\"PT1S\" unit=\"Percent\"><annotation displayName=\"CPU utilization\" locale=\"en-us\"/>
</PerformanceCounterConfiguration></PerformanceCounters>",
"wadcfgxstart": "[concat(variables('wadlogs'), variables('wadperfcounter'), '<Metrics resourceId=''))",
"wadmetricsresourceid": "
[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name
,'/providers/', 'Microsoft.Compute/virtualMachineScaleSets/', parameters('vmssName'))]",
"wadcfgxend": "[concat('\'><MetricAggregation scheduledTransferPeriod=\"PT1H\"/><MetricAggregation
scheduledTransferPeriod=\"PT1M\"/><Metrics></DiagnosticMonitorConfiguration></WadCfg>')]"

```

- DNS names that are used by the network interfaces.
 - The IP address names and prefixes for the virtual network and subnets.
 - The names and identifiers of the virtual network, load balancer, and network interfaces.
 - Storage account names for the accounts associated with the machines in the scale set.
 - Settings for the Diagnostics extension that is installed on the virtual machines. For more information about the Diagnostics extension, see [Create a Windows Virtual machine with monitoring and diagnostics using Azure Resource Manager Template](#).

4. Add the storage account resource under the resources parent element that you added to the template. This template uses a loop to create the recommended five storage accounts where the operating system disks and diagnostic data are stored. This set of accounts can support up to 100 virtual machines in a scale set, which is the current maximum. Each storage account is named with a letter designator that was defined in the variables combined with the prefix that you provide in the parameters for the template.

```
{
  "type": "Microsoft.Storage/storageAccounts",
  "name": "[concat(parameters('resourcePrefix'), variables('storageAccountSuffix')[copyIndex()])]",
  "apiVersion": "2015-06-15",
  "copy": {
    "name": "storageLoop",
    "count": 5
  },
  "location": "[resourceGroup().location]",
  "properties": { "accountType": "Standard_LRS" }
},
```

5. Add the virtual network resource. For more information, see [Network Resource Provider](#).

```
{
  "apiVersion": "2015-06-15",
  "type": "Microsoft.Network/virtualNetworks",
  "name": "[variables('virtualNetworkName')]",
  "location": "[resourceGroup().location]",
  "properties": {
    "addressSpace": { "addressPrefixes": [ "10.0.0.0/16" ] },
    "subnets": [
      {
        "name": "subnet1",
        "properties": { "addressPrefix": "10.0.0.0/24" }
      }
    ]
  }
},
```

- Add the public IP address resources that are used by the load balancer and network interface.

```
{
  "apiVersion": "2016-03-30",
  "type": "Microsoft.Network/publicIPAddresses",
  "name": "[variables('publicIP1')]",
  "location": "[resourceGroup().location]",
  "properties": {
    "publicIPAllocationMethod": "Dynamic",
    "dnsSettings": {
      "domainNameLabel": "[variables('dnsName1')]"
    }
  }
},
{
  "apiVersion": "2016-03-30",
  "type": "Microsoft.Network/publicIPAddresses",
  "name": "[variables('publicIP2')]",
  "location": "[resourceGroup().location]",
  "properties": {
    "publicIPAllocationMethod": "Dynamic",
    "dnsSettings": {
      "domainNameLabel": "[variables('dnsName2')]"
    }
  }
},
```

- Add the load balancer resource that is used by the scale set. For more information, see [Azure Resource Manager Support for Load Balancer](#).

```
{
  "apiVersion": "2015-06-15",
  "name": "[variables('loadBalancerName')]",
  "type": "Microsoft.Network/loadBalancers",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Network/publicIPAddresses/', variables('publicIP1'))]"
  ],
  "properties": {
    "frontendIPConfigurations": [
      {
        "name": "loadBalancerFrontEnd",
        "properties": {
          "publicIPAddress": {
            "id": "[concat('Microsoft.Network/publicIPAddresses/', variables('publicIP1'))]"
          }
        }
      }
    ],
    "backendAddressPools": [ { "name": "bePool1" } ],
    "inboundNatPools": [
      {
        "name": "natpool1",
        "properties": {
          "frontendIPConfiguration": {
            "id": "[variables('frontEndIPConfigID')]"
          },
          "protocol": "tcp",
          "frontendPortRangeStart": 50000,
          "frontendPortRangeEnd": 50500,
          "backendPort": 3389
        }
      }
    ]
  }
},
```

8. Add the network interface resource that is used by the separate virtual machine. Because machines in a scale set aren't accessible through a public IP address, a separate virtual machine is created in the same virtual network to remotely access the machines.

```
{
  "apiVersion": "2016-03-30",
  "type": "Microsoft.Network/networkInterfaces",
  "name": "[variables('nicName')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Network/publicIPAddresses/', variables('publicIP2'))]",
    "[concat('Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'))]"
  ],
  "properties": {
    "ipConfigurations": [
      {
        "name": "ipconfig1",
        "properties": {
          "privateIPAllocationMethod": "Dynamic",
          "publicIPAddress": {
            "id": "[resourceId('Microsoft.Network/publicIPAddresses', variables('publicIP2'))]"
          },
          "subnet": {
            "id": "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/', resourceGroup().name, '/providers/Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'), '/subnets/subnet1')]"
          }
        }
      }
    ]
  }
},
```

9. Add the separate virtual machine in the same network as the scale set.

```
{
  "apiVersion": "2016-03-30",
  "type": "Microsoft.Compute/virtualMachines",
  "name": "[parameters('vmName')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "storageLoop",
    "[concat('Microsoft.Network/networkInterfaces/', variables('nicName'))]"
  ],
  "properties": {
    "hardwareProfile": { "vmSize": "Standard_A1" },
    "osProfile": {
      "computerName": "[parameters('vmName')]",
      "adminUsername": "[parameters('adminUsername')]",
      "adminPassword": "[parameters('adminPassword')]"
    },
    "storageProfile": {
      "imageReference": {
        "publisher": "MicrosoftWindowsServer",
        "offer": "WindowsServer",
        "sku": "2012-R2-Datacenter",
        "version": "latest"
      },
      "osDisk": {
        "name": "[concat(parameters('resourcePrefix'), 'os1')]",
        "vhd": {
          "uri": "[concat('https://', parameters('resourcePrefix'), 'a.blob.core.windows.net/vhds/', parameters('resourcePrefix'), 'os1.vhd')]"
        },
        "caching": "ReadWrite",
        "createOption": "FromImage"
      }
    },
    "networkProfile": {
      "networkInterfaces": [
        {
          "id": "[resourceId('Microsoft.Network/networkInterfaces', variables('nicName'))]"
        }
      ]
    }
  }
},
```

10. Add the virtual machine scale set resource and specify the diagnostics extension that is installed on all virtual machines in the scale set. Many of the settings for this resource are similar with the virtual machine resource. The main differences are the capacity element that specifies the number of virtual machines in the scale set, and upgradePolicy that specifies how updates are made to virtual machines. The scale set is not created until all the storage accounts are created as specified with the dependsOn element.

```
{
  "type": "Microsoft.Compute/virtualMachineScaleSets",
  "apiVersion": "2016-03-30",
  "name": "[parameters('vmSSName')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "storageLoop",
    "[concat('Microsoft.Network/virtualNetworks/', variables('virtualNetworkName'))]",
    "[concat('Microsoft.Network/loadBalancers/', variables('loadBalancerName'))]"
  ],
  "sku": {
    "name": "Standard_A1",
    "tier": "Standard",
    "capacity": "[parameters('instanceCount')]"
  },
```

```

    "properties": {
        "upgradePolicy": {
            "mode": "Manual"
        },
        "virtualMachineProfile": {
            "storageProfile": {
                "osDisk": {
                    "vhdContainers": [
                        "[concat('https://', parameters('resourcePrefix'), variables('storageAccountSuffix')[0],
'.blob.core.windows.net/vhds')]",
                        "[concat('https://', parameters('resourcePrefix'), variables('storageAccountSuffix')[1],
'.blob.core.windows.net/vhds')]",
                        "[concat('https://', parameters('resourcePrefix'), variables('storageAccountSuffix')[2],
'.blob.core.windows.net/vhds')]",
                        "[concat('https://', parameters('resourcePrefix'), variables('storageAccountSuffix')[3],
'.blob.core.windows.net/vhds')]",
                        "[concat('https://', parameters('resourcePrefix'), variables('storageAccountSuffix')[4],
'.blob.core.windows.net/vhds')]"
                    ],
                    "name": "vmssosdisk",
                    "caching": "ReadOnly",
                    "createOption": "FromImage"
                },
                "imageReference": {
                    "publisher": "MicrosoftWindowsServer",
                    "offer": "WindowsServer",
                    "sku": "2012-R2-Datacenter",
                    "version": "latest"
                }
            },
            "osProfile": {
                "computerNamePrefix": "[parameters('vmSSName')]",
                "adminUsername": "[parameters('adminUsername')]",
                "adminPassword": "[parameters('adminPassword')]"
            },
            "networkProfile": {
                "networkInterfaceConfigurations": [
                    {
                        "name": "networkconfig1",
                        "properties": {
                            "primary": "true",
                            "ipConfigurations": [
                                {
                                    "name": "ip1",
                                    "properties": {
                                        "subnet": {
                                            "id": "[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/virtualNetworks/',
variables('virtualNetworkName'), '/subnets/subnet1')]"
                                        },
                                        "loadBalancerBackendAddressPools": [
                                            {
                                                "id": "
[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('loadBalancerName'), '/backendAddressPools/bepool1')]"
                                            }
                                        ],
                                        "loadBalancerInboundNatPools": [
                                            {
                                                "id": "
[concat('/subscriptions/', subscription().subscriptionId, '/resourceGroups/',
resourceGroup().name, '/providers/Microsoft.Network/loadBalancers/',
variables('loadBalancerName'), '/inboundNatPools/natpool1')]"
                                            }
                                        ]
                                    }
                                }
                            ]
                        }
                    }
                ]
            }
        }
    }
}

```

```
        ]
    },
    "extensionProfile": {
        "extensions": [
            {
                "name": "Microsoft.Insights.VMDiagnosticsSettings",
                "properties": {
                    "publisher": "Microsoft.Azure.Diagnostics",
                    "type": "IaaSDiagnostics",
                    "typeHandlerVersion": "1.5",
                    "autoUpgradeMinorVersion": true,
                    "settings": {
                        "xmlCfg": "[base64(concat(variables('wadcfgxstart'),variables('wadmetricsresourceid'),variables('wadcfgxend')))]",
                        "storageAccount": "[variables('diagnosticsStorageAccountName')]"
                    },
                    "protectedSettings": {
                        "storageAccountName": "[variables('diagnosticsStorageAccountName')]",
                        "storageAccountKey": "[listkeys(variables('accountid'), '2015-06-15').key1]",
                        "storageAccountEndPoint": "https://core.windows.net"
                    }
                }
            }
        ]
    }
},
```

11. Add the autoscaleSettings resource that defines how the scale set adjusts based on the processor usage on the machines in the scale set.

```
{
  "type": "Microsoft.Insights/autoscaleSettings",
  "apiVersion": "2015-04-01",
  "name": "[concat(parameters('resourcePrefix'),'as1')]",
  "location": "[resourceGroup().location]",
  "dependsOn": [
    "[concat('Microsoft.Compute/virtualMachineScaleSets/',parameters('vmSSName'))]"
  ],
  "properties": {
    "enabled": true,
    "name": "[concat(parameters('resourcePrefix'),'as1')]",
    "profiles": [
      {
        "name": "Profile1",
        "capacity": {
          "minimum": "1",
          "maximum": "10",
          "default": "1"
        },
        "rules": [
          {
            "metricTrigger": {
              "metricName": "\Processor(_Total)\% Processor Time",
              "metricNamespace": "",
              "metricResourceUri": "[concat('/subscriptions/',subscription().subscriptionId,'/resourceGroups/',resourceGroup().name,'/providers/Microsoft.Compute/virtualMachineScaleSets/',parameters('vmSSName'))]",
              "timeGrain": "PT1M",
              "statistic": "Average",
              "timeWindow": "PT5M",
              "timeAggregation": "Average",
              "operator": "GreaterThan",
              "threshold": 50.0
            },
            "scaleAction": {
              "direction": "Increase",
              "type": "ChangeCount",
              "value": "1",
              "cooldown": "PT5M"
            }
          }
        ]
      }
    ],
    "targetResourceUri": "[concat('/subscriptions/',subscription().subscriptionId,'/resourceGroups/',resourceGroup().name,'/providers/Microsoft.Compute/virtualMachineScaleSets/',parameters('vmSSName'))]"
  }
}
```

For this tutorial, these values are important:

- **metricName**

This value is the same as the performance counter that we defined in the wadperfcounter variable.

Using that variable, the Diagnostics extension collects the **Processor(_Total)\% Processor Time** counter.

- **metricResourceUri**

This value is the resource identifier of the virtual machine scale set.

- **timeGrain**

This value is the granularity of the metrics that are collected. In this template, it is set to one minute.

- **statistic**

This value determines how the metrics are combined to accommodate the automatic scaling action.

The possible values are: Average, Min, Max. In this template, the average total CPU usage of the virtual machines is collected.

- **timeWindow**

This value is the range of time in which instance data is collected. It must be between 5 minutes and 12 hours.

- **timeAggregation**

This value determines how the data that is collected should be combined over time. The default value is Average. The possible values are: Average, Minimum, Maximum, Last, Total, Count.

- **operator**

This value is the operator that is used to compare the metric data and the threshold. The possible values are: Equals, NotEquals, GreaterThan, GreaterThanOrEqual, LessThan, LessThanOrEqual.

- **threshold**

This value is the value that triggers the scale action. In this template, machines are added to the scale set when the average CPU usage among machines in the set is over 50%.

- **direction**

This value determines the action that is taken when the threshold value is achieved. The possible values are Increase or Decrease. In this template, the number of virtual machines in the scale set is increased if the threshold is over 50% in the defined time window.

- **type**

This value is the type of action that should occur and must be set to ChangeCount.

- **value**

This value is the number of virtual machines that are added or removed from the scale set. This value must be 1 or greater. The default value is 1. In this template, the number of machines in the scale set increases by 1 when the threshold is met.

- **cooldown**

This value is the amount of time to wait since the last scaling action before the next action occurs.

This value must be between one minute and one week.

12. Save the template file.

Step 4: Upload the template to storage

The template can be uploaded as long as you know the name and primary key of the storage account that you created in step 1.

1. In the Microsoft Azure PowerShell window, set a variable that specifies the name of the storage account that you created in step 1.

```
$storageAccountName = "vmstestsa"
```

2. Set a variable that specifies the primary key of the storage account.

```
$storageAccountKey = "<primary-account-key>"
```

You can get this key by clicking the key icon when viewing the storage account resource in the Azure portal.

3. Create the storage account context object that is used to validate operations with the storage account.

```
$ctx = New-AzureStorageContext -StorageAccountName $storageAccountName -StorageAccountKey  
$storageAccountKey
```

4. Create the container for storing the template.

```
$containerName = "templates"  
New-AzureStorageContainer -Name $containerName -Context $ctx -Permission Blob
```

5. Upload the template file to the new container.

```
$blobName = "VMSSTemplate.json"  
$fileName = "C:\" + $blobName  
Set-AzureStorageBlobContent -File $fileName -Container $containerName -Blob $blobName -Context $ctx
```

Step 5: Deploy the template

Now that you created the template, you can start deploying the resources. Use this command to start the process:

```
New-AzureRmResourceGroupDeployment -Name "vmsstestdp1" -ResourceGroupName "vmsstestrg1" -TemplateUri  
"https://vmsstestsa.blob.core.windows.net/templates/VMSSTemplate.json"
```

When you press enter, you are prompted to provide values for the variables you assigned. Provide these values:

```
vmName: vmsstestvm1  
vmSSName: vmsstest1  
instanceCount: 5  
adminUserName: vmadmin1  
adminPassword: VMpass1  
resourcePrefix: vmsstest
```

It should take about 15 minutes for all the resources to successfully be deployed.

NOTE

You can also use the portal's ability to deploy the resources. Use this link:
["https://portal.azure.com/#create/Microsoft.Template/uri/"](https://portal.azure.com/#create/Microsoft.Template/uri/)

Step 6: Monitor resources

You can get some information about virtual machine scale sets using these methods:

- The Azure portal - You can currently get a limited amount of information using the portal.
- The [Azure Resource Explorer](#) - This tool is the best for exploring the current state of your scale set. Follow this path and you should see the instance view of the scale set that you created:

subscriptions > {your subscription} > resourceGroups > vmsstestrg1 > providers > Microsoft.Compute > virtualMachineScaleSets > vmsstest1 > virtualMachines

- Azure PowerShell - Use this command to get some information:

```
Get-AzureRmVmss -ResourceGroupName "resource group name" -VMScaleSetName "scale set name"
```

Or

```
Get-AzureRmVmss -ResourceGroupName "resource group name" -VMScaleSetName "scale set name" -InstanceView
```

- Connect to the separate virtual machine just like you would any other machine and then you can remotely access the virtual machines in the scale set to monitor individual processes.

NOTE

A complete REST API for obtaining information about scale sets can be found in [Virtual Machine Scale Sets](#)

Step 7: Remove the resources

Because you are charged for resources used in Azure, it is always a good practice to delete resources that are no longer needed. You don't need to delete each resource separately from a resource group. You can delete the resource group and all its resources are automatically deleted.

```
Remove-AzureRmResourceGroup -Name vmsstestrg1
```

If you want to keep your resource group, you can delete the scale set only.

```
Remove-AzureRmVmss -ResourceGroupName "resource group name" -VMScaleSetName "scale set name"
```

Next steps

- Manage the scale set that you just created using the information in [Manage virtual machines in a Virtual Machine Scale Set](#).
- Learn more about vertical scaling by reviewing [Vertical autoscale with Virtual Machine Scale sets](#)
- Find examples of Azure Monitor monitoring features in [Azure Monitor PowerShell quick start samples](#)
- Learn about notification features in [Use autoscale actions to send email and webhook alert notifications in Azure Monitor](#)
- Learn how to [Use audit logs to send email and webhook alert notifications in Azure Monitor](#)

Use autoscale actions to send email and webhook alert notifications in Azure Monitor

7/18/2017 • 3 min to read • [Edit Online](#)

This article shows you how set up triggers so that you can call specific web URLs or send emails based on autoscale actions in Azure.

Webhooks

Webhooks allow you to route the Azure alert notifications to other systems for post-processing or custom notifications. For example, routing the alert to services that can handle an incoming web request to send SMS, log bugs, notify a team using chat or messaging services, etc. The webhook URI must be a valid HTTP or HTTPS endpoint.

Email

Email can be sent to any valid email address. Administrators and co-administrators of the subscription where the rule is running will also be notified.

Cloud Services and Web Apps

You can opt-in from the Azure portal for Cloud Services and Server Farms (Web Apps).

- Choose the **scale by** metric.

The screenshot shows the Azure portal interface for managing autoscale settings. At the top, it says "Autoscale setting" and "WebAppAutoscaleTest (App Service plan)". Below that are buttons for "Save", "Discard", and "Disable autoscale". A navigation bar includes "Configure", "Run history", "JSON", and "Notify", with "Notify" being the active tab and highlighted with a red border. Under the "Notify" tab, there are two checkboxes: "Email administrators" and "Email co-administrators". Below these are fields for "Additional administrator email(s)" and "Add email addresses separated by semicolons". At the bottom, there is a "Webhook" section with a "HTTP or HTTPS endpoint to route autoscale notifications to" input field.

Virtual Machine scale sets

For newer Virtual Machines created with Resource Manager (Virtual Machine scale sets), you can configure this using REST API, Resource Manager templates, PowerShell, and CLI. A portal interface is not yet available. When using the REST API or Resource Manager template, include the notifications element with the following options.

```

"notifications": [
  {
    "operation": "Scale",
    "email": {
      "sendToSubscriptionAdministrator": false,
      "sendToSubscriptionCoAdministrators": false,
      "customEmails": [
        "user1@mycompany.com",
        "user2@mycompany.com"
      ]
    },
    "webhooks": [
      {
        "serviceUri": "https://foo.webhook.example.com?token=abcd1234",
        "properties": {
          "optional_key1": "optional_value1",
          "optional_key2": "optional_value2"
        }
      }
    ]
  }
]

```

| FIELD | MANDATORY? | DESCRIPTION |
|------------------------------------|------------|---|
| operation | yes | value must be "Scale" |
| sendToSubscriptionAdministrator | yes | value must be "true" or "false" |
| sendToSubscriptionCoAdministrators | yes | value must be "true" or "false" |
| customEmails | yes | value can be null [] or string array of emails |
| webhooks | yes | value can be null or valid Uri |
| serviceUri | yes | a valid https Uri |
| properties | yes | value must be empty {} or can contain key-value pairs |

Authentication in webhooks

The webhook can authenticate using token-based authentication, where you save the webhook URI with a token ID as a query parameter. For example, <https://mysamplealert/webcallback?tokenid=sometokenid&someparameter=somevalue>

Autoscale notification webhook payload schema

When the autoscale notification is generated, the following metadata is included in the webhook payload:

```
{
    "version": "1.0",
    "status": "Activated",
    "operation": "Scale In",
    "context": {
        "timestamp": "2016-03-11T07:31:04.5834118Z",
        "id": ""
    },
    "/subscriptions/s1/resourceGroups/rg1/providers/microsoft.insights/autoscalesettings/myautoscaleSetting",
    "name": "myautoscaleSetting",
    "details": "Autoscale successfully started scale operation for resource 'MyCSRole' from capacity '3' to capacity '2'",
    "subscriptionId": "s1",
    "resourceGroupName": "rg1",
    "resourceName": "MyCSRole",
    "resourceType": "microsoft.classiccompute/domainnames/slots/roles",
    "resourceId": "",
    "/subscriptions/s1/resourceGroups/rg1/providers/microsoft.classicCompute/domainNames/myCloudService/slots/Production/roles/MyCSRole",
    "portalLink": "",
    "https://portal.azure.com/#resource/subscriptions/s1/resourceGroups/rg1/providers/microsoft.classicCompute/domainNames/myCloudService",
    "oldCapacity": "3",
    "newCapacity": "2"
},
    "properties": {
        "key1": "value1",
        "key2": "value2"
    }
}
}
```

| FIELD | MANDATORY? | DESCRIPTION |
|-------------------|------------|---|
| status | yes | The status that indicates that an autoscale action was generated |
| operation | yes | For an increase of instances, it will be "Scale Out" and for a decrease in instances, it will be "Scale In" |
| context | yes | The autoscale action context |
| timestamp | yes | Time stamp when the autoscale action was triggered |
| id | Yes | Resource Manager ID of the autoscale setting |
| name | Yes | The name of the autoscale setting |
| details | Yes | Explanation of the action that the autoscale service took and the change in the instance count |
| subscriptionId | Yes | Subscription ID of the target resource that is being scaled |
| resourceGroupName | Yes | Resource Group name of the target resource that is being scaled |

| FIELD | MANDATORY? | DESCRIPTION |
|--------------|------------|--|
| resourceName | Yes | Name of the target resource that is being scaled |
| resourceType | Yes | The three supported values:
"microsoft.classiccompute/domainnameslots/roles" - Cloud Service roles,
"microsoft.compute/virtualmachinescalesets" - Virtual Machine Scale Sets, and
"Microsoft.Web/serverfarms" - Web App |
| resourceId | Yes | Resource Manager ID of the target resource that is being scaled |
| portalLink | Yes | Azure portal link to the summary page of the target resource |
| oldCapacity | Yes | The current (old) instance count when Autoscale took a scale action |
| newCapacity | Yes | The new instance count that Autoscale scaled the resource to |
| Properties | No | Optional. Set of pairs (for example, Dictionary). The properties field is optional. In a custom user interface or Logic app based workflow, you can enter keys and values that can be passed using the payload. An alternate way to pass custom properties back to the outgoing webhook call is to use the webhook URI itself (as query parameters) |

View activity logs to audit actions on resources

6/27/2017 • 3 min to read • [Edit Online](#)

Through activity logs, you can determine:

- what operations were taken on the resources in your subscription
- who initiated the operation (although operations initiated by a backend service do not return a user as the caller)
- when the operation occurred
- the status of the operation
- the values of other properties that might help you research the operation

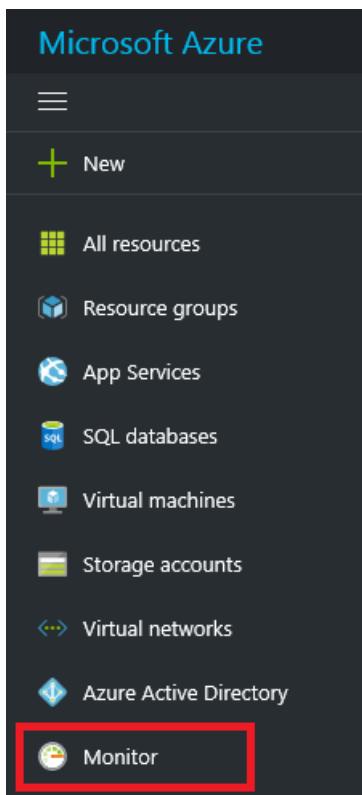
The activity log contains all write operations (PUT, POST, DELETE) performed on your resources. It does not include read operations (GET). You can use the audit logs to find an error when troubleshooting or to monitor how a user in your organization modified a resource.

Activity logs are retained for 90 days. You can query for any range of dates, as long as the starting date is not more than 90 days in the past.

You can retrieve information from the activity logs through the portal, PowerShell, Azure CLI, Insights REST API, or [Insights .NET Library](#).

Portal

1. To view the activity logs through the portal, select **Monitor**.



Or, to automatically filter the activity log for a particular resource or resource group, select **Activity log** from that resource blade. Notice that the activity log is automatically filtered by the selected resource.

Storage demo - Activity log

Storage account

Columns Export

Search (Ctrl+ /)

Overview

Activity log (highlighted)

Access control (IAM)

Select query ...

* Subscription: Windows Azure MSDN

Resource group: demo-group (highlighted)

Resource: storagedemo (highlighted)

Timespan: Last 1 hour

Event category: All categories

Event severity: 4 selected

Apply Reset

Insights (Last 24 hours): 1 failed dep
outage notifications

2. In the **Activity Log** blade, you see a summary of recent operations.

Activity log

Columns Export

Select query ...

* Subscription: Windows Azure MSDN - Visual...

Resource group: All resource groups

Resource: All resources

Timespan: Last 1 hour

Event category: All categories

Event severity: 4 selected

Apply (highlighted)

Reset

Insights (Last 24 hours): 5 items. Click here to download all the items as csv.

| OPERATION NAME | STATUS | TIME | TIME STAMP |
|-----------------------|-----------|-----------|---------------|
| Write Deployments | Failed | Just now | Mon Aug 22... |
| Validate | Succeeded | Just now | Mon Aug 22... |
| Register | Succeeded | 3 min ago | Mon Aug 22... |
| Delete resource group | Succeeded | 3 min ago | Mon Aug 22... |
| Register | Succeeded | 3 min ago | Mon Aug 22... |

3. To restrict the number of operations displayed, select different conditions. For example, the following image shows the **Timespan** and **Event initiated by** fields changed to view the actions taken by a particular user or application for the past month. Select **Apply** to view the results of your query.

Activity log

Columns Export

Select query ...

* Subscription: Windows Azure MSDN

Resource group: All resource groups

Resource: All resources

Resource type: All resource types

Timespan: Last month (highlighted)

Event category: All categories

Event severity: 4 selected

Event initiated by: CloudSense (highlighted)

Apply Reset

Insights (Last 24 hours): 1 failed dep

4. If you need to run the query again later, select **Save** and give the query a name.

Activity log

Columns Export

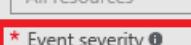
Select query ... 

5. To quickly run a query, you can select one of the built-in queries, such as failed deployments.

Insights (Last 24 hours): **1 failed deployment** | 0 role assignments | 1 error

The selected query automatically sets the required filter values.

Select query ...  Insights (Last 24 hours): **1 failed deployment** | 0 role assignments | 1 error

* Subscription  Resource group  Resource  Resource type 
2 selected All resource groups All resources Deployment (deployments)
Timespan  Event category  Event severity  Event initiated by 
Last 24 hours All categories Error All
Apply Reset

Query returned 1 items. [Click here to download all the items as csv.](#)

| OPERATION NAME | STATUS | TIME | TIME STAMP | SUBSCR... |
|----------------|--------|-----------|-----------------|------------|
| ! Validate | Failed | 4 min ago | Mon Jan 09 2... | Third I... |

6. Select one of the operations to see a summary of the event.

OPERATION NAME

 Validate

Validate

Summary JSON

Operation name
Validate

Time stamp
Mon Jan 09 2017 14:14:39 GMT-0800 (Pacific Standard Time)

Event initiated by

Error code
KeyVaultParameterReferenceNotFound

Message
The specified KeyVault '/subscriptions/ ... could not be found.

PowerShell

1. To retrieve log entries, run the **Get-AzureRmLog** command. You provide additional parameters to filter the list of entries. If you do not specify a start and end time, entries for the last hour are returned. For example, to retrieve the operations for a resource group during the past hour run:

```
Get-AzureRmLog -ResourceGroup ExampleGroup
```

The following example shows how to use the activity log to research operations taken during a specified time. The start and end dates are specified in a date format.

```
Get-AzureRmLog -ResourceGroup ExampleGroup -StartTime 2015-08-28T06:00 -EndTime 2015-09-10T06:00
```

Or, you can use date functions to specify the date range, such as the last 14 days.

```
Get-AzureRmLog -ResourceGroup ExampleGroup -StartTime (Get-Date).AddDays(-14)
```

2. Depending on the start time you specify, the previous commands can return a long list of operations for the resource group. You can filter the results for what you are looking for by providing search criteria. For example, if you are trying to research how a web app was stopped, you could run the following command:

```
Get-AzureRmLog -ResourceGroup ExampleGroup -StartTime (Get-Date).AddDays(-14) | Where-Object  
OperationName -eq Microsoft.Web/sites/stop/action
```

Which for this example shows that a stop action was performed by someone@contoso.com.

```
Authorization      :  
Scope      : /subscriptions/xxxxx/resourcegroups/ExampleGroup/providers/Microsoft.Web/sites/ExampleSite  
Action     : Microsoft.Web/sites/stop/action  
Role       : Subscription Admin  
Condition   :  
Caller      : someone@contoso.com  
CorrelationId : 84beae59-92aa-4662-a6fc-b6fecc0ff8da  
EventSource   : Administrative  
EventTimestamp : 8/28/2015 4:08:18 PM  
OperationName : Microsoft.Web/sites/stop/action  
ResourceGroupName : ExampleGroup  
ResourceId    :  
/subscriptions/xxxxx/resourcegroups/ExampleGroup/providers/Microsoft.Web/sites/ExampleSite  
Status       : Succeeded  
SubscriptionId : xxxxx  
SubStatus    : OK
```

3. You can look up the actions taken by a particular user, even for a resource group that no longer exists.

```
Get-AzureRmLog -ResourceGroup deletedgroup -StartTime (Get-Date).AddDays(-14) -Caller  
someone@contoso.com
```

4. You can filter for failed operations.

```
Get-AzureRmLog -ResourceGroup ExampleGroup -Status Failed
```

5. You can focus on one error by looking at the status message for that entry.

```
((Get-AzureRmLog -Status Failed -ResourceGroup ExampleGroup -  
DetailedOutput).Properties[1].Content["statusMessage"] | ConvertFrom-Json).error
```

Which returns:

```
code          message
----          -----
DnsRecordInUse DNS record dns.westus.cloudapp.azure.com is already used by another public IP.
```

Azure CLI

- To retrieve log entries, you run the **azure group log show** command.

```
azure group log show ExampleGroup --json
```

REST API

The REST operations for working with the activity log are part of the [Insights REST API](#). To retrieve activity log events, see [List the management events in a subscription](#).

Next steps

- Azure Activity logs can be used with Power BI to gain greater insights about the actions in your subscription. See [View and analyze Azure Activity Logs in Power BI and more](#).
- To learn about setting security policies, see [Azure Role-based Access Control](#).
- To learn about the commands for viewing deployment operations, see [View deployment operations](#).
- To learn how to prevent deletions on a resource for all users, see [Lock resources with Azure Resource Manager](#).

Create Activity Log Alerts

7/18/2017 • 4 min to read • [Edit Online](#)

Overview

Activity Log Alerts are alerts that activate when a new Activity Log event matches the conditions specified in the alert. They are Azure resources so they can be managed using the Azure portal or the Azure Resource Manager (ARM). This article shows you how to use the Azure portal to set up alerts on activity log events.

You can receive alerts about operations that were performed on resources in your subscription or about service health events that may impact the health of your resources. An Activity Log Alert monitors the activity log events for the specific subscription the alert is deployed to.

You can configure the alert based on:

- Event Category (for [service health events click here](#))
- Resource Group
- Resource
- Resource Type
- Operation name
- The level of the notifications (Verbose, Informational, Warning, Error, Critical)
- Status
- Event initiated by

NOTE

You must specify at least one of the above criteria in your alert. You may not create an alert that activates every time an event is created in the Activity Logs.

You can also the configure the actions that will be executed when the alert is activated:

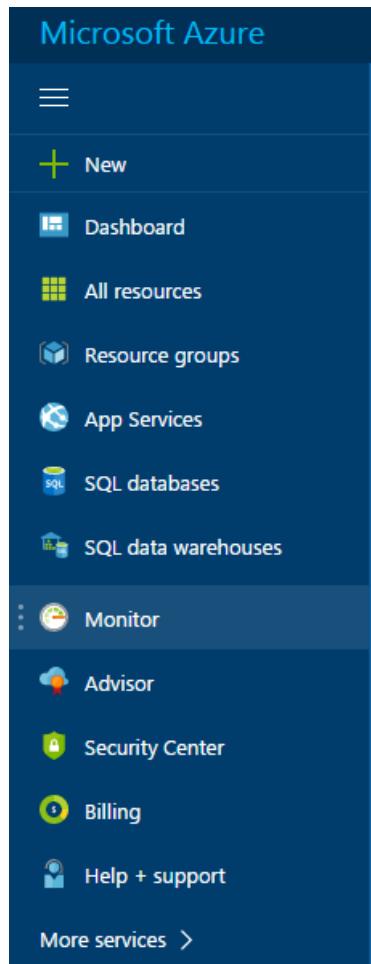
- Select an existing Action Group
- Create a new Action Group (that can be later used for future alerts)

You can learn more about [Action Groups here](#)

You can learn about Activity Log Alerts for service health notification [here](#)

Create an alert on an activity log event with a new action group with the Azure Portal

1. In the [portal](#), navigate to the **Monitor** service



2. Click the **Monitor** option to open the Monitor blade. It first opens to the **Activity log** section.
3. Now click on the **Alerts** section.

The image shows the 'Monitor - Alerts' blade. At the top is a search bar labeled 'Search (Ctrl+ /)'. Below it are two sections: 'EXPLORE' and 'MANAGE'. The 'EXPLORE' section contains links for 'Activity log', 'Metrics', 'Diagnostics logs', 'Log search', and 'Service notifications'. The 'MANAGE' section contains links for 'Alerts' (which is highlighted with a blue background) and 'Action groups'.

4. Select the **Add activity log alert** command and fill in the fields
5. **Name** your activity log alert, and choose a **Description**.

Monitor - Alerts

Search (Ctrl+ /)

Columns

+ Add metric alert

+ Add activity log alert

- The **Subscription** is the one the Activity log alert will be saved in. It will be auto filled to the subscription you are currently operating under. This is the subscription the alert resource will be deployed to and monitor.

Add activity log alert

* Activity log alert name: Sample Service Health Alert ✓

Description: Alerts whenever an incident post occurs for this subscription. ✓

* Subscription: Azure LSM and Communications - Incubation (Test Env on...) ✓

* Resource group: Default-Storage-EastUS ✓

Source

Input: Activity log

Criteria

* Event category: Administrative ✓

Resource group: Default-Storage-SoutheastAsia ✓

Resource: matsappstoragetestsg1 ✓

Resource type: Storage Account (storageAccounts) ✓

Operation name: Delete Storage Account (storageAccounts) ✓

Level: All ✓

Status: Started ✓

Event initiated by: [empty]

Notify via

Action group: New Existing

* Action group name: Sample-Action-Group ✓

* Short name: SampleAG ✓

Receivers

| NAME | ACTION TYPE | DETAILS |
|-----------------------------|-------------|----------------------------------|
| SMS-Receiver | SMS | 555-555-5555 ... |
| Email-Receiver | Email | sample@contoso.com ... |
| Webhook-Receiver | Webhook | https://sample.webhook... ... |
| The name of the receiver... | SMS | Depending on the action type ... |

OK

- Choose the **Resource Group** this alert will be associated with in the **Subscription**.
- Provide the **Event Category, Resource Group, Resource Type, Operation Name, Level, Status** and **Event initiated by** values to identify which events this alert should monitor.

NOTE

You must specify at least one of the above criteria in your alert. You may not create an alert that activates every time an event is created in the Activity Logs.

1. Create a **New** Action Group by giving it **Name** and **Short Name**; the Short Name will be referenced in the notifications sent when this alert is activated.
2. Then, define a list of actions by providing the action's
 - a. **Name:** Action's name, alias or identifier.
 - b. **Action Type:** Choose the action type: SMS, Email, or Webhook
 - c. **Details:** Based on the action type chosen, provide a phone number, email address or webhook URI.
3. Select **OK** when done to create the alert.

Within a few minutes, the alert is active and triggers as previously described.

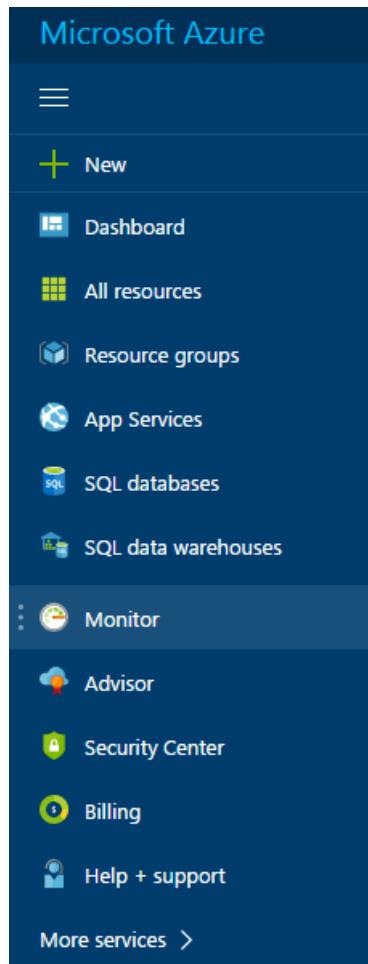
For details on the webhook schema for activity log alerts [click here](#)

NOTE

The action group defined in these steps will be reusable, as an existing action group, for all future alert definition.

Create an alert on an activity log event for an existing action group with the Azure Portal

1. In the [portal](#), navigate to the **Monitor** service



2. Click the **Monitor** option to open the Monitor blade. It first opens to the **Activity log** section.
3. Now click on the **Alerts** section.

This screenshot shows the 'Monitor - Alerts' blade. At the top is a dark blue header bar with the title 'Monitor - Alerts'. Below it is a search bar labeled 'Search (Ctrl+ /)'. The main area is divided into two sections: 'EXPLORE' on the left and 'MANAGE' on the right. The 'EXPLORE' section contains links for 'Activity log', 'Metrics', 'Diagnostics logs', 'Log search', and 'Service notifications'. The 'MANAGE' section contains links for 'Alerts' (which is highlighted with a light blue background) and 'Action groups'. A vertical grey sidebar is visible on the right side of the blade.

4. Select the **Add activity log alert** command and fill in the fields
5. **Name** your activity log alert, and choose a **Description**. These will appear in the notifications sent when this alert fires.

Monitor - Alerts

Search (Ctrl+ /)

Columns

Add metric alert

+ Add activity log alert

6. The **Subscription** is the one the Activity log alert will be saved in. It will be auto filled to the subscription you are currently operating under. This is the subscription the alert resource will be deployed to and monitor.

Add activity log alert

* Activity log alert name: Sample Service Health Alert ✓

Description: Alerts whenever an incident post occurs for this subscription ✓

* Subscription: Azure LSM and Communications - Incubation (Test Env on...)

* Resource group: Default-Storage-EastUS

Source

Input: Activity log

Criteria

* Event category: Administrative

Resource group: Default-Storage-SoutheastAsia

Resource: matsappstoragetestsg1

Resource type: Storage Account (storageAccounts)

Operation name: Delete Storage Account (storageAccounts)

Level: All

Status: Started

Event initiated by:

Notify via

Action group: New Existing

* Action group: first action group

Note: Note that only the country code '1' is currently supported for SMS.
It can take up to 5 minutes for an Activity log alert to become active.

[Privacy Statement](#)

[Pricing](#)

OK

7. Choose the **Resource Group** this alert will be associated with in the **Subscription**.
8. Provide the **Event Category, Resource Group, Resource, Resource Type, Operation Name, Level, Status** and **Event initiated by** values to scope for what events this alert should apply.
9. Choose to **Notify Via** an **Existing action group**. Select the respective action group.
10. Select **OK** when done to create the alert.

Within a few minutes, the alert is active and triggers as previously described.

Managing your alerts

Once you have created an alert, it will be visible in the Alerts section of the Monitor service. Select the alert you

wish to manage, you will be able to:

- **Edit** it.
- **Delete** it.
- **Disable** or **Enable** it if you want to temporarily stop or resume receiving notifications for the alert.

Next Steps

- Get an [overview of alerts](#)
- Learn about [notification rate limiting](#)
- Review the [activity log alert webhook schema](#)
- Learn more about [action groups](#)
- Learn about [Service Health Notifications](#)
- [Create an Activity Log Alert to monitor all autoscale engine operations on your subscription.](#)
- [Create an Activity Log Alert to monitor all failed autoscale scale in/scale out operations on your subscription](#)

Archive the Azure Activity Log

7/18/2017 • 5 min to read • [Edit Online](#)

In this article, we show how you can use the Azure portal, PowerShell Cmdlets, or Cross-Platform CLI to archive your [Azure Activity Log](#) in a storage account. This option is useful if you would like to retain your Activity Log longer than 90 days (with full control over the retention policy) for audit, static analysis, or backup. If you only need to retain your events for 90 days or less you do not need to set up archival to a storage account, since Activity Log events are retained in the Azure platform for 90 days without enabling archival.

Prerequisites

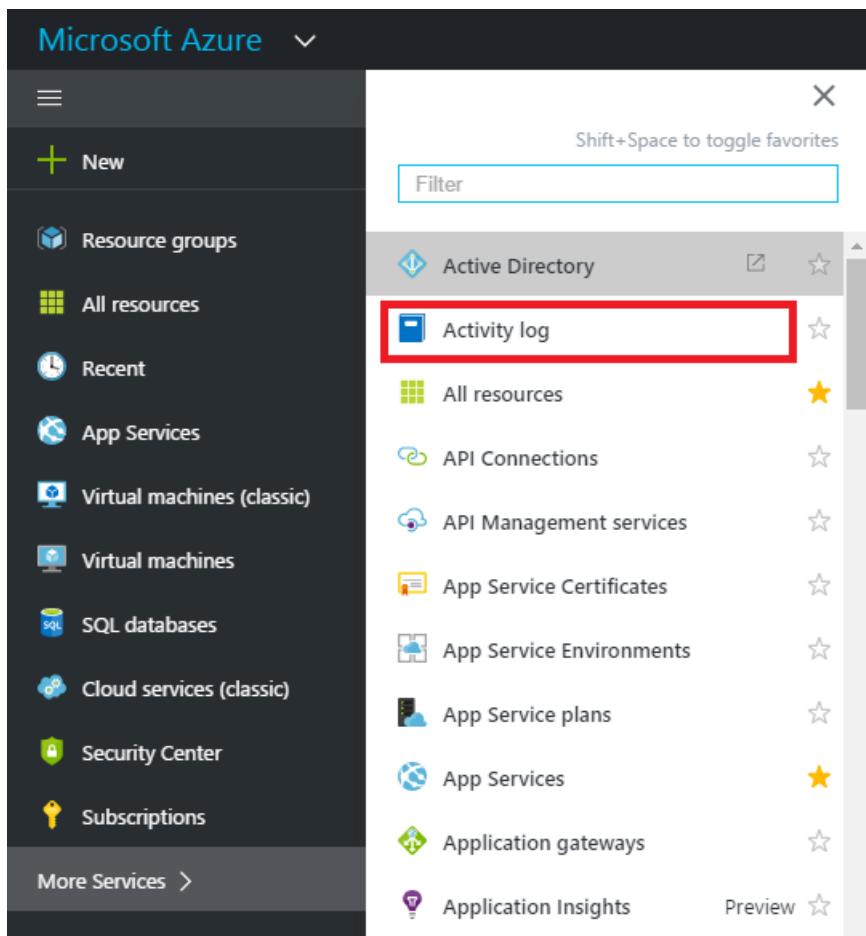
Before you begin, you need to [create a storage account](#) to which you can archive your Activity Log. We highly recommend that you do not use an existing storage account that has other, non-monitoring data stored in it so that you can better control access to monitoring data. However, if you are also archiving Diagnostic Logs and metrics to a storage account, it may make sense to use that storage account for your Activity Log as well to keep all monitoring data in a central location. The storage account you use must be a general purpose storage account, not a blob storage account. The storage account does not have to be in the same subscription as the subscription emitting logs as long as the user who configures the setting has appropriate RBAC access to both subscriptions.

Log Profile

To archive the Activity Log using any of the methods below, you set the **Log Profile** for a subscription. The Log Profile defines the type of events that are stored or streamed and the outputs—storage account and/or event hub. It also defines the retention policy (number of days to retain) for events stored in a storage account. If the retention policy is set to zero, events are stored indefinitely. Otherwise, this can be set to any value between 1 and 2147483647. Retention policies are applied per-day, so at the end of a day (UTC), logs from the day that is now beyond the retention policy will be deleted. For example, if you had a retention policy of one day, at the beginning of the day today the logs from the day before yesterday would be deleted. [You can read more about log profiles here.](#)

Archive the Activity Log using the portal

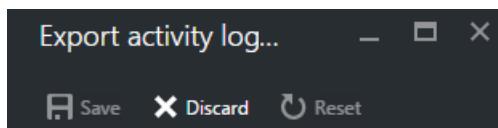
1. In the portal, click the **Activity Log** link on the left-side navigation. If you don't see a link for the Activity Log, click the **More Services** link first.



2. At the top of the blade, click **Export**.

A screenshot of the 'Activity log' blade. At the top, there's a toolbar with 'Columns' and an 'Export' button, which is highlighted with a red box. Below the toolbar are several filter options: 'Select query ...', 'Subscription' (set to 'Subscription1'), 'Resource group' (set to 'All resource groups'), 'Resource' (set to 'All resources'), 'Timespan' (set to 'Last 1 hour'), 'Event category' (set to 'All categories'), 'Event severity' (set to '4 selected'). At the bottom of the blade, it says 'Query returned 38 items. Click here to download all the items as csv.'

3. In the blade that appears, check the box for **Export to a storage account** and select a storage account.



4. Using the slider or text box, define a number of days for which Activity Log events should be kept in your storage account. If you prefer to have your data persisted in the storage account indefinitely, set this number to zero.
5. Click **Save**.

Archive the Activity Log via PowerShell

```
Add-AzureRmLogProfile -Name my_log_profile -StorageAccountId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/my_storage -Locations  
global,westus,eastus -RetentionInDays 180 -Categories Write,Delete,Action
```

| PROPERTY | REQUIRED | DESCRIPTION |
|------------------|----------|---|
| StorageAccountId | No | Resource ID of the Storage Account to which Activity Logs should be saved. |
| Locations | Yes | Comma-separated list of regions for which you would like to collect Activity Log events. You can view a list of all regions by visiting this page or by using the Azure Management REST API . |
| RetentionInDays | Yes | Number of days for which events should be retained, between 1 and 2147483647. A value of zero stores the logs indefinitely (forever). |
| Categories | Yes | Comma-separated list of event categories that should be collected. Possible values are Write, Delete, and Action. |

Archive the Activity Log via CLI

```
azure insights logprofile add --name my_log_profile --storageId /subscriptions/s1/resourceGroups/insights-integration/providers/Microsoft.Storage/storageAccounts/my_storage --locations global,westus,eastus,northeurope --retentionInDays 180 -categories Write,Delete,Action
```

| PROPERTY | REQUIRED | DESCRIPTION |
|-----------------|----------|---|
| name | Yes | Name of your log profile. |
| storageId | No | Resource ID of the Storage Account to which Activity Logs should be saved. |
| locations | Yes | Comma-separated list of regions for which you would like to collect Activity Log events. You can view a list of all regions by visiting this page or by using the Azure Management REST API . |
| retentionInDays | Yes | Number of days for which events should be retained, between 1 and 2147483647. A value of zero will store the logs indefinitely (forever). |
| categories | Yes | Comma-separated list of event categories that should be collected. Possible values are Write, Delete, and Action. |

Storage schema of the Activity Log

Once you have set up archival, a storage container will be created in the storage account as soon as an Activity Log event occurs. The blobs within the container follow the same format across the Activity Log and Diagnostic Logs. The structure of these blobs is:

```
insights-operational-logs/name=default/resourceId=/SUBSCRIPTIONS/{subscription ID}/y={four-digit numeric year}/m={two-digit numeric month}/d={two-digit numeric day}/h={two-digit 24-hour clock hour}/m=00/PT1H.json
```

For example, a blob name might be:

```
insights-operational-logs/name=default/resourceId=/SUBSCRIPTIONS/s1id1234-5679-0123-4567-890123456789/y=2016/m=08/d=22/h=18/m=00/PT1H.json
```

Each PT1H.json blob contains a JSON blob of events that occurred within the hour specified in the blob URL (e.g. h=12). During the present hour, events are appended to the PT1H.json file as they occur. The minute value (m=00) is always 00, since Activity Log events are broken into individual blobs per hour.

Within the PT1H.json file, each event is stored in the "records" array, following this format:

```
{
  "records": [
    {
      "time": "2015-01-21T22:14:26.9792776Z",
      "resourceId":
      "/subscriptions/s1/resourceGroups/MSSupportGroup/providers/microsoft.support/supporttickets/115012112305841",
      "operationName": "microsoft.support/supporttickets/write",
      "category": "Write",
      "resultType": "Success",
      "resultSignature": "Succeeded.Created",
      "durationMs": 2826,
      "callerIpAddress": "111.111.111.11",
      "correlationId": "c776f9f4-36e5-4e0e-809b-c9b3c3fb62a8",
      "identity": {
        "authorization": {
          "scope": "
        "/subscriptions/s1/resourceGroups/MSSupportGroup/providers/microsoft.support/supporttickets/115012112305841",
          "action": "microsoft.support/supporttickets/write",
          "evidence": {
            "role": "Subscription Admin"
          }
        },
        "claims": {
          "aud": "https://management.core.windows.net/",
          "iss": "https://sts.windows.net/72f988bf-86f1-41af-91ab-2d7cd011db47/",
          "iat": "1421876371",
          "nbf": "1421876371",
          "exp": "1421880271",
          "ver": "1.0",
          "http://schemas.microsoft.com/identity/claims/tenantid": "1e8d8218-c5e7-4578-9acc-9abbd5d23315",
          "http://schemas.microsoft.com/claims/authnmethodsreferences": "pwd",
          "http://schemas.microsoft.com/identity/claims/objectidentifier": "2468adf0-8211-44e3-95xq-85137af64708",
          "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn": "admin@contoso.com",
          "puid": "20030000801A118C",
          "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier": "
        "9vckmEGF7zDKk1YzIY8k0t1_EAPaXoeHyPRn6f413zM",
          "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname": "John",
          "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname": "Smith",
          "name": "John Smith",
          "groups": "cacfe77c-e058-4712-83qw-f9b08849fd60,7f71d11d-4c41-4b23-99d2-d32ce7aa621c,31522864-0578-4ea0-9gdc-e66cc564d18c",
          "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name": " admin@contoso.com",
          "appid": "c44b4083-3bq0-49c1-b47d-974e53cbdf3c",
          "appidacr": "2",
          "http://schemas.microsoft.com/identity/claims/scope": "user_impersonation",
          "http://schemas.microsoft.com/claims/authnclassreference": "1"
        }
      },
      "level": "Information",
      "location": "global",
      "properties": {
        "statusCode": "Created",
        "serviceRequestId": "50d5cddb-8ca0-47ad-9b80-6cde2207f97c"
      }
    }
  ]
}
```

| ELEMENT NAME | DESCRIPTION |
|--------------|---|
| time | Timestamp when the event was generated by the Azure service processing the request corresponding the event. |

| ELEMENT NAME | DESCRIPTION |
|-----------------|--|
| resourceId | Resource ID of the impacted resource. |
| operationName | Name of the operation. |
| category | Category of the action, eg. Write, Read, Action. |
| resultType | The type of the result, eg. Success, Failure, Start |
| resultSignature | Depends on the resource type. |
| durationMs | Duration of the operation in milliseconds |
| callerIpAddress | IP address of the user who has performed the operation, UPN claim, or SPN claim based on availability. |
| correlationId | Usually a GUID in the string format. Events that share a correlationId belong to the same uber action. |
| identity | JSON blob describing the authorization and claims. |
| authorization | Blob of RBAC properties of the event. Usually includes the "action", "role" and "scope" properties. |
| level | Level of the event. One of the following values: "Critical", "Error", "Warning", "Informational" and "Verbose" |
| location | Region in which the location occurred (or global). |
| properties | Set of <Key, Value> pairs (i.e. Dictionary) describing the details of the event. |

NOTE

The properties and usage of those properties can vary depending on the resource.

Next steps

- [Download blobs for analysis](#)
- [Stream the Activity Log to Event Hubs](#)
- [Read more about the Activity Log](#)

Stream the Azure Activity Log to Event Hubs

7/18/2017 • 3 min to read • [Edit Online](#)

The [Azure Activity Log](#) can be streamed in near real time to any application using the built-in "Export" option in the portal, or by enabling the Service Bus Rule Id in a Log Profile via the Azure PowerShell Cmdlets or Azure CLI.

What you can do with the Activity Log and Event Hubs

Here are just a few ways you might use the streaming capability for the Activity Log:

- **Stream to third-party logging and telemetry systems** – Over time, Event Hubs streaming will become the mechanism to pipe your Activity Log into third-party SIEMs and log analytics solutions.
- **Build a custom telemetry and logging platform** – If you already have a custom-built telemetry platform or are just thinking about building one, the highly scalable publish-subscribe nature of Event Hubs allows you to flexibly ingest the activity log. [See Dan Rosanova's guide to using Event Hubs in a global scale telemetry platform here.](#)

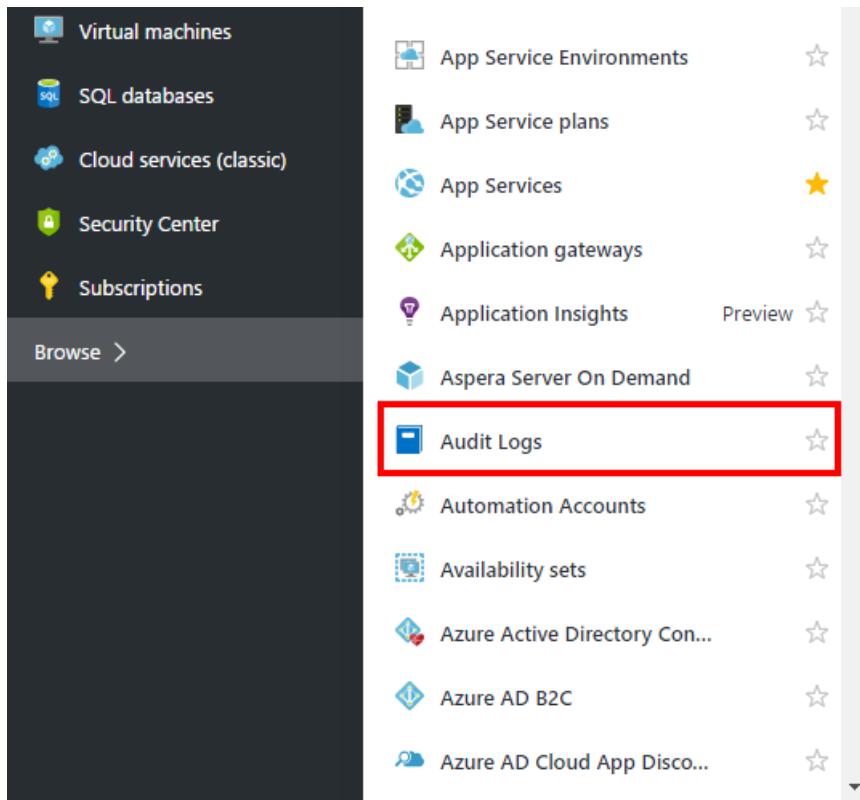
Enable streaming of the Activity Log

You can enable streaming of the Activity Log either programmatically or via the portal. Either way, you pick a Service Bus Namespace and a shared access policy for that namespace, and an Event Hub is created in that namespace when the first new Activity Log event occurs. If you do not have a Service Bus Namespace, you first need to create one. If you have previously streamed Activity Log events to this Service Bus Namespace, the Event Hub that was previously created will be reused. The shared access policy defines the permissions that the streaming mechanism has. Today, streaming to an Event Hubs requires **Manage**, **Send**, and **Listen** permissions. You can create or modify Service Bus Namespace shared access policies in the classic portal under the "Configure" tab for your Service Bus Namespace. To update the Activity Log log profile to include streaming, the user making the change must have the ListKey permission on that Service Bus Authorization Rule.

The service bus or event hub namespace does not have to be in the same subscription as the subscription emitting logs as long as the user who configures the setting has appropriate RBAC access to both subscriptions.

Via Azure portal

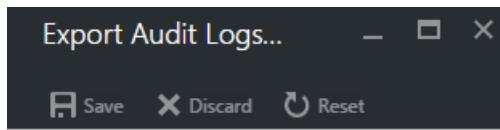
1. Navigate to the **Activity Log** blade using the menu on the left side of the portal.



2. Click the **Export** button at the top of the blade.

A screenshot of the 'Audit logs' blade. At the top, there's a header with 'Audit logs' and a 'Columns' button. Below the header is a red box around the 'Export' button. The blade contains several filter options: 'Select query ...' (with a dropdown arrow), 'Subscription' set to 'Monitoring vNext Test Environment Services', 'Resource group' set to 'All resource groups', 'Timespan' set to 'Last 1 hour', 'Event category' set to 'All categories', and 'Apply' and 'Reset' buttons at the bottom.

3. In the blade that appears, you can select the regions for which you would like to stream events and the Service Bus Namespace in which you would like an Event Hub to be created for streaming these events.



- Click **Save** to save these settings. The settings are immediately applied to your subscription.

Via PowerShell Cmdlets

If a log profile already exists, you first need to remove that profile.

- Use `Get-AzureRmLogProfile` to identify if a log profile exists
- If so, use `Remove-AzureRmLogProfile` to remove it.
- Use `Set-AzureRmLogProfile` to create a profile:

```
Add-AzureRmLogProfile -Name my_log_profile -serviceBusRuleId /subscriptions/s1/resourceGroups/Default-ServiceBus-EastUS/providers/Microsoft.ServiceBus/namespaces/mytestSB/authorizationrules/RootManageSharedAccessKey -Locations global,westus,eastus -RetentionInDays 90 -Categories Write,Delete,Action
```

The Service Bus Rule ID is a string with this format: {service bus resource ID}/authorizationrules/{key name}, for example

Via Azure CLI

If a log profile already exists, you first need to remove that profile.

- Use `azure insights logprofile list` to identify if a log profile exists
- If so, use `azure insights logprofile delete` to remove it.
- Use `azure insights logprofile add` to create a profile:

```
azure insights logprofile add --name my_log_profile --storageId /subscriptions/s1/resourceGroups/insights-integration/providers/Microsoft.Storage/storageAccounts/my_storage --serviceBusRuleId /subscriptions/s1/resourceGroups/Default-ServiceBus-EastUS/providers/Microsoft.ServiceBus/namespaces/mytestSB/authorizationrules/RootManageSharedAccessKey --locations global,westus,eastus,northeurope --retentionInDays 90 --categories Write,Delete,Action
```

The Service Bus Rule ID is a string with this format: {service bus resource ID}/authorizationrules/{key name}.

How do I consume the log data from Event Hubs?

The schema for the Activity Log is available [here](#). Each event is in an array of JSON blobs called "records."

Next Steps

- [Archive the Activity Log to a storage account](#)
- [Read the overview of the Azure Activity Log](#)
- [Set up an alert based on an Activity Log event](#)

View activity logs to audit actions on resources

6/27/2017 • 3 min to read • [Edit Online](#)

Through activity logs, you can determine:

- what operations were taken on the resources in your subscription
- who initiated the operation (although operations initiated by a backend service do not return a user as the caller)
- when the operation occurred
- the status of the operation
- the values of other properties that might help you research the operation

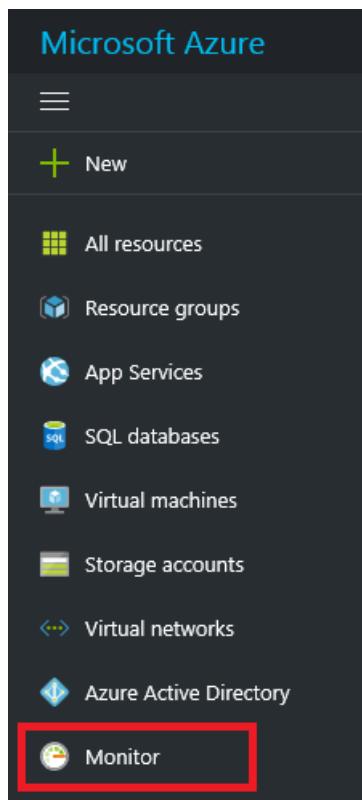
The activity log contains all write operations (PUT, POST, DELETE) performed on your resources. It does not include read operations (GET). You can use the audit logs to find an error when troubleshooting or to monitor how a user in your organization modified a resource.

Activity logs are retained for 90 days. You can query for any range of dates, as long as the starting date is not more than 90 days in the past.

You can retrieve information from the activity logs through the portal, PowerShell, Azure CLI, Insights REST API, or [Insights .NET Library](#).

Portal

1. To view the activity logs through the portal, select **Monitor**.



Or, to automatically filter the activity log for a particular resource or resource group, select **Activity log** from that resource blade. Notice that the activity log is automatically filtered by the selected resource.

Storage account

Columns Export

Search (Ctrl+ /)

Overview

Activity log

Access control (IAM)

Select query ...

* Subscription: Windows Azure MSDN

Resource group: demo-group

Resource: storagedemo

Timespan: Last 1 hour

Event category: All categories

* Event severity: 4 selected

Apply Reset

Insights (Last 24 hours): 1 failed deployment

2. In the **Activity Log** blade, you see a summary of recent operations.

Activity log

Columns Export

Select query ...

* Subscription: Windows Azure MSDN - Visual...

Resource group: All resource groups

Resource: All resources

Timespan: Last 1 hour

Event category: All categories

* Event severity: 4 selected

Apply Reset

Query returned 5 items. Click here to download all the items as csv.

| OPERATION NAME | STATUS | TIME | TIME STAMP |
|----------------------------|-----------|-----------|---------------|
| ▶ ❌ Write Deployments | Failed | Just now | Mon Aug 22... |
| ℹ️ Validate | Succeeded | Just now | Mon Aug 22... |
| ℹ️ Register | Succeeded | 3 min ago | Mon Aug 22... |
| ▶ ℹ️ Delete resource group | Succeeded | 3 min ago | Mon Aug 22... |
| ℹ️ Register | Succeeded | 3 min ago | Mon Aug 22... |

3. To restrict the number of operations displayed, select different conditions. For example, the following image shows the **Timespan** and **Event initiated by** fields changed to view the actions taken by a particular user or application for the past month. Select **Apply** to view the results of your query.

Activity log

Columns Export

Select query ...

* Subscription: Windows Azure MSDN

Resource group: All resource groups

Resource type: All resource types

Timespan: Last month

Event category: All categories

* Event severity: 4 selected

Event initiated by: CloudSense

Apply Reset

Insights (Last 24 hours): 1 failed deployment

4. If you need to run the query again later, select **Save** and give the query a name.

Activity log

Columns Export

Select query ...  

5. To quickly run a query, you can select one of the built-in queries, such as failed deployments.

Insights (Last 24 hours): **1 failed deployment** | 0 role assignments | 1 error

The selected query automatically sets the required filter values.

Select query ...    Insights (Last 24 hours): **1 failed deployment** | 0 role assignments |

* Subscription  Resource group  Resource  Resource type 
2 selected All resource groups All resources Deployment (deployments)
Timespan  Event category  Event severity  Event initiated by 
Last 24 hours All categories Error All
Apply Reset

Query returned 1 items. [Click here to download all the items as csv.](#)

| OPERATION NAME | STATUS | TIME | TIME STAMP | SUBSCR |
|----------------|--------|-----------|-----------------|---------|
| ! Validate | Failed | 4 min ago | Mon Jan 09 2... | Third I |

6. Select one of the operations to see a summary of the event.

OPERATION NAME

! Validate

Validate

Summary JSON

Operation name
Validate

Time stamp
Mon Jan 09 2017 14:14:39 GMT-0800 (Pacific Standard Time)

Event initiated by

Error code
KeyVaultParameterReferenceNotFound

Message
The specified KeyVault '/subscriptions/ ... could not be found.

PowerShell

1. To retrieve log entries, run the **Get-AzureRmLog** command. You provide additional parameters to filter the list of entries. If you do not specify a start and end time, entries for the last hour are returned. For example, to retrieve the operations for a resource group during the past hour run:

```
Get-AzureRmLog -ResourceGroup ExampleGroup
```

The following example shows how to use the activity log to research operations taken during a specified time. The start and end dates are specified in a date format.

```
Get-AzureRmLog -ResourceGroup ExampleGroup -StartTime 2015-08-28T06:00 -EndTime 2015-09-10T06:00
```

Or, you can use date functions to specify the date range, such as the last 14 days.

```
Get-AzureRmLog -ResourceGroup ExampleGroup -StartTime (Get-Date).AddDays(-14)
```

2. Depending on the start time you specify, the previous commands can return a long list of operations for the resource group. You can filter the results for what you are looking for by providing search criteria. For example, if you are trying to research how a web app was stopped, you could run the following command:

```
Get-AzureRmLog -ResourceGroup ExampleGroup -StartTime (Get-Date).AddDays(-14) | Where-Object  
OperationName -eq Microsoft.Web/sites/stop/action
```

Which for this example shows that a stop action was performed by someone@contoso.com.

```
Authorization      :  
Scope      : /subscriptions/xxxxx/resourcegroups/ExampleGroup/providers/Microsoft.Web/sites/ExampleSite  
Action     : Microsoft.Web/sites/stop/action  
Role       : Subscription Admin  
Condition   :  
Caller      : someone@contoso.com  
CorrelationId  : 84beae59-92aa-4662-a6fc-b6fecc0ff8da  
EventSource    : Administrative  
EventTimestamp : 8/28/2015 4:08:18 PM  
OperationName  : Microsoft.Web/sites/stop/action  
ResourceGroupName : ExampleGroup  
ResourceId     :  
/subscriptions/xxxxx/resourcegroups/ExampleGroup/providers/Microsoft.Web/sites/ExampleSite  
Status       : Succeeded  
SubscriptionId : xxxx  
SubStatus     : OK
```

3. You can look up the actions taken by a particular user, even for a resource group that no longer exists.

```
Get-AzureRmLog -ResourceGroup deletedgroup -StartTime (Get-Date).AddDays(-14) -Caller  
someone@contoso.com
```

4. You can filter for failed operations.

```
Get-AzureRmLog -ResourceGroup ExampleGroup -Status Failed
```

5. You can focus on one error by looking at the status message for that entry.

```
((Get-AzureRmLog -Status Failed -ResourceGroup ExampleGroup -  
DetailedOutput).Properties[1].Content["statusMessage"] | ConvertFrom-Json).error
```

Which returns:

```
code          message
----          -----
DnsRecordInUse DNS record dns.westus.cloudapp.azure.com is already used by another public IP.
```

Azure CLI

- To retrieve log entries, you run the **azure group log show** command.

```
azure group log show ExampleGroup --json
```

REST API

The REST operations for working with the activity log are part of the [Insights REST API](#). To retrieve activity log events, see [List the management events in a subscription](#).

Next steps

- Azure Activity logs can be used with Power BI to gain greater insights about the actions in your subscription. See [View and analyze Azure Activity Logs in Power BI and more](#).
- To learn about setting security policies, see [Azure Role-based Access Control](#).
- To learn about the commands for viewing deployment operations, see [View deployment operations](#).
- To learn how to prevent deletions on a resource for all users, see [Lock resources with Azure Resource Manager](#).

Create an activity log alert with a Resource Manager Template

7/18/2017 • 1 min to read • [Edit Online](#)

This article shows how you can use an [Azure Resource Manager template](#) to configure activity log alerts. This enables you to automatically set up alerts on your resources when they are created as part of your automated deployment process.

The basic steps are as follows:

1. Create a template as a JSON file that describes how to create the activity log alert.
2. [Deploy the template using any deployment method.](#)

Resource Manager template for an activity log alert

To create an activity log alert using a Resource Manager template, you create a resource of type `microsoft.insights/activityLogAlerts` and fill in all related properties. Below is a template that creates an activity log alert.

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "activityLogAlertName": {
      "type": "string",
      "metadata": {
        "description": "Unique name (within the Resource Group) for the Activity log alert."
      }
    },
    "activityLogAlertEnabled": {
      "type": "bool",
      "defaultValue": true,
      "metadata": {
        "description": "Indicates whether or not the alert is enabled."
      }
    },
    "actionGroupResourceId": {
      "type": "string",
      "metadata": {
        "description": "Resource Id for the Action group."
      }
    }
  },
  "resources": [
    {
      "type": "Microsoft.Insights/activityLogAlerts",
      "apiVersion": "2017-04-01",
      "name": "[parameters('activityLogAlertName')]",
      "location": "Global",
      "properties": {
        "enabled": "[parameters('activityLogAlertEnabled')]",
        "scopes": [
          "[subscription().id]"
        ],
        "condition": {
          "allOf": [
            {
              "field": "category",
              "equals": "Administrative"
            },
            {
              "field": "operationName",
              "equals": "Microsoft.Resources/deployments/write"
            },
            {
              "field": "resourceType",
              "equals": "Microsoft.Resources/deployments"
            }
          ]
        },
        "actions": {
          "actionGroups": [
            {
              "actionGroupId": "[parameters('actionGroupResourceId')]"
            }
          ]
        }
      }
    }
  ]
}
```

You can also [visit our Quickstart Gallery](#) for some examples of Activity Log alert templates.

Next Steps

- Learn more about [Alerts](#)
- How to add [action groups](#) using a Resource Manager template
- [Create an Activity Log Alert to monitor all autoscale engine operations on your subscription.](#)
- [Create an Activity Log Alert to monitor all failed autoscale scale in/scale out operations on your subscription](#)

Service Health Notifications

7/18/2017 • 3 min to read • [Edit Online](#)

Overview

This article shows you how to view service health notifications using the Azure portal.

Service health notifications allow you to view service health messages published by the Azure team that may be affecting the resources under your subscription. These notifications are a sub-class of activity log events and can also be found on the activity log blade. Service health notifications can be informational or actionable depending on the class.

There are five classes of service health notifications:

- **Action Required:** From time to time we may notice something unusual happen on your account. We may need to work with you to remedy this. We will send you a notification either detailing the actions you will need to take or with details on how to contact Azure engineering or support.
- **Assisted Recovery:** An event has occurred and engineers have confirmed that you are still experiencing impact. Engineering will need to work with you directly to bring your services to restoration.
- **Incident:** A service impacting event is currently affecting one or more of the resources in your subscription.
- **Maintenance:** This is a notification informing you of a planned maintenance activity that may impact one or more of the resources under your subscription.
- **Information:** From time to time we may send you notifications that communicate to you about potential optimizations that may help improve your resource utilization.
- **Security:** Urgent security related information regarding your solution(s) running on Azure.

Each service health notification will carry details on the scope and impact to your resources. Details will include:

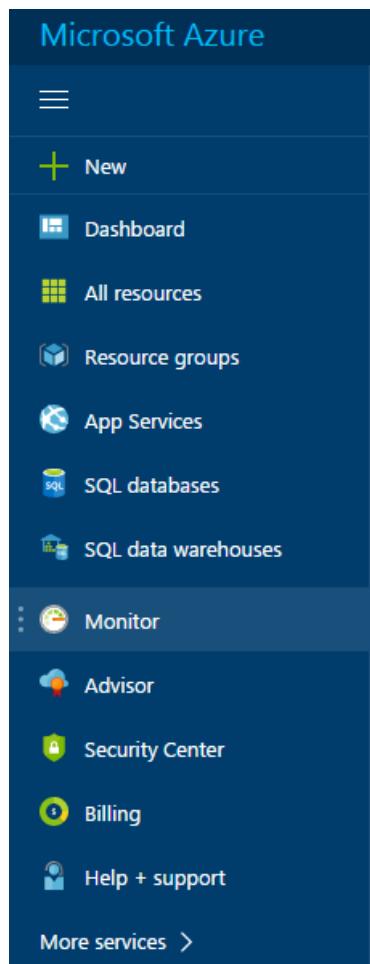
| PROPERTY NAME | DESCRIPTION |
|----------------------|---|
| channels | Is one of the following values: "Admin", "Operation" |
| correlationId | Is usually a GUID in the string format. Events with that belong to the same uber action usually share the same correlationId. |
| eventDataId | Is the unique identifier of an event |
| eventName | Is the title of the event |
| level | Level of the event. One of the following values: "Critical", "Error", "Warning", "Informational" and "Verbose" |
| resourceProviderName | Name of the resource provider for the impacted resource |
| resourceType | The type of resource of the impacted resource |

| PROPERTY NAME | DESCRIPTION |
|-----------------------------------|--|
| subStatus | Usually the HTTP status code of the corresponding REST call, but can also include other strings describing a substatus, such as these common values: OK (HTTP Status Code: 200), Created (HTTP Status Code: 201), Accepted (HTTP Status Code: 202), No Content (HTTP Status Code: 204), Bad Request (HTTP Status Code: 400), Not Found (HTTP Status Code: 404), Conflict (HTTP Status Code: 409), Internal Server Error (HTTP Status Code: 500), Service Unavailable (HTTP Status Code: 503), Gateway Timeout (HTTP Status Code: 504). |
| eventTimestamp | Timestamp when the event was generated by the Azure service processing the request corresponding the event. |
| submissionTimestamp | Timestamp when the event became available for querying. |
| subscriptionId | The Azure subscription in which this event was logged |
| status | String describing the status of the operation. Some common values are: Started, In Progress, Succeeded, Failed, Active, Resolved. |
| operationName | Name of the operation. |
| category | "ServiceHealth" |
| resourceId | Resource id of the impacted resource. |
| Properties.title | The localized title for this communication. English is the default language. |
| Properties.communication | The localized details of the communication with HTML markup. English is the default. |
| Properties.incidentType | Possible values: AssistedRecovery, ActionRequired, Information, Incident, Maintenance, Security |
| Properties.trackingId | Identifies the incident this event is associated with. Use this to correlate the events related to an incident. |
| Properties.impactedServices | An escaped JSON blob which describes the services and regions that are impacted by the incident. A list of Services, each of which has a ServiceName and a list of ImpactedRegions, each of which has a RegionName. |
| Properties.defaultLanguageTitle | The communication in English |
| Properties.defaultLanguageContent | The communication in English as either html markup or plain text |
| Properties.stage | Possible values for AssistedRecovery, ActionRequired, Information, Incident, Security: are Active, Resolved. For Maintenance they are: Active, Planned, InProgress, Canceled, Rescheduled, Resolved, Complete |

| PROPERTY NAME | DESCRIPTION |
|----------------------------|---|
| Properties.communicationId | The communication this event is associated. |

Viewing your service health notifications in the Azure portal

1. In the [portal](#), navigate to the **Monitor** service



2. Click the **Monitor** option to open up the Monitor blade. This blade brings together all your monitoring settings and data into one consolidated view. It first opens to the **Activity log** section.
3. Now click on **Service Notifications** section

The screenshot shows the Azure portal interface. On the left, there's a sidebar with navigation links: EXPLORE (Activity log, Metrics, Diagnostics logs, Log search, Service notifications), MANAGE (Alerts, Action groups), HEALTH (Resource health), and ADVANCED (Application Insights, Management solutions, Network watcher). The main area displays a list of service health notifications. One notification, titled 'Republish Test Title' and status 'Active', is highlighted with a blue background. Below this, there's a summary card for the notification.

| Notification Details | Status |
|--|----------------|
| Resolved: App Service \ Web Apps Recovered | Service Health |
| Resolved: App Service \ Web Apps Recovered | Service Health |
| Resolved: App Service \ Web Apps Recovered | Service Health |
| Resolved: App Service \ Web Apps Recovered | Service Health |
| Resolved: App Service \ Web Apps Recovered | Service Health |
| Republish Test Title | Service Health |
| Active: App Service \ Web Apps Recovered | Service Health |
| Active: App Service \ Web Apps Recovered | Service Health |
| Active: App Service \ Web Apps Recovered | Service Health |
| Active: App Service \ Web Apps Recovered | Service Health |

Republish Test Title

+ Add activity log alert

Summary JSON

Title: Republish Test Title

Status: Active

Incident type: Maintenance

Communication: Active: Republish Test Message

Service: Virtual Machines

Region: Global

4. Click on any of the line items to view more details
5. Click on the **+Add Activity Log Alert** operation to receive notifications to ensure you are notified for future service notifications of this type. To learn more on configuring alerts on service notifications [click here](#)

Next Steps:

Receive [alert notifications whenever a service health notification](#) is posted
Learn more about [activity log alerts](#)

Create Activity Log Alerts on Service Notifications

7/18/2017 • 3 min to read • [Edit Online](#)

Overview

This article shows you how to set up activity log alerts for service health notifications using the Azure portal.

You can receive an alert when Azure sends service health notifications to your Azure subscription.

You can configure the alert based on:

- The class of service health notification (Incident, Maintenance, Information, etc.)
- The service(s) affected
- The region(s) affected
- The status of the notification (Active vs. Resolved)
- The level of the notifications (Informational, Warning, Error)

You can also the configure who the alert should be sent to:

- Select an existing Action Group
- Create a new Action Group (that can be later used for future alerts)

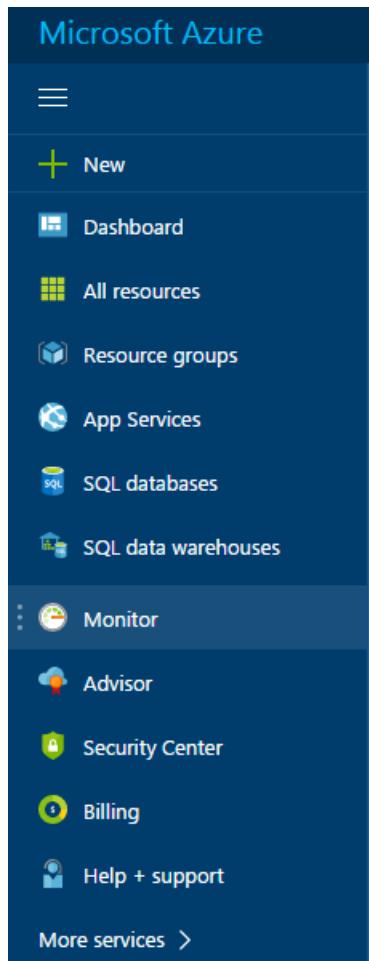
You can learn more about Action Groups [here](#)

For information on configuring service health notification alerts using Azure Resource Manager templates:

[Resource Manager templates](#)

Create an alert on a service health notification for a new action group with the Azure Portal

1. In the [portal](#), navigate to the **Monitor** service



2. Click the **Monitor** option to open the Monitor blade. It first opens to the **Activity log** section.

3. Now click on the **Alerts** section.

This screenshot of the 'Monitor - Alerts' blade shows the 'Alerts' section selected under the 'MANAGE' category. The 'Alerts' item has a yellow background and a green exclamation mark icon. Other options like 'Action groups' and 'Autoscale' are also listed. On the left, there's an 'EXPLORE' sidebar with links to 'Activity log', 'Metrics', 'Diagnostics logs', and 'Log search'. A search bar at the top is labeled 'Search (Ctrl+ /)'.

4. Select the **Add activity log alert** command and fill in the fields

This screenshot of the 'Monitor - Alerts' blade shows the 'Add activity log alert' button highlighted with a red box. The button has a plus sign icon and the text 'Add activity log alert'. The rest of the interface includes a search bar, column headers, and other standard blade controls.

5. **Name** your activity log alert, and provide a **Description**.

Add activity log alert

| * Activity log alert name <small>1</small> | Sample Service Health Alert | <input checked="" type="checkbox"/> |
|--|---|--|
| Description <small>1</small> | Alerts whenever an incident post occurs for this subscription | <input checked="" type="checkbox"/> |
| * Subscription <small>1</small> | Duke's MSDN Subscription | <input type="button" value="▼"/> |
| * Resource group <small>1</small> | Default-ActivityLogAlerts | <input type="button" value="▼"/> |
| Source | | |
| Input <small>1</small> Activity log | | |
| Criteria | | |
| * Event category <small>1</small> | Service Health | |
| * Service(s) <small>1</small> | 123 selected | |
| * Region(s) <small>1</small> | 30 selected | |
| Type <small>1</small> | All | |
| Status <small>1</small> | All | |
| Level <small>1</small> | All | |
| Alert via | | |
| Action group | <input checked="" type="radio"/> New | <input type="radio"/> Existing |
| * Action group name <small>1</small> | <input type="text"/> | |
| * Short name <small>1</small> | <input type="text"/> | |
| Actions | | |
| NAME | ACTION TYPE | DETAILS |
| SMS-Action | SMS | 555-555-5555 <input type="button" value="..."/> |
| Email-Action | Email | sample@contoso.com <input type="button" value="..."/> |
| Webhook-Action | Webhook | <input checked="" type="checkbox"/> https://sample.webhoc <input type="button" value="..."/> |
| <i>The name of the receiver</i> | SMS | <i>Depending on the acti...</i> <input type="button" value="..."/> |
|  Note that only the country code '1' is currently supported for SMS.
It can take up to 5 minutes for an Activity log alert to become active. | | |
| Privacy Statement
Pricing | | |

OK

6. The **Subscription** is the one the Activity log alert will be saved in. It will be auto filled to the subscription you are currently operating under. This is the subscription the alert resource will be deployed to and monitor.

7. Choose the **Resource Group** this alert will be associated with in the **Subscription**.

8. Under **Event Category** select the 'Service Health' option. Choose for what **Service, Region, Type, Status** and **Level** of Service Health Notifications you would like to be notified.

9. Create a **New Action Group** by giving it **Name** and **Short Name**; the Short Name will be referenced in the notifications sent when this alert fires.

10. Then, define a list of receivers by providing the receiver's

a. **Name:** Receiver's name, alias or identifier.

b. **Action Type:** Choose to contact the receiver via SMS, Email, or Webhook

c. **Details:** Based on the action type chosen, provide a phone number, email address or webhook URI.

11. Select **OK** when done to create the alert.

Within a few minutes, the alert is active and triggers as previously described.

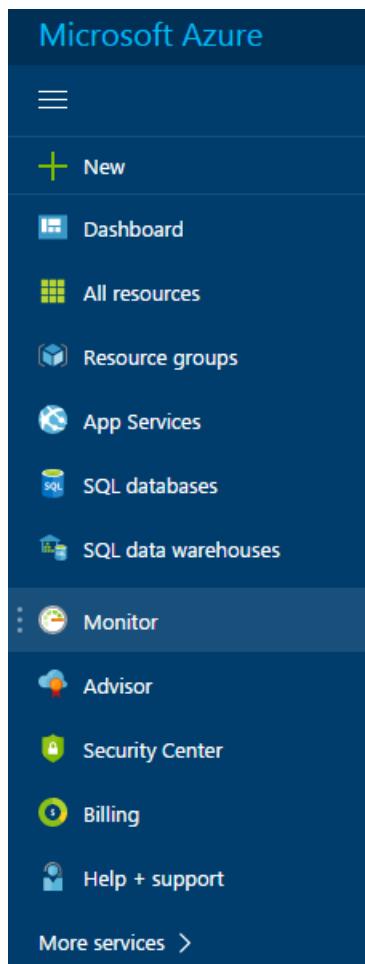
For details on the webhook schema for activity log alerts [click here](#)

NOTE

The action group defined in these steps will be reusable, as an existing action group, for all future alert definitions.

Create an alert on a service health notification using an existing action group with the Azure Portal

1. In the [portal](#), navigate to the **Monitor** service



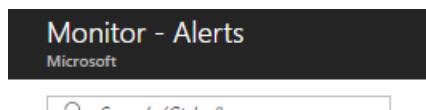
2. Click the **Monitor** option to open the Monitor blade. It first opens to the **Activity log** section.

3. Now click on the **Alerts** section.

Monitor - Alerts

Microsoft

Search (Ctrl+ /)



EXPLORE

Activity log

Metrics

Diagnostics logs

Log search

MANAGE

Alerts

Action groups

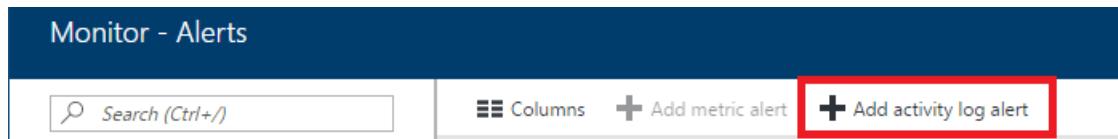
Autoscale

4. Select the **Add activity log alert** command and fill in the fields

Monitor - Alerts

Search (Ctrl+ /)

Columns Add metric alert **Add activity log alert**



5. **Name** your activity log alert, and choose a **Description**.

Add activity log alert

□ X

* Activity log alert name ⓘ ✓

Description ⓘ ✓

* Subscription ⓘ

* Resource group ⓘ

Source

Input ⓘ Activity log

Criteria

* Event category ⓘ

* Service(s) ⓘ

* Region(s) ⓘ

Type ⓘ

Status ⓘ

Level ⓘ

Alert via

Action group New Existing

* Subscription ⓘ

* Action group ⓘ



Note that only the country code '1' is currently supported for SMS.

It can take up to 5 minutes for an Activity log alert to become active.

[Privacy Statement](#)

[Pricing](#)

OK

6. The **Subscription** is the one the Activity log alert will be saved in. It will be auto filled to the subscription you are currently operating under. This is the subscription the alert resource will be deployed to and monitor.
7. Choose the **Resource Group** this alert will be associated with in the **Subscription**.
8. Under **Event Category** select the 'Service Health' option. Choose for what **Service, Region, Type, Status** and **Level** of Service Health Notifications you would like to be notified.
9. Choose to **Notify Via** an **Existing action group**. Select the appropriate action group.
10. Select **OK** when done to create the alert.

Within a few minutes, the alert is active and triggers as previously described.

Managing your alerts

Once you have created an alert, it will be visible in the Alerts section of the Monitor service. Select the alert you wish to manage, you will be able to:

- **Edit** it.
- **Delete** it.
- **Disable** or **Enable** it if you want to temporarily stop or resume receiving notifications for the alert.

Next Steps:

- Learn about [Service Health Notifications](#)
- Learn about [notification rate limiting](#)
- Review the [activity log alert webhook schema](#)
- Get an [overview of activity log alerts](#) and learn how to get alerted
- Learn more about [action groups](#)

Webhooks for Azure Activity Log alerts

7/31/2017 • 3 min to read • [Edit Online](#)

As part of the definition of an Action Group you are able to configure Webhook endpoints to receive Activity Log Alert notifications. Webhooks allow you to route these notifications to other systems for post-processing or custom actions. This article shows what the payload for the HTTP POST to a webhook looks like.

For information on creating Azure Activity Log alerts, [see this page instead](#).

For information on creating Action Groups, [see this page instead](#)

Authenticating the webhook

The webhook can authenticate using token-based authorization - The webhook URI is saved with a token ID, for example, `https://mysamplealert/webcallback?tokenid=sometokenid&someparameter=somevalue`

Payload schema

The JSON payload contained in the POST operation differs based on the payload's `data.context.activityLog.eventSource` field.

Common

```
{  
    "schemaId": "Microsoft.Insights/activityLogs",  
    "data": {  
        "status": "Activated",  
        "context": {  
            "activityLog": {  
                "channels": "Operation",  
                "correlationId": "6ac88262-43be-4adf-a11c-bd2179852898",  
                "eventSource": "Administrative",  
                "eventTimestamp": "2017-03-29T15:43:08.0019532+00:00",  
                "eventId": "8195a56a-85de-4663-943e-1a2bf401ad94",  
                "level": "Informational",  
                "operationName": "Microsoft.Insights/actionGroups/write",  
                "operationId": "6ac88262-43be-4adf-a11c-bd2179852898",  
                "status": "Started",  
                "subStatus": "",  
                "subscriptionId": "52c65f65-0518-4d37-9719-7dbbfcc68c57a",  
                "submissionTimestamp": "2017-03-29T15:43:20.3863637+00:00",  
                ...  
            }  
        },  
        "properties": {}  
    }  
}
```

Administrative

```
{
  "schemaId": "Microsoft.Insights/activityLogs",
  "data": {
    "status": "Activated",
    "context": {
      "activityLog": {
        "authorization": {
          "action": "Microsoft.Insights/actionGroups/write",
          "scope": "/subscriptions/52c65f65-0518-4d37-9719-7dbbf68c57b/resourceGroups/CONTOSO-TEST/providers/Microsoft.Insights/actionGroups/IncidentActions"
        },
        "claims": "...",
        "caller": "me@contoso.com",
        "description": "",
        "httpRequest": "...",
        "resourceId": "/subscriptions/52c65f65-0518-4d37-9719-7dbbf68c57b/resourceGroups/CONTOSO-TEST/providers/Microsoft.Insights/actionGroups/IncidentActions",
        "resourceGroupName": "CONTOSO-TEST",
        "resourceProviderName": "Microsoft.Insights",
        "resourceType": "Microsoft.Insights/actionGroups"
      }
    },
    "properties": {}
  }
}
```

ServiceHealth

```
{
  "schemaId": "unknown",
  "data": {
    "status": "Activated",
    "context": {
      "activityLog": {
        "properties": {
          "title": "...",
          "service": "...",
          "region": "...",
          "communication": "...",
          "incidentType": "Incident",
          "trackingId": "...",
          "groupId": "...",
          "impactStartTime": "3/29/2017 3:43:21 PM",
          "impactMitigationTime": "3/29/2017 3:43:21 PM",
          "eventCreationTime": "3/29/2017 3:43:21 PM",
          "impactedServices": "[...]",
          "defaultLanguageTitle": "...",
          "defaultLanguageContent": "...",
          "stage": "Active",
          "communicationId": "...",
          "version": "0.1"
        }
      }
    },
    "properties": {}
  }
}
```

For specific schema details on Service Health Notification activity log alerts [click here](#) For specific schema details on all other Activity Log alerts [click here](#)

| ELEMENT NAME | DESCRIPTION |
|----------------------|---|
| status | Used for metric alerts. Always set to "activated" for Activity Log alerts. |
| context | Context of the event. |
| resourceProviderName | The resource provider of the impacted resource. |
| conditionType | Always "Event." |
| name | Name of the alert rule. |
| id | Resource ID of the alert. |
| description | Alert description as set during creation of the alert. |
| subscriptionId | Azure Subscription ID. |
| timestamp | Time at which the event was generated by the Azure service that processed the request. |
| resourceId | Resource ID of the impacted resource. |
| resourceGroupName | Name of the resource group for the impacted resource |
| properties | Set of <Key, Value> pairs (that is, Dictionary<String, String>) that includes details about the event. |
| event | Element containing metadata about the event. |
| authorization | The RBAC properties of the event. These properties usually include the "action", "role" and the "scope." |
| category | Category of the event. Supported values include: Administrative, Alert, Security, ServiceHealth, Recommendation. |
| caller | Email address of the user who performed the operation, UPN claim, or SPN claim based on availability. Can be null for certain system calls. |
| correlationId | Usually a GUID in string format. Events with correlationId belong to the same larger action and usually share a correlationId. |
| eventDescription | Static text description of the event. |
| eventDataId | Unique identifier for the event. |
| eventSource | Name of the Azure service or infrastructure that generated the event. |

| ELEMENT NAME | DESCRIPTION |
|---------------|---|
| httpRequest | The request usually includes the "clientRequestId", "clientIpAddress" and HTTP "method" (for example, PUT). |
| level | One of the following values: "Critical", "Error", "Warning", "Informational" and "Verbose." |
| operationId | Usually a GUID shared among the events corresponding to single operation. |
| operationName | Name of the operation. |
| properties | Properties of the event. |
| status | String. Status of the operation. Common values include: "Started", "In Progress", "Succeeded", "Failed", "Active", "Resolved". |
| subStatus | Usually includes the HTTP status code of the corresponding REST call. It might also include other strings describing a substatus. Common substatus values include: OK (HTTP Status Code: 200), Created (HTTP Status Code: 201), Accepted (HTTP Status Code: 202), No Content (HTTP Status Code: 204), Bad Request (HTTP Status Code: 400), Not Found (HTTP Status Code: 404), Conflict (HTTP Status Code: 409), Internal Server Error (HTTP Status Code: 500), Service Unavailable (HTTP Status Code: 503), Gateway Timeout (HTTP Status Code: 504) |

Next steps

- [Learn more about the Activity Log](#)
- [Execute Azure Automation scripts \(Runbooks\) on Azure alerts](#)
- [Use Logic App to send an SMS via Twilio from an Azure alert](#). This example is for metric alerts, but could be modified to work with an Activity Log alert.
- [Use Logic App to send a Slack message from an Azure alert](#). This example is for metric alerts, but could be modified to work with an Activity Log alert.
- [Use Logic App to send a message to an Azure Queue from an Azure alert](#). This example is for metric alerts, but could be modified to work with an Activity Log alert.

SMS Alert Behavior in Action Groups

7/18/2017 • 2 min to read • [Edit Online](#)

Overview

Action groups enable you to configure a list of receivers. These groups can then be leveraged when defining activity log alerts; ensuring that a particular action group is notified when the activity log alert is triggered. One of the alerting mechanisms supported is SMS; the alerts support bi-directional communication. A user can respond to an alert to:

- **Unsubscribe from alerts:** A user can unsubscribe from all SMS alerts for all action groups, or a singular action group.
- **Resubscribe to alerts:** A user can resubscribe to all SMS alerts for all action groups, or a singular action group.
- **Request help:** A user can ask for more information on the SMS. They will be redirected to this article

This article covers the behavior of the SMS alerts and the response actions the user can take based on the locale of the user:

USA/Canada SMS behavior

Receiving an SMS Alert

An SMS receiver, who is configured as part of an action group, will receive an SMS when an alert fires. The SMS will carry the following information:

- Shortname of the action group this alert was sent to
- Title of the alert

Unsubscribing from SMS alerts for one action group

A user can unsubscribe from SMS for alerts for one action group by responding to the shortcode 20873 with the keywords: "DISABLE <Shortname of action group>" .

Ex. A user wishing to unsubscribe from alerts for an action group with the shortname "Azure", would send an SMS to the shortcode 20873 that says "DISABLE Azure"

Unsubscribing from SMS alerts for all action groups

A user can unsubscribe from all SMS alerts for all action groups by responding to the shortcode 20873 with any of the following keywords:

- STOP

Ex. A user wishing to unsubscribe from all SMS alerts for all action groups, would send an SMS to the shortcode 20873 that says "STOP"

NOTE

If a user has unsubscribed from SMS alerts, but is then added to a new action group; they WILL receive SMS alerts for that new action group, but remain unsubscribed from all previous action groups.

Resubscribing to SMS alerts for one action group

A user can resubscribe to SMS for alerts for one action group by responding to the shortcode 20873 with the keywords: "ENABLE <Shortname of action group>" .

Ex. A user wishing to resubscribe to alerts for an action group with the shortname "Azure", would send an SMS to the shortcode 20873 that says "ENABLE Azure"

Resubscribing to SMS alerts for all action groups

A user can resubscribe to all SMS for alerts for all action groups by responding to the shortcode 20873 with any of the following keywords:

- START

Ex. A user wishing to unsubscribe from all SMS alerts for all action groups, would send an SMS to the shortcode 20873 that says "START"

Requesting help via SMS

A user can ask for more information about the SMS they have received by responding to the shortcode 20873 with any of the following keywords:

- HELP

A response will be sent to the user with a link to this article.

Next Steps

Get an [overview of activity log alerts](#) and learn how to get alerted

Learn more about [SMS rate limiting](#)

Learn more about [action groups](#)

Rate Limiting for SMS, Emails and Webhooks

7/18/2017 • 1 min to read • [Edit Online](#)

Rate limiting is a suspension of notifications that occurs when too many are sent to a particular phone number or email. Rate limiting ensures that alerts are manageable and actionable.

The rules for SMS and Email are the same. The rate limit threshold for

- SMS - 10 messages in an hour
- Email - 100 messages in an hour

Rate Limit rules

- A particular phone number or email is rate limited when it receives more than the threshold
- A phone number or email can be part of action groups across many subscriptions. Rate limiting applies across all subscriptions, that is, it applies as soon as the threshold is reached even if sent from multiple subscriptions.
- When a phone number or email is rate limited, an additional notification is sent to communicate the rate limiting. The notification states when the rate limiting expires.

Rate Limit of Webhooks

There is no rate limiting in place for webhooks today.

Next Steps

Learn more on [SMS alert behavior](#)

Get an [overview of activity log alerts](#) and learn how to get alerted

[How to configure alerts whenever a service health notification is posted](#)

Create an action group with a Resource Manager Template

7/18/2017 • 1 min to read • [Edit Online](#)

This article shows how you can use an [Azure Resource Manager template](#) to configure action groups. Templates enable you to automatically set up action groups on your resources when they are created to ensure that all the correct parties are notified when an alert is triggered.

The basic steps are as follows:

1. Create a template as a JSON file that describes how to create the action group.
2. [Deploy the template using any deployment method.](#)

Below we describe how to create a Resource Manager template first for an action group where the action definitions are hard coded in the template, then for a template that takes the webhook configuration information as input parameters when deploying the template.

Resource Manager template for an action group

To create an action group using a Resource Manager template, you create a resource of type

`Microsoft.Insights/actionGroups` and fill in all related properties. Following are a couple sample templates that create an action group.

```
{
  "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
  "contentVersion": "1.0.0.0",
  "parameters": {
    "actionGroupName": {
      "type": "string",
      "metadata": {
        "description": "Unique name (within the Resource Group) for the Action group."
      }
    },
    "actionGroupShortName": {
      "type": "string",
      "metadata": {
        "description": "Short name (maximum 12 characters) for the Action group."
      }
    }
  },
  "resources": [
    {
      "type": "Microsoft.Insights/actionGroups",
      "apiVersion": "2017-04-01",
      "name": "[parameters('actionGroupName')]",
      "location": "Global",
      "properties": {
        "groupShortName": "[parameters('actionGroupShortName')]",
        "enabled": true,
        "smsReceivers": [
          {
            "name": "contosoSMS",
            "countryCode": "1",
            "phoneNumber": "5555551212"
          },
          {
            "name": "contosoSMS2",
            "countryCode": "1",
            "phoneNumber": "5555551213"
          }
        ]
      }
    }
  ]
}
```

```
        "countryCode": "1",
        "phoneNumber": "5555552121"
    },
],
"emailReceivers": [
{
    "name": "contosoEmail",
    "emailAddress": "devops@contoso.com"
},
{
    "name": "contosoEmail2",
    "emailAddress": "devops2@contoso.com"
}
],
"webhookReceivers": [
{
    "name": "contosoHook",
    "serviceUri": "http://requestb.in/1bq62iu1"
},
{
    "name": "contosoHook2",
    "serviceUri": "http://requestb.in/1bq62iu2"
}
]
},
"outputs":{
    "actionGroupId":{
        "type":"string",
        "value":"[resourceId('Microsoft.Insights/actionGroups',parameters('actionGroupName'))]"
    }
}
}
```

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "actionGroupName": {
            "type": "string",
            "metadata": {
                "description": "Unique name (within the Resource Group) for the Action group."
            }
        },
        "actionGroupShortName": {
            "type": "string",
            "metadata": {
                "description": "Short name (maximum 12 characters) for the Action group."
            }
        },
        "webhookReceiverName": {
            "type": "string",
            "metadata": {
                "description": "Webhook receiver service URI."
            }
        },
        "webhookServiceUri": {
            "type": "string",
            "metadata": {
                "description": "Webhook receiver service URI."
            }
        }
    },
    "resources": [
        {
            "type": "Microsoft.Insights/actionGroups",
            "apiVersion": "2017-04-01",
            "name": "[parameters('actionGroupName')]",
            "location": "Global",
            "properties": {
                "groupShortName": "[parameters('actionGroupShortName')]",
                "enabled": true,
                "smsReceivers": [
                    ],
                "emailReceivers": [
                    ],
                "webhookReceivers": [
                    {
                        "name": "[parameters('webhookReceiverName')]",
                        "serviceUri": "[parameters('webhookServiceUri')]"
                    }
                ]
            }
        }
    ],
    "outputs":{
        "actionGroupResourceId": {
            "type":"string",
            "value": "[resourceId('Microsoft.Insights/actionGroups',parameters('actionGroupName'))]"
        }
    }
}
```

Next Steps

[Learn more about Action Groups](#)

[Learn more about Alerts](#)

[How to add Alerts using a Resource Manager template](#)

Archive Azure Diagnostic Logs

7/18/2017 • 4 min to read • [Edit Online](#)

In this article, we show how you can use the Azure portal, PowerShell Cmdlets, CLI, or REST API to archive your [Azure Diagnostic Logs](#) in a storage account. This option is useful if you would like to retain your Diagnostic Logs with an optional retention policy for audit, static analysis, or backup. The storage account does not have to be in the same subscription as the resource emitting logs as long as the user who configures the setting has appropriate RBAC access to both subscriptions.

Prerequisites

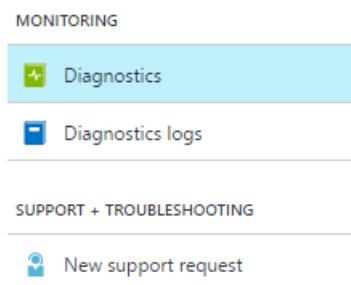
Before you begin, you need to [create a storage account](#) to which you can archive your Diagnostic Logs. We highly recommend that you do not use an existing storage account that has other, non-monitoring data stored in it so that you can better control access to monitoring data. However, if you are also archiving your Activity Log and diagnostic metrics to a storage account, it may make sense to use that storage account for your Diagnostic Logs as well to keep all monitoring data in a central location. The storage account you use must be a general purpose storage account, not a blob storage account.

Diagnostic Settings

To archive your Diagnostic Logs using any of the methods below, you set a **Diagnostic Setting** for a particular resource. A diagnostic setting for a resource defines the categories of logs that are stored or streamed and the outputs—storage account and/or event hub. It also defines the retention policy (number of days to retain) for events of each log category stored in a storage account. If a retention policy is set to zero, events for that log category are stored indefinitely (that is to say, forever). A retention policy can otherwise be any number of days between 1 and 2147483647. [You can read more about diagnostic settings here](#). Retention policies are applied per-day, so at the end of a day (UTC), logs from the day that is now beyond the retention policy will be deleted. For example, if you had a retention policy of one day, at the beginning of the day today the logs from the day before yesterday would be deleted.

Archive Diagnostic Logs using the portal

1. In the portal, click into the resource blade for the resource on which you would like to enable archival of Diagnostic Logs.
2. In the **Monitoring** section of the resource settings menu, select **Diagnostics**.



3. Check the box for **Export to Storage Account**, then select a storage account. Optionally, set a number of days to retain these logs by using the **Retention (days)** sliders. A retention of zero days stores the logs indefinitely.

Save Discard

Status
 Off On

Export to Event Hubs

Export to Storage Account

Storage Account
johnkemtest8162 >

LOGS

| | | | |
|-------------------------------------|--------------------------------|-------------------------------------|-----------------------------------|
| <input checked="" type="checkbox"/> | NetworkSecurityGroupEvent | Retention (days) <small> ⓘ </small> | <input type="range" value="0"/> 0 |
| <input checked="" type="checkbox"/> | NetworkSecurityGroupRuleCou... | Retention (days) <small> ⓘ </small> | <input type="range" value="0"/> 0 |

 The storage account must be in the same region as the resource.

4. Click **Save**.

Diagnostic Logs are archived to that storage account as soon as new event data is generated.

Archive Diagnostic Logs via the PowerShell Cmdlets

```
Set-AzureRmDiagnosticSetting -ResourceId /subscriptions/s1id1234-5679-0123-4567-  

890123456789/resourceGroups/testresourcegroup/providers/Microsoft.Network/networkSecurityGroups/testnsg -  

StorageAccountId /subscriptions/s1id1234-5679-0123-4567-  

890123456789/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/my_storage -Categories  

networksecuritygroupevent,networksecuritygrouprulecounter -Enabled $true -RetentionEnabled $true -  

RetentionInDays 90
```

| PROPERTY | REQUIRED | DESCRIPTION |
|------------------|----------|--|
| ResourceId | Yes | Resource ID of the resource on which you want to set a diagnostic setting. |
| StorageAccountId | No | Resource ID of the Storage Account to which Diagnostic Logs should be saved. |
| Categories | No | Comma-separated list of log categories to enable. |
| Enabled | Yes | Boolean indicating whether diagnostics are enabled or disabled on this resource. |
| RetentionEnabled | No | Boolean indicating if a retention policy are enabled on this resource. |

| PROPERTY | REQUIRED | DESCRIPTION |
|-----------------|----------|--|
| RetentionInDays | No | Number of days for which events should be retained between 1 and 2147483647. A value of zero stores the logs indefinitely. |

Archive Diagnostic Logs via the Cross-Platform CLI

```
azure insights diagnostic set --resourceId /subscriptions/s1id1234-5679-0123-4567-890123456789/resourceGroups/testresourcegroup/providers/Microsoft.Network/networkSecurityGroups/testnsg --storageId /subscriptions/s1id1234-5679-0123-4567-890123456789/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/my_storage -categories networksecuritygroupevent,networksecuritygrouprulecounter --enabled true
```

| PROPERTY | REQUIRED | DESCRIPTION |
|------------|----------|--|
| resourceId | Yes | Resource ID of the resource on which you want to set a diagnostic setting. |
| storageId | No | Resource ID of the Storage Account to which Diagnostic Logs should be saved. |
| categories | No | Comma-separated list of log categories to enable. |
| enabled | Yes | Boolean indicating whether diagnostics are enabled or disabled on this resource. |

Archive Diagnostic Logs via the REST API

[See this document](#) for information on how you can set up a diagnostic setting using the Azure Monitor REST API.

Schema of Diagnostic Logs in the storage account

Once you have set up archival, a storage container is created in the storage account as soon as an event occurs in one of the log categories you have enabled. The blobs within the container follow the same format across Diagnostic Logs and the Activity Log. The structure of these blobs is:

```
insights-logs-{log category name}/resourceld=/SUBSCRIPTIONS/{subscription ID}/RESOURCEGROUPS/{resource group name}/PROVIDERS/{resource provider name}/{resource type}/{resource name}/y={four-digit numeric year}/m={two-digit numeric month}/d={two-digit numeric day}/h={two-digit 24-hour clock hour}/m=00/PT1H.json
```

Or, more simply,

```
insights-logs-{log category name}/resourceld=/resource Id/y={four-digit numeric year}/m={two-digit numeric month}/d={two-digit numeric day}/h={two-digit 24-hour clock hour}/m=00/PT1H.json
```

For example, a blob name might be:

```
insights-logs-networksecuritygrouprulecounter/resourceld=/SUBSCRIPTIONS/s1id1234-5679-0123-4567-
```

890123456789/RESOURCEGROUPS/TESTRESOURCEGROUP/PROVIDERS/MICROSOFT.NETWORK-NETWORKSECURITYGROUP/TESTNSG/y=2016/m=08/d=22/h=18/m=00/PT1H.json

Each PT1H.json blob contains a JSON blob of events that occurred within the hour specified in the blob URL (for example, h=12). During the present hour, events are appended to the PT1H.json file as they occur. The minute value (m=00) is always 00, since Diagnostic Log events are broken into individual blobs per hour.

Within the PT1H.json file, each event is stored in the "records" array, following this format:

```
{  
    "records": [  
        {  
            "time": "2016-07-01T00:00:37.2040000Z",  
            "systemId": "46cdbb41-cb9c-4f3d-a5b4-1d458d827ff1",  
            "category": "NetworkSecurityGroupRuleCounter",  
            "resourceId": "/SUBSCRIPTIONS/s1id1234-5679-0123-4567-  
890123456789/RESOURCEGROUPS/TESTRESOURCEGROUP/PROVIDERS/MICROSOFT.NETWORK-NETWORKSECURITYGROUPS/TESTNSG",  
            "operationName": "NetworkSecurityGroupCounters",  
            "properties": {  
                "vnetResourceGuid": "{12345678-9012-3456-7890-123456789012}",  
                "subnetPrefix": "10.3.0.0/24",  
                "macAddress": "000123456789",  
                "ruleName": "/subscriptions/ s1id1234-5679-0123-4567-  
890123456789/resourceGroups/testresourcegroup/providers/Microsoft.Network/networkSecurityGroups/testnsg/securityRules/default-allow-rdp",  
                "direction": "In",  
                "type": "allow",  
                "matchedConnections": 1988  
            }  
        }  
    ]  
}
```

| ELEMENT NAME | DESCRIPTION |
|---------------|---|
| time | Timestamp when the event was generated by the Azure service processing the request corresponding the event. |
| resourceId | Resource ID of the impacted resource. |
| operationName | Name of the operation. |
| category | Log category of the event. |
| properties | Set of <Key, Value> pairs (i.e. Dictionary) describing the details of the event. |

NOTE

The properties and usage of those properties can vary depending on the resource.

Next Steps

- [Download blobs for analysis](#)
- [Stream Diagnostic Logs to Event Hubs](#)
- [Read more about Diagnostic Logs](#)

Stream Azure Diagnostic Logs to Event Hubs

7/18/2017 • 4 min to read • [Edit Online](#)

Azure Diagnostic Logs can be streamed in near real time to any application using the built-in "Export to Event Hubs" option in the Portal, or by enabling the Service Bus Rule Id in a Diagnostic Setting via the Azure PowerShell Cmdlets or Azure CLI.

What you can do with Diagnostics Logs and Event Hubs

Here are just a few ways you might use the streaming capability for Diagnostic Logs:

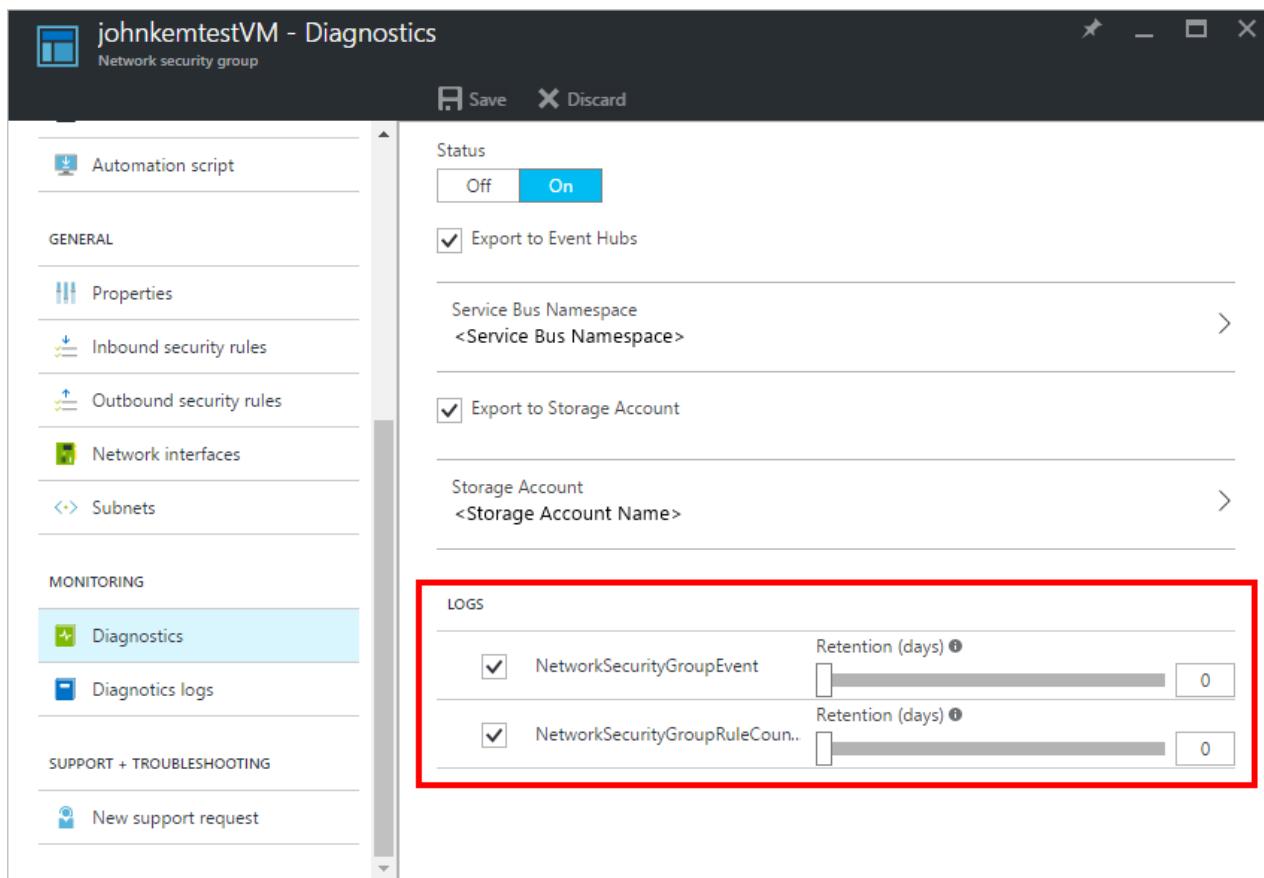
- **Stream logs to 3rd party logging and telemetry systems** – Over time, Event Hubs streaming will become the mechanism to pipe your Diagnostic Logs into third party SIEMs and log analytics solutions.
- **View service health by streaming “hot path” data to PowerBI** – Using Event Hubs, Stream Analytics, and PowerBI, you can easily transform your diagnostics data into near real-time insights on your Azure services. [This documentation article gives a great overview of how to set up Event Hubs, process data with Stream Analytics, and use PowerBI as an output](#). Here are a few tips for getting set up with Diagnostic Logs:
 - An event hub for a category of Diagnostic Logs is created automatically when you check the option in the portal or enable it through PowerShell, so you want to select the event hub in the namespace with the name that starts with **insights-**.
 - The following SQL code is a sample Stream Analytics query that you can use to parse all the log data into a PowerBI table:

```
SELECT
records.ArrayValue.[Properties you want to track]
INTO
[OutputSourceName - the PowerBI source]
FROM
[InputSourceName] AS e
CROSS APPLY GetArrayElements(e.records) AS records
```

- **Build a custom telemetry and logging platform** – If you already have a custom-built telemetry platform or are just thinking about building one, the highly scalable publish-subscribe nature of Event Hubs allows you to flexibly ingest diagnostic logs. [See Dan Rosanova’s guide to using Event Hubs in a global scale telemetry platform here](#).

Enable streaming of Diagnostic Logs

You can enable streaming of Diagnostic Logs programmatically, via the portal, or using the [Azure Monitor REST API](#). Either way, you pick an Event Hubs namespace and an event hub is created in the namespace for each log category you enable. A Diagnostic **Log Category** is a type of log that a resource may collect. You can select which log categories you’d like to collect for a particular resource in the Azure Portal under the Diagnostics blade.



WARNING

Enabling and streaming diagnostic logs from Compute resources (for example, VMs or Service Fabric) [requires a different set of steps](#).

The Service Bus or Event Hubs namespace does not have to be in the same subscription as the resource emitting logs as long as the user who configures the setting has appropriate RBAC access to both subscriptions.

Via PowerShell Cmdlets

To enable streaming via the [Azure PowerShell Cmdlets](#), you can use the `Set-AzureRmDiagnosticSetting` cmdlet with these parameters:

```
Set-AzureRmDiagnosticSetting -ResourceId [your resource Id] -ServiceBusRuleId [your Service Bus rule id] -Enabled $true
```

The Service Bus Rule ID is a string with this format: `{Service Bus resource ID}/authorizationrules/{key name}`, for example,

```
/subscriptions/{subscription ID}/resourceGroups/Default-ServiceBus-WestUS/providers/Microsoft.ServiceBus/namespaces/{Service Bus namespace}/authorizationrules/RootManageSharedAccessKey
```

Via Azure CLI

To enable streaming via the [Azure CLI](#), you can use the `insights diagnostic set` command like this:

```
azure insights diagnostic set --resourceId <resourceId> --serviceBusRuleId <serviceBusRuleId> --enabled true
```

Use the same format for Service Bus Rule ID as explained for the PowerShell Cmdlet.

Via Azure Portal

To enable streaming via the Azure Portal, navigate to the diagnostics settings of a resource and select **Export to Event Hub**.

The screenshot shows the 'Diagnostics' settings for a Network security group. The 'Status' is set to 'On'. The 'Export to Event Hubs' checkbox is selected and highlighted with a red box. The 'Export to Storage Account' checkbox is also checked. Under 'LOGS', two categories are listed: 'NetworkSecurityGroupEvent' and 'NetworkSecurityGroupRuleCoun..'. Both have their 'Retention (days)' set to 0. A note at the bottom states: 'The storage account must be in the same region as the resource.'

To configure it, select an existing Event Hubs namespace. The namespace selected will be where the event hub is created (if this is your first time streaming diagnostic logs) or streamed to (if there are already resources that are streaming that log category to this namespace), and the policy defines the permissions that the streaming mechanism has. Today, streaming to an event hub requires Manage, Send, and Listen permissions. You can create or modify Event Hubs namespace shared access policies in the portal under the **Configure** tab for your namespace. To update one of these diagnostic settings, the client must have the **ListKey** permission on the Event Hubs authorization rule.

How do I consume the log data from Event Hubs?

Here is sample output data from Event Hubs:

```
{
  "records": [
    {
      "time": "2016-07-15T18:00:22.6235064Z",
      "workflowId": "/SUBSCRIPTIONS/DF602C9C-7AA0-407D-A6FB-
EB20C8BD1192/RESOURCEGROUPS/JOHNKEMTEST/PROVIDERS/MICROSOFT.LOGIC/WORKFLOWS/JOHNKEMTESTLA",
      "resourceId": "/SUBSCRIPTIONS/DF602C9C-7AA0-407D-A6FB-
EB20C8BD1192/RESOURCEGROUPS/JOHNKEMTEST/PROVIDERS/MICROSOFT.LOGIC/WORKFLOWS/JOHNKEMTESTLA/RUNS/085873300135099
21957/ACTIONS/SEND_EMAIL",
      "category": "WorkflowRuntime",
      "level": "Error",
      "operationName": "Microsoft.Logic/workflows/workflowActionCompleted",
      "properties": {
        "$schema": "2016-04-01-preview",
        "startTime": "2016-07-15T17:58:55.048482Z",
        "endTime": "2016-07-15T18:00:22.4109204Z",
        "status": "Failed",
        "code": "BadGateway",
        "resource": {
          "subscriptionId": "df602c9c-7aa0-407d-a6fb-eb20c8bd1192",
          "resourceGroupName": "JohnKemTest",
          "workflowId": "243aac67fe904cf195d4a28297803785",
          "workflowName": "JohnKemTestLA",
          "runId": "08587330013509921957",
          "location": "westus",
          "actionName": "Send_email"
        },
        "correlation": {
          "actionTrackingId": "29a9862f-969b-4c70-90c4-dfbdc814e413",
          "clientTrackingId": "08587330013509921958"
        }
      }
    },
    {
      "time": "2016-07-15T18:01:15.7532989Z",
      "workflowId": "/SUBSCRIPTIONS/DF602C9C-7AA0-407D-A6FB-
EB20C8BD1192/RESOURCEGROUPS/JOHNKEMTEST/PROVIDERS/MICROSOFT.LOGIC/WORKFLOWS/JOHNKEMTESTLA",
      "resourceId": "/SUBSCRIPTIONS/DF602C9C-7AA0-407D-A6FB-
EB20C8BD1192/RESOURCEGROUPS/JOHNKEMTEST/PROVIDERS/MICROSOFT.LOGIC/WORKFLOWS/JOHNKEMTESTLA/RUNS/085873300121067
02630/ACTIONS/SEND_EMAIL",
      "category": "WorkflowRuntime",
      "level": "Information",
      "operationName": "Microsoft.Logic/workflows/workflowActionStarted",
      "properties": {
        "$schema": "2016-04-01-preview",
        "startTime": "2016-07-15T18:01:15.5828115Z",
        "status": "Running",
        "resource": {
          "subscriptionId": "df602c9c-7aa0-407d-a6fb-eb20c8bd1192",
          "resourceGroupName": "JohnKemTest",
          "workflowId": "243aac67fe904cf195d4a28297803785",
          "workflowName": "JohnKemTestLA",
          "runId": "08587330012106702630",
          "location": "westus",
          "actionName": "Send_email"
        },
        "correlation": {
          "actionTrackingId": "042fb72c-7bd4-439e-89eb-3cf4409d429e",
          "clientTrackingId": "08587330012106702632"
        }
      }
    }
  ]
}
```

| ELEMENT NAME | DESCRIPTION |
|---------------|--|
| records | An array of all log events in this payload. |
| time | Time at which the event occurred. |
| category | Log category for this event. |
| resourceId | Resource ID of the resource that generated this event. |
| operationName | Name of the operation. |
| level | Optional. Indicates the log event level. |
| properties | Properties of the event. |

You can view a list of all resource providers that support streaming to Event Hubs [here](#).

Stream data from Compute resources

You can also stream diagnostic logs from Compute resources using the Windows Azure Diagnositcs agent. See [this article](#) for how to set that up.

Next Steps

- [Read more about Azure Diagnostic Logs](#)
- [Get started with Event Hubs](#)

Automatically enable Diagnostic Settings at resource creation using a Resource Manager template

7/18/2017 • 3 min to read • [Edit Online](#)

In this article we show how you can use an [Azure Resource Manager template](#) to configure Diagnostic Settings on a resource when it is created. This enables you to automatically start streaming your Diagnostic Logs and metrics to Event Hubs, archiving them in a Storage Account, or sending them to Log Analytics when a resource is created.

The method for enabling Diagnostic Logs using a Resource Manager template depends on the resource type.

- **Non-Compute** resources (for example, Network Security Groups, Logic Apps, Automation) use [Diagnostic Settings described in this article](#).
- **Compute** (WAD/LAD-based) resources use the [WAD/LAD configuration file described in this article](#).

In this article we describe how to configure diagnostics using either method.

The basic steps are as follows:

1. Create a template as a JSON file that describes how to create the resource and enable diagnostics.
2. [Deploy the template using any deployment method](#).

Below we give an example of the template JSON file you need to generate for non-Compute and Compute resources.

Non-Compute resource template

For non-Compute resources, you will need to do two things:

1. Add parameters to the parameters blob for the storage account name, service bus rule ID, and/or OMS Log Analytics workspace ID (enabling archival of Diagnostic Logs in a storage account, streaming of logs to Event Hubs, and/or sending logs to Log Analytics).

```
"storageAccountName": {  
    "type": "string",  
    "metadata": {  
        "description": "Name of the Storage Account in which Diagnostic Logs should be saved."  
    }  
},  
"serviceBusRuleId": {  
    "type": "string",  
    "metadata": {  
        "description": "Service Bus Rule Id for the Service Bus Namespace in which the Event Hub should be created or streamed to."  
    }  
},  
"workspaceId":{  
    "type": "string",  
    "metadata": {  
        "description": "Log Analytics workspace ID for the Log Analytics workspace to which logs will be sent."  
    }  
}
```

2. In the resources array of the resource for which you want to enable Diagnostic Logs, add a resource of type `[resource namespace]/providers/diagnosticSettings`.

```

"resources": [
    {
        "type": "providers/diagnosticSettings",
        "name": "Microsoft.Insights/service",
        "dependsOn": [
            "[/*resource Id for which Diagnostic Logs will be enabled*/]"
        ],
        "apiVersion": "2015-07-01",
        "properties": {
            "storageAccountId": "[resourceId('Microsoft.Storage/storageAccounts',
parameters('storageAccountName'))]",
            "serviceBusRuleId": "[parameters('serviceBusRuleId')]",
            "workspaceId": "[parameters('workspaceId')]",
            "logs": [
                {
                    "category": "/* log category name */",
                    "enabled": true,
                    "retentionPolicy": {
                        "days": 0,
                        "enabled": false
                    }
                }
            ],
            "metrics": [
                {
                    "timeGrain": "PT1M",
                    "enabled": true,
                    "retentionPolicy": {
                        "enabled": false,
                        "days": 0
                    }
                }
            ]
        }
    }
]

```

The properties blob for the Diagnostic Setting follows [the format described in this article](#). Adding the `metrics` property will enable you to also send resource metrics to these same outputs, provided that [the resource supports Azure Monitor metrics](#).

Here is a full example that creates a Logic App and turns on streaming to Event Hubs and storage in a storage account.

```
{
    "$schema": "https://schema.management.azure.com/schemas/2015-01-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "logicAppName": {
            "type": "string",
            "metadata": {
                "description": "Name of the Logic App that will be created."
            }
        },
        "testUri": {
            "type": "string",
            "defaultValue": "http://azure.microsoft.com/en-us/status/feed/"
        },
        "storageAccountName": {
            "type": "string",
            "metadata": {
                "description": "Name of the Storage Account in which Diagnostic Logs should be saved."
            }
        }
    }
}
```

```

    },
    "serviceBusRuleId": {
        "type": "string",
        "metadata": {
            "description": "Service Bus Rule Id for the Service Bus Namespace in which the Event Hub should be created or streamed to."
        }
    },
    "workspaceId": {
        "type": "string",
        "metadata": {
            "description": "Log Analytics workspace ID for the Log Analytics workspace to which logs will be sent."
        }
    },
    "variables": {},
    "resources": [
        {
            "type": "Microsoft.Logic/workflows",
            "name": "[parameters('logicAppName')]",
            "apiVersion": "2016-06-01",
            "location": "[resourceGroup().location]",
            "properties": {
                "definition": {
                    "$schema": "http://schema.management.azure.com/providers/Microsoft.Logic/schemas/2016-06-01/workflowdefinition.json#",
                    "contentVersion": "1.0.0.0",
                    "parameters": {
                        "testURI": {
                            "type": "string",
                            "defaultValue": "[parameters('testUri')]"
                        }
                    },
                    "triggers": {
                        "recurrence": {
                            "type": "recurrence",
                            "recurrence": {
                                "frequency": "Hour",
                                "interval": 1
                            }
                        }
                    },
                    "actions": {
                        "http": {
                            "type": "Http",
                            "inputs": {
                                "method": "GET",
                                "uri": "@parameters('testUri')"
                            },
                            "runAfter": {}
                        }
                    },
                    "outputs": {}
                },
                "parameters": {}
            },
            "resources": [
                {
                    "type": "providers/diagnosticSettings",
                    "name": "Microsoft.Insights/service",
                    "dependsOn": [
                        "[resourceId('Microsoft.Logic/workflows', parameters('logicAppName'))]"
                    ],
                    "apiVersion": "2015-07-01",
                    "properties": {
                        "storageAccountId": "[resourceId('Microsoft.Storage/storageAccounts', parameters('storageAccountName'))]",
                        "serviceBusRuleId": "[parameters('serviceBusRuleId')]",
                        "workspaceId": "[parameters('workspaceId')]"
                    }
                }
            ]
        }
    ]
}

```

```
        "logs": [
            {
                "category": "WorkflowRuntime",
                "enabled": true,
                "retentionPolicy": {
                    "days": 0,
                    "enabled": false
                }
            }
        ],
        "metrics": [
            {
                "timeGrain": "PT1M",
                "enabled": true,
                "retentionPolicy": {
                    "enabled": false,
                    "days": 0
                }
            }
        ]
    },
    "dependsOn": []
}
]
}
```

Compute resource template

To enable diagnostics on a Compute resource, for example a Virtual Machine or Service Fabric cluster, you need to:

1. Add the Azure Diagnostics extension to the VM resource definition.
2. Specify a storage account and/or event hub as a parameter.
3. Add the contents of your WADCfg XML file into the XMLCfg property, escaping all XML characters properly.

WARNING

This last step can be tricky to get right. [See this article](#) for an example that splits the Diagnostics Configuration Schema into variables that are escaped and formatted correctly.

The entire process, including samples, is described [in this document](#).

Next Steps

- [Read more about Azure Diagnostic Logs](#)
- [Stream Azure Diagnostic Logs to Event Hubs](#)

Azure Monitoring REST API Walkthrough

7/18/2017 • 6 min to read • [Edit Online](#)

This article shows you how to perform authentication so your code can use the [Microsoft Azure Monitor REST API Reference](#).

The Azure Monitor API makes it possible to programmatically retrieve the available default metric definitions (the type of metric such as CPU Time, Requests, etc.), granularity, and metric values. Once retrieved, the data can be saved in a separate data store such as Azure SQL Database, Azure Cosmos DB, or Azure Data Lake. From there additional analysis can be performed as needed.

Besides working with various metric data points, as this article demonstrates, the Monitor API makes it possible to list alert rules, view activity logs, and much more. For a full list of available operations, see the [Microsoft Azure Monitor REST API Reference](#).

Authenticating Azure Monitor Requests

The first step is to authenticate the request.

All the tasks executed against the Azure Monitor API use the Azure Resource Manager authentication model. Therefore, all requests must be authenticated with Azure Active Directory (Azure AD). One approach to authenticate the client application is to create an Azure AD service principal and retrieve the authentication (JWT) token. The following sample script demonstrates creating an Azure AD service principal via PowerShell. For a more detailed walk-through, refer to the documentation on [using Azure PowerShell to create a service principal to access resources](#). It is also possible to [create a service principal via the Azure portal](#).

```
$subscriptionId = "{azure-subscription-id}"
$resourceGroupName = "{resource-group-name}"
$location = "centralus"

# Authenticate to a specific Azure subscription.
Login-AzureRmAccount -SubscriptionId $subscriptionId

# Password for the service principal
$pwd = "{service-principal-password}"

# Create a new Azure AD application
$azureAdApplication = New-AzureRmADApplication ` 
    -DisplayName "My Azure Monitor" ` 
    -HomePage "https://localhost/azure-monitor" ` 
    -IdentifierUris "https://localhost/azure-monitor" ` 
    -Password $pwd

# Create a new service principal associated with the designated application
New-AzureRmADServicePrincipal -ApplicationId $azureAdApplication.ApplicationId

# Assign Reader role to the newly created service principal
New-AzureRmRoleAssignment -RoleDefinitionName Reader ` 
    -ServicePrincipalName $azureAdApplication.ApplicationId.Guid
```

To query the Azure Monitor API, the client application should use the previously created service principal to authenticate. The following example PowerShell script shows one approach, using the [Active Directory Authentication Library](#) (ADAL) to help get the JWT authentication token. The JWT token is passed as part of an HTTP Authorization parameter in requests to the Azure Monitor REST API.

```

$azureAdApplication = Get-AzureRmADApplication -IdentifierUri "https://localhost/azure-monitor"

$subcription = Get-AzureRmSubscription -SubscriptionId $subscriptionId

$clientID = $azureAdApplication.ApplicationId.Guid
$tenantId = $subcription.TenantId
$authUrl = "https://login.microsoftonline.com/${tenantId}"

$AuthContext = [Microsoft.IdentityModel.Clients.ActiveDirectory.AuthenticationContext]$authUrl
$cred = New-Object -TypeName Microsoft.IdentityModel.Clients.ActiveDirectory.ClientCredential -ArgumentList
($clientID, $pwd)

$result = $AuthContext.AcquireToken("https://management.core.windows.net/", $cred)

# Build an array of HTTP header values
$authHeader = @{
    'Content-Type'='application/json'
    'Accept'='application/json'
    'Authorization'=$result.CreateAuthorizationHeader()
}

```

Once the authentication setup step is complete, queries can then be executed against the Azure Monitor REST API. There are two helpful queries:

1. List the metric definitions for a resource
2. Retrieve the metric values

Retrieve Metric Definitions

NOTE

To retrieve metric definitions using the Azure Monitor REST API, use "2016-03-01" as the API version.

```

$apiVersion = "2016-03-01"
$request =
"https://management.azure.com/subscriptions/${subscriptionId}/resourceGroups/${resourceGroupName}/providers/${resourceProviderNamespace}/${resourceType}/${resourceName}/providers/microsoft.insights/metricDefinitions?api-
version=${apiVersion}"

Invoke-RestMethod -Uri $request ` 
    -Headers $authHeader ` 
    -Method Get ` 
    -Verbose

```

For an Azure Logic App, the metric definitions would appear similar to the following screenshot:

```

1 {
2   "id": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic/providers/microsoft.insights/metricdefinitions",
3   "value": [
4     {
5       "resourceUri": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
6       "resourceId": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
7       "name": {
8         "value": "RunsStarted",
9         "localizedValue": "RunsStarted"
10      },
11      "startTime": "0 0 0 1 -0 1T00: 0 0: 0 0Z",
12      "endTime": "0 0 0 1 -0 1T00: 0 0: 0 0Z",
13      "unit": "Count",
14      "primaryAggregationType": "Total",
15      "ResourceUri": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
16      "ResourceId": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
17      "metricAvailabilities": [
18        {
19          "timeGrain": "PT1M",
20          "retention": "P30D"
21        },
22        {
23          "timeGrain": "PT1H",
24          "retention": "P30D"
25        }
26      ],
27      "id": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic/providers/microsoft.insights/metricdefinitions/RunsStarted"
28    },
29    {
30      "resourceUri": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
31      "resourceId": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
32      "name": {
33        "value": "RunsCompleted",
34        "localizedValue": "RunsCompleted"
35      },
36      "startTime": "0 0 0 1 -0 1T00: 0 0: 0 0Z",
37      "endTime": "0001-01-01T00:00:00Z",
38      "unit": "Count",
39      "primaryAggregationType": "Total",
40      "ResourceUri": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
41      "ResourceId": "/subscriptions//resourceGroups/contosoweb001/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
42      "metricAvailabilities": [
43        {
44          "timeGrain": "PT1M",
45          "retention": "P30D"
46        },
47        {
48          "timeGrain": "PT1H",
49          "retention": "P30D"
50        }
51      ]
52    }
53  ]
54}

```

For more information, see the [List the metric definitions for a resource in Azure Monitor REST API](#) documentation.

Retrieve Metric Values

Once the available metric definitions are known, it is then possible to retrieve the related metric values. Use the metric's name 'value' (not the 'localizedValue') for any filtering requests (for example, retrieve the 'CpuTime' and 'Requests' metric data points). If no filters are specified, the default metric is returned.

NOTE

To retrieve metric values using the Azure Monitor REST API, use "2016-06-01" as the API version.

Method: GET

Request URI: [https://management.azure.com/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/{resource-provider-namespace}/{resource-type}/{resource-name}/providers/microsoft.insights/metrics?\\$filter={filter}&api-version={apiVersion}](https://management.azure.com/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/{resource-provider-namespace}/{resource-type}/{resource-name}/providers/microsoft.insights/metrics?$filter={filter}&api-version={apiVersion})

For example, to retrieve the RunsSucceeded metric data points for the given time range and for a time grain of 1 hour, the request would be as follows:

```

$apiVersion = "2016-06-01"
$filter = "(name.value eq 'RunsSucceeded') and aggregationType eq 'Total' and startTime eq 2016-09-23 and
endTime eq 2016-09-24 and timeGrain eq duration'PT1H'"
$request =
"https://management.azure.com/subscriptions/${subscriptionId}/resourceGroups/${resourceGroupName}/providers/${resourceProviderNamespace}/${resourceType}/${resourceName}/providers/microsoft.insights/metrics?
`$filter=${filter}&api-version=${apiVersion}"
(Invoke-RestMethod -Uri $request `-
-Headers $authHeader `-
-Method Get `-
-Verbose).Value | ConvertTo-Json

```

The result would appear similar to the example following screenshot:

```

1  "data": [
2      {"timeStamp": "2016-09-23T00:00:00Z"},  

3      {"timeStamp": "2016-09-23T01:00:00Z"},  

4      {"timeStamp": "2016-09-23T02:00:00Z"},  

5      {"timeStamp": "2016-09-23T03:00:00Z"},  

6      {"timeStamp": "2016-09-23T04:00:00Z"},  

7      {"timeStamp": "2016-09-23T05:00:00Z"},  

8      {"timeStamp": "2016-09-23T06:00:00Z"},  

9      {"timeStamp": "2016-09-23T07:00:00Z"},  

10     {"timeStamp": "2016-09-23T08:00:00Z"},  

11     {"timeStamp": "2016-09-23T09:00:00Z"},  

12     {"timeStamp": "2016-09-23T10:00:00Z"},  

13     {"timeStamp": "2016-09-23T11:00:00Z"},  

14     {"timeStamp": "2016-09-23T12:00:00Z"},  

15     {"timeStamp": "2016-09-23T13:00:00Z",  

16         "total": 8.0  

17     },  

18     {"timeStamp": "2016-09-23T14:00:00Z",  

19         "total": 52.0  

20     },  

21     {"timeStamp": "2016-09-23T15:00:00Z",  

22         "total": 22.0  

23     },  

24     {"timeStamp": "2016-09-23T16:00:00Z"},  

25     {"timeStamp": "2016-09-23T17:00:00Z"},  

26     {"timeStamp": "2016-09-23T18:00:00Z"},  

27     {"timeStamp": "2016-09-23T19:00:00Z"},  

28     {"timeStamp": "2016-09-23T20:00:00Z"},  

29     {"timeStamp": "2016-09-23T21:00:00Z"},  

30     {"timeStamp": "2016-09-23T22:00:00Z"},  

31     {"timeStamp": "2016-09-23T23:00:00Z"}  

32 ],  

33     "name": {  

34         "value": "RunsSucceeded",  

35         "localizedValue": "Runs Succeeded"  

36     },  

37     "unit": "θ"  

38 }  

39

```

To retrieve multiple data or aggregation points, add the metric definition names and aggregation types to the filter, as seen in the following example:

```

$apiVersion = "2016-06-01"
$filter = "(name.value eq 'ActionsCompleted' or name.value eq 'RunsSucceeded') and (aggregationType eq 'Total'
or aggregationType eq 'Average') and startTime eq 2016-09-23T13:30:00Z and endTime eq 2016-09-23T14:30:00Z and
timeGrain eq duration'PT1M'"
$request =
"https://management.azure.com/subscriptions/${subscriptionId}/resourceGroups/${resourceGroupName}/providers/${resourceProviderNamespace}/${resourceType}/${resourceName}/providers/microsoft.insights/metrics?
`$filter=${filter}&api-version=${apiVersion}"
(Invoke-RestMethod -Uri $request `-
    -Headers $authHeader `-
    -Method Get `-
    -Verbose).Value | ConvertTo-Json

```

Use ARMClient

An alternative to using PowerShell (as shown above), is to use [ARMClient](#) on your Windows machine. ARMClient handles the Azure AD authentication (and resulting JWT token) automatically. The following steps outline use of ARMClient for retrieving metric data:

1. Install [Chocolatey](#) and [ARMClient](#).
2. In a terminal window, type `armclient.exe login`. This prompts you to log in to Azure.
3. Type `armclient GET [your_resource_id]/providers/microsoft.insights/metricdefinitions?api-version=2016-03-01`
4. Type `armclient GET [your_resource_id]/providers/microsoft.insights/metrics?api-version=2016-06-01`

```
C:\>armclient GET /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Logic/workflows/ContosoTweetsLogic/providers/microsoft.insights/metricdefinitions?api-version=2016-03-01
{
  "id": "/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Logic/workflows/ContosoTweetsLogic/providers/microsoft.insights/metricdefinitions",
  "value": [
    {
      "resourceUri": "/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
      "resourceId": "{resource-id}",
      "name": {
        "value": "RunsStarted",
        "localizedValue": "Runs Started"
      },
      "startTime": "2001-01-01T00:00:00Z",
      "endTime": "2001-01-01T00:00:00Z",
      "unit": "Count",
      "primaryAggregationType": "Total",
      "ResourceUri": "/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
      "ResourceId": "{resource-id}",
      "metricAvailabilities": [
        {
          "timeGrain": "PT1M",
          "retention": "P30D"
        },
        {
          "timeGrain": "PT1H",
          "retention": "P30D"
        }
      ],
      "id": "/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Logic/workflows/ContosoTweetsLogic/providers/microsoft.insights/metricdefinitions/RunsStarted"
    },
    {
      "resourceUri": "/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
      "resourceId": "{resource-id}",
      "name": {
        "value": "RunsCompleted",
        "localizedValue": "Runs Completed"
      },
      "startTime": "2001-01-01T00:00:00Z",
      "endTime": "2001-01-01T00:00:00Z",
      "unit": "Count",
      "primaryAggregationType": "Total",
      "ResourceUri": "/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Logic/workflows/ContosoTweetsLogic",
      "ResourceId": "{resource-id}",
      "metricAvailabilities": [
        {
          "timeGrain": "PT1M",
          "retention": "P30D"
        }
      ]
    }
  ]
}
```

Retrieve the Resource ID

Using the REST API can really help to understand the available metric definitions, granularity, and related values. That information is helpful when using the [Azure Management Library](#).

For the preceding code, the resource ID to use is the full path to the desired Azure resource. For example, to query against an Azure Web App, the resource ID would be:

```
/subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Web/sites/{site-name}/
```

The following list contains a few examples of resource ID formats for various Azure resources:

- **IoT Hub** - /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Devices/iotHubs/{iot-hub-name}
- **Elastic SQL Pool** - /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Sql/servers/{pool-db}/elasticpools/{sql-pool-name}
- **SQL Database (v12)** - /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Sql/servers/{server-name}/databases/{database-name}
- **Service Bus** - /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.ServiceBus/{namespace}/{servicebus-name}
- **VM Scale Sets** - /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Compute/virtualMachineScaleSets/{vm-name}
- **VMs** - /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.Compute/virtualMachines/{vm-name}
- **Event Hubs** - /subscriptions/{subscription-id}/resourceGroups/{resource-group-name}/providers/Microsoft.EventHub/namespaces/{eventhub-namespace}

There are alternative approaches to retrieving the resource ID, including using Azure Resource Explorer, viewing the desired resource in the Azure portal, and via PowerShell or the Azure CLI.

Azure Resource Explorer

To find the resource ID for a desired resource, one helpful approach is to use the [Azure Resource Explorer](#) tool. Navigate to the desired resource and then look at the ID shown, as in the following screenshot:

```

1< {
2   "id": "/subscriptions/8e95e0bb-d7cc-45d3-848c-e4b4a18a56ba/resourceGroups/contosoweb001/providers/Microsoft.Web/sites/contosoweb001",
3   "name": "contosoweb001",
4   "type": "Microsoft.Web/sites",
5   "location": "Central US",
6   "tags": null,
7   "properties": {
8     "name": "contosoweb001",
9     "state": "Running",
10    "hostNames": [
11      "contosoweb001.azurewebsites.net"
12    ],
13    "webSpace": "contosoweb001-CentralUSWebspace",
14    "selfLink": "https://waws-prod-dm1-015.api.azurewebsites.net:454/subscriptions/8e95e0bb-d7cc-45d3-848c-e4b4a18a56ba/webspaces/contosoweb001-CentralUSWebspace/sites/contosoweb001",
15    "repositorySiteName": "contosoweb001",
16    "connectionString": null,
17    "usageState": 0,
18    "enabled": true,
19    "adminEnabled": true,
20    "enableDiagnostics": true,
21    "enableDiagnosticsForHostNames": [
22      "contosoweb001.azurewebsites.net",
23      "contosoweb001.scm.azurewebsites.net"
24    ],
25  }
}

```

Azure portal

The resource ID can also be obtained from the Azure portal. To do so, navigate to the desired resource and then select Properties. The Resource ID is displayed in the Properties blade, as seen in the following screenshot:

contosoweb001 - Properties

App Service

- Quickstart
- Deployment credentials
- Deployment slots
- Deployment options

SETTINGS

- Application settings
- Authentication / Authorization
- Backups
- Custom domains
- SSL certificates
- Networking
- Scale up (App Service plan)
- Scale out (App Service plan)
- Security Scanning
- WebJobs
- MySQL In App (preview)
- Properties**
- Locks
- Automation script

FTP/DEPLOYMENT USER

FTP HOST NAME

FTP DIAGNOSTIC LOGS

FTPS HOST NAME

FTPS DIAGNOSTIC LOGS

RESOURCE ID

LOCATION

Central US

RESOURCE GROUP

[Change resource group](#)

Azure PowerShell

The resource ID can be retrieved using Azure PowerShell cmdlets as well. For example, to obtain the resource ID for an Azure Web App, execute the `Get-AzureRmWebApp` cmdlet, as in the following screenshot:

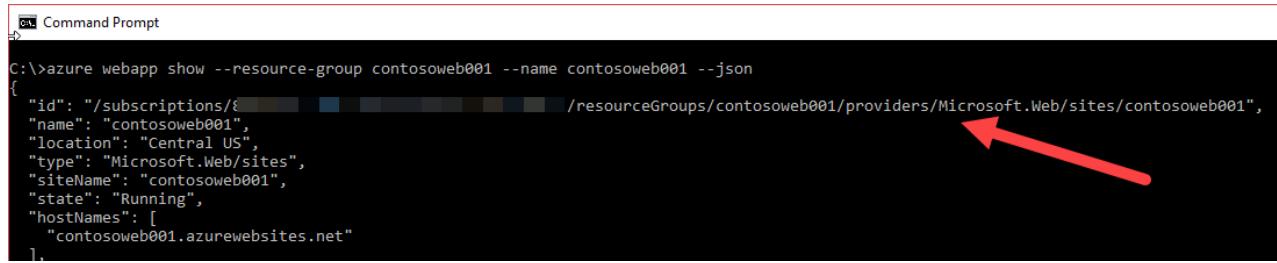
```
PS[C:\]> Get-AzureRmWebApp -ResourceGroupName $resourceGroupName -Name $webAppName

SiteName      : contosoweb001
State         : Running
HostNames     : {contosoweb001.azurewebsites.net}
RepositorySiteName : contosoweb001
UsageState    : Normal
Enabled       : True
EnabledHostNames : {contosoweb001.azurewebsites.net, contosoweb001.scm.azurewebsites.net}
AvailabilityState : Normal
HostNameSslStates : {contosoweb001.azurewebsites.net, contosoweb001.scm.azurewebsites.net}
ServerFarmId   : /subscriptions/{REDACTED}/resourceGroups/contosoweb001/providers/Microsoft.Web/serverfarms/contosowebplan
LastModifiedTimeUtc : 9/20/2016 1:35:40 AM
SiteConfig     : Microsoft.Azure.Management.WebSites.Models.SiteConfig
TrafficManagerHostNames :
PremiumAppDeployed :
ScmSiteAlwaysStopped :
TargetSwapSlot :
HostingEnvironmentProfile :
Microservice   :
GatewaySiteName :
ClientAffinityEnabled : True
ClientCertEnabled  : False
HostNamesDisabled : False
OutboundIpAddresses : 40.86.95.108,40.86.95.228,40.86.92.253,40.86.95.159
ContainersSize  : 0
MaxNumberOfWorkers :
CloningInfo    :
ResourceGroup  : contosoweb001
IsDefaultContainer :
DefaultHostName : contosoweb001.azurewebsites.net
Id            : /subscriptions/{REDACTED}/resourceGroups/contosoweb001/providers/Microsoft.Web/sites/contosoweb001
Name          : contosoweb001
Location      : Central US
Type          : Microsoft.Web/sites
Tags          :
```



Azure CLI

To retrieve the resource ID using the Azure CLI, execute the 'azure webapp show' command, specifying the '--json' option, as shown in the following screenshot:



```
C:\> az webapp show --resource-group contosoweb001 --name contosoweb001 --json
{
  "id": "/subscriptions/{REDACTED}/resourceGroups/contosoweb001/providers/Microsoft.Web/sites/contosoweb001",
  "name": "contosoweb001",
  "location": "Central US",
  "type": "Microsoft.Web/sites",
  "siteName": "contosoweb001",
  "state": "Running",
  "hostNames": [
    "contosoweb001.azurewebsites.net"
  ]
}.
```

Retrieve Activity Log Data

In addition to working with metric definitions and related values, it is also possible to retrieve additional interesting insights related to Azure resources. As an example, it is possible to query [activity log](#) data. The following sample demonstrates using the Azure Monitor REST API to query activity log data within a specific date range for an Azure subscription:

```
$apiVersion = "2014-04-01"
$filter = "eventTimestamp ge '2016-09-23' and eventTimestamp le '2016-09-24' and eventChannels eq 'Admin,
Operation'"
$request =
"https://management.azure.com/subscriptions/${subscriptionId}/providers/microsoft.insights/eventtypes/management/values?api-version=${apiVersion}&$filter=${filter}"
(Invoke-RestMethod -Uri $request `-
    -Headers $authHeader `-
    -Method Get `-
    -Verbose).Value | ConvertTo-Json
```

Next steps

- Review the [Overview of Monitoring](#).
- View the [Supported metrics with Azure Monitor](#).
- Review the [Microsoft Azure Monitor REST API Reference](#).
- Review the [Azure Management Library](#).

Send Cloud Service, Virtual Machine, or Service Fabric diagnostic data to Application Insights

6/27/2017 • 4 min to read • [Edit Online](#)

Cloud services, Virtual Machines, Virtual Machine Scale Sets and Service Fabric all use the Azure Diagnostics extension to collect data. Azure diagnostics sends data to Azure Storage tables. However, you can also pipe all or a subset of the data to other locations using Azure Diagnostics extension 1.5 or later.

This article describes how to send data from the Azure Diagnostics extension to Application Insights.

Diagnostics configuration explained

The Azure diagnostics extension 1.5 introduced sinks, which are additional locations where you can send diagnostic data.

Example configuration of a sink for Application Insights:

```
<SinksConfig>
  <Sink name="ApplicationInsights">
    <ApplicationInsights>{Insert InstrumentationKey}</ApplicationInsights>
    <Channels>
      <Channel logLevel="Error" name="MyTopDiagData" />
      <Channel logLevel="Verbose" name="MyLogData" />
    </Channels>
  </Sink>
</SinksConfig>
```

```
"SinksConfig": {
  "Sink": [
    {
      "name": "ApplicationInsights",
      "ApplicationInsights": "{Insert InstrumentationKey}",
      "Channels": {
        "Channel": [
          {
            "logLevel": "Error",
            "name": "MyTopDiagData"
          },
          {
            "logLevel": "Error",
            "name": "MyLogData"
          }
        ]
      }
    }
  ]
}
```

- The **Sink name** attribute is a string value that uniquely identifies the sink.
- The **ApplicationInsights** element specifies instrumentation key of the Application insights resource where the Azure diagnostics data is sent.
 - If you don't have an existing Application Insights resource, see [Create a new Application Insights resource](#) for more information on creating a resource and getting the instrumentation key.

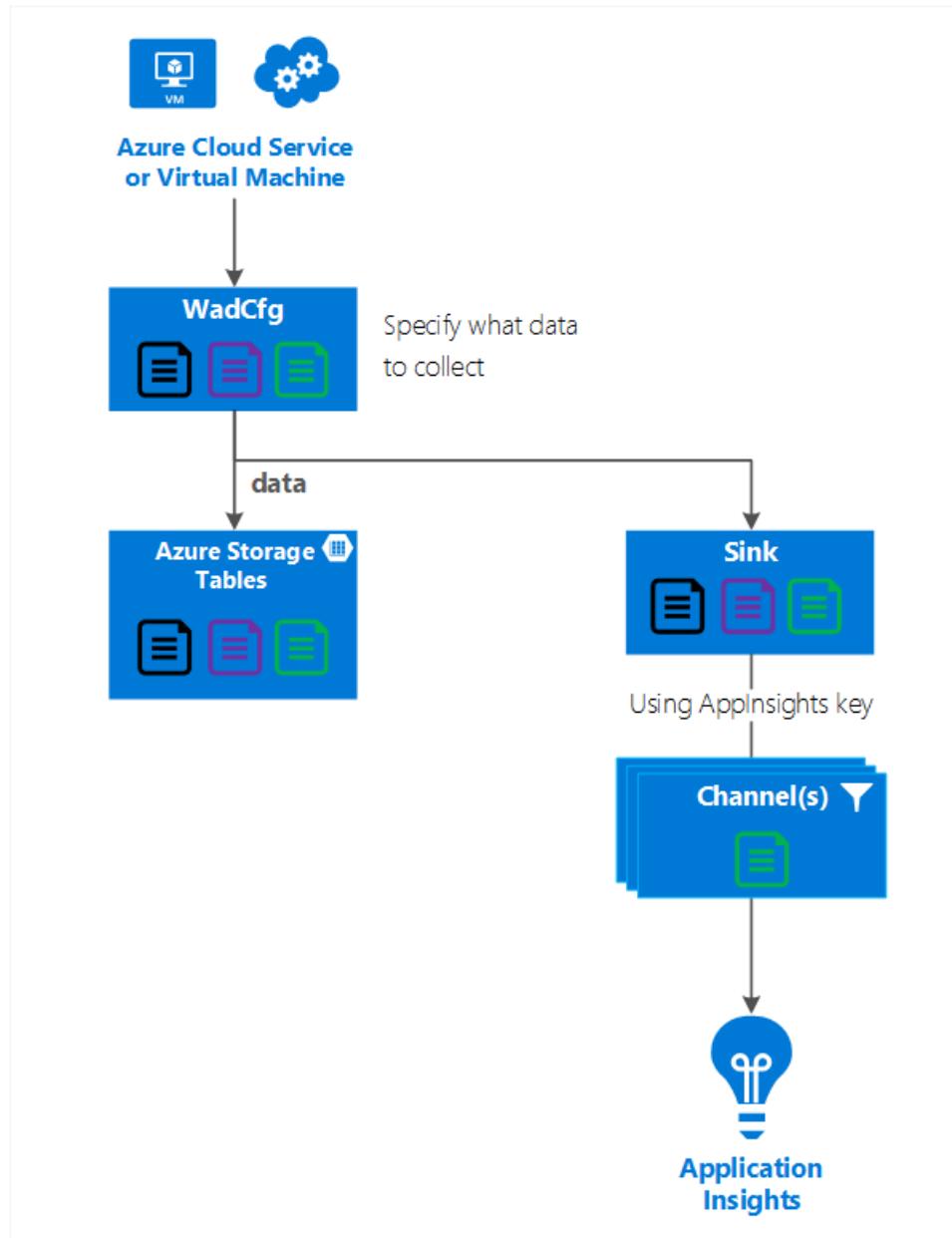
- If you are developing a Cloud Service with Azure SDK 2.8 and later, this instrumentation key is automatically populated. The value is based on the **APPINSIGHTS_INSTRUMENTATIONKEY** service configuration setting when packaging the Cloud Service project. See [Use Application Insights with Azure Diagnostics to troubleshoot Cloud Service issues](#).

- The **Channels** element contains one or more **Channel** elements.

- The *name* attribute uniquely refers to that channel.
- The *loglevel* attribute lets you specify the log level that the channel allows. The available log levels in order of most to least information are:
 - Verbose
 - Information
 - Warning
 - Error
 - Critical

A channel acts like a filter and allows you to select specific log levels to send to the target sink. For example, you could collect verbose logs and send them to storage, but send only Errors to the sink.

The following graphic shows this relationship.



The following graphic summarizes the configuration values and how they work. You can include multiple sinks in

the configuration at different levels in the hierarchy. The sink at the top level acts as a global setting and the one specified at the individual element acts like an override to that global setting.

```
<WadCfg>
  <DiagnosticMonitorConfiguration overallQuotaInMB="4096" sinks="ApplicationInsights.MyErrors">
    <DiagnosticInfrastructureLogs />

    <Directories>
      <IISLogs containerName="wad-iis-logfiles" />
    </Directories>

    <PerformanceCounters>
      <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Processor Time" sampleRate="PT3M" />
      <PerformanceCounterConfiguration counterSpecifier="\Memory\Available MBytes" sampleRate="PT3M" />
      <PerformanceCounterConfiguration counterSpecifier="\Web Service(_Total)\Bytes Total/Sec" sampleRate="PT3M" />
    </PerformanceCounters>

    <WindowsEventLog scheduledTransferPeriod="PT1M">
      <DataSource name="Application!*" />
    </WindowsEventLog>

    <Logs scheduledTransferPeriod="PT1M" scheduledTransferLogLevelFilter="Verbose"
          sinks="ApplicationInsights.MyAppLogs"/>
  </DiagnosticMonitorConfiguration>

  <SinksConfig>
    <Sink name="ApplicationInsights">
      <!-- Your personal Instrumentation Key below -->
      <ApplicationInsights>12345678-1234-1234-1234-123456789abc</ApplicationInsights>
      <Channels>
        <Channel logLevel="Error" name="MyErrors" />
        <Channel logLevel="Verbose" name="MyAppLogs" />
      </Channels>
    </Sink>
  </SinksConfig>
</WadCfg>
```

Complete sink configuration example

Here is a complete example of the public configuration file that

1. sends all errors to Application Insights (specified at the **DiagnosticMonitorConfiguration** node)
2. also sends Verbose level logs for the Application Logs (specified at the **Logs** node).

```
<WadCfg>
  <DiagnosticMonitorConfiguration overallQuotaInMB="4096"
    sinks="ApplicationInsights.MyTopDiagData"> <!-- All info below sent to this channel -->
    <DiagnosticInfrastructureLogs />
    <PerformanceCounters>
      <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Processor Time"
sampleRate="PT3M" />
      <PerformanceCounterConfiguration counterSpecifier="\Memory\Available MBytes" sampleRate="PT3M" />
    </PerformanceCounters>
    <WindowsEventLog scheduledTransferPeriod="PT1M">
      <DataSource name="Application!*" />
    </WindowsEventLog>
    <Logs scheduledTransferPeriod="PT1M" scheduledTransferLogLevelFilter="Verbose"
      sinks="ApplicationInsights.MyLogData"/> <!-- This specific info sent to this channel -->
  </DiagnosticMonitorConfiguration>

  <SinksConfig>
    <Sink name="ApplicationInsights">
      <ApplicationInsights>{Insert InstrumentationKey}</ApplicationInsights>
      <Channels>
        <Channel logLevel="Error" name="MyTopDiagData" />
        <Channel logLevel="Verbose" name="MyLogData" />
      </Channels>
    </Sink>
  </SinksConfig>
</WadCfg>
```

```

"WadCfg": {
    "DiagnosticMonitorConfiguration": {
        "overallQuotaInMB": 4096,
        "sinks": "ApplicationInsights.MyTopDiagData", "_comment": "All info below sent to this channel",
        "DiagnosticInfrastructureLogs": {
        },
        "PerformanceCounters": {
            "PerformanceCounterConfiguration": [
                {
                    "counterSpecifier": "\Processor(_Total)\% Processor Time",
                    "sampleRate": "PT3M"
                },
                {
                    "counterSpecifier": "\Memory\Available MBytes",
                    "sampleRate": "PT3M"
                }
            ]
        },
        "WindowsEventLog": {
            "scheduledTransferPeriod": "PT1M",
            "DataSource": [
                {
                    "name": "Application!*"
                }
            ],
            "Logs": {
                "scheduledTransferPeriod": "PT1M",
                "scheduledTransferLogLevelFilter": "Verbose",
                "sinks": "ApplicationInsights.MyLogData", "_comment": "This specific info sent to this channel"
            }
        },
        "SinksConfig": {
            "Sink": [
                {
                    "name": "ApplicationInsights",
                    "ApplicationInsights": "{Insert InstrumentationKey}",
                    "Channels": {
                        "Channel": [
                            {
                                "logLevel": "Error",
                                "name": "MyTopDiagData"
                            },
                            {
                                "logLevel": "Verbose",
                                "name": "MyLogData"
                            }
                        ]
                    }
                }
            ]
        }
    }
}

```

In the previous configuration, the following lines have the following meanings:

Send all the data that is being collected by Azure diagnostics

```
<DiagnosticMonitorConfiguration overallQuotaInMB="4096" sinks="ApplicationInsights">
```

```
"DiagnosticMonitorConfiguration": {  
    "overallQuotaInMB": 4096,  
    "sinks": "ApplicationInsights",  
}
```

Send only error logs to the Application Insights sink

```
<DiagnosticMonitorConfiguration overallQuotaInMB="4096" sinks="ApplicationInsights.MyTopDiagdata">
```

```
"DiagnosticMonitorConfiguration": {  
    "overallQuotaInMB": 4096,  
    "sinks": "ApplicationInsights.MyTopDiagData",  
}
```

Send Verbose application logs to Application Insights

```
<Logs scheduledTransferPeriod="PT1M" scheduledTransferLogLevelFilter="Verbose"  
sinks="ApplicationInsights.MyLogData"/>
```

```
"DiagnosticMonitorConfiguration": {  
    "overallQuotaInMB": 4096,  
    "sinks": "ApplicationInsights.MyLogData",  
}
```

Limitations

- **Channels only log type and not performance counters.** If you specify a channel with a performance counter element, it is ignored.
- **The log level for a channel cannot exceed the log level for what is being collected by Azure diagnostics.** For example, you cannot collect Application Log errors in the Logs element and try to send Verbose logs to the Application Insight sink. The *scheduledTransferLogLevelFilter* attribute must always collect equal or more logs than the logs you are trying to send to a sink.
- **You cannot send blob data collected by Azure diagnostics extension to Application Insights.** For example, anything specified under the *Directories* node. For Crash Dumps the actual crash dump is sent to blob storage and only a notification that the crash dump was generated is sent to Application Insights.

Next Steps

- Learn how to [view your Azure diagnostics information](#) in Application Insights.
- Use [PowerShell](#) to enable the Azure diagnostics extension for your application.
- Use [Visual Studio](#) to enable the Azure diagnostics extension for your application

Streaming Azure Diagnostics data in the hot path by using Event Hubs

7/13/2017 • 11 min to read • [Edit Online](#)

Azure Diagnostics provides flexible ways to collect metrics and logs from cloud services virtual machines (VMs) and transfer results to Azure Storage. Starting in the March 2016 (SDK 2.9) time frame, you can send Diagnostics to custom data sources and transfer hot path data in seconds by using [Azure Event Hubs](#).

Supported data types include:

- Event Tracing for Windows (ETW) events
- Performance counters
- Windows event logs
- Application logs
- Azure Diagnostics infrastructure logs

This article shows you how to configure Azure Diagnostics with Event Hubs from end to end. Guidance is also provided for the following common scenarios:

- How to customize the logs and metrics that get sent to Event Hubs
- How to change configurations in each environment
- How to view Event Hubs stream data
- How to troubleshoot the connection

Prerequisites

Event Hubs receiving data from Azure Diagnostics is supported in Cloud Services, VMs, Virtual Machine Scale Sets, and Service Fabric starting in the Azure SDK 2.9 and the corresponding Azure Tools for Visual Studio.

- Azure Diagnostics extension 1.6 ([Azure SDK for .NET 2.9 or later](#) targets this by default)
- [Visual Studio 2013 or later](#)
- Existing configurations of Azure Diagnostics in an application by using a `.wadcfgx` file and one of the following methods:
 - Visual Studio: [Configuring Diagnostics for Azure Cloud Services and Virtual Machines](#)
 - Windows PowerShell: [Enable diagnostics in Azure Cloud Services using PowerShell](#)
- Event Hubs namespace provisioned per the article, [Get started with Event Hubs](#)

Connect Azure Diagnostics to Event Hubs sink

By default, Azure Diagnostics always sends logs and metrics to an Azure Storage account. An application may also send data to Event Hubs by adding a new **Sinks** section under the **PublicConfig / WadCfg** element of the `.wadcfgx` file. In Visual Studio, the `.wadcfgx` file is stored in the following path: **Cloud Service Project > Roles > (RoleName) > diagnostics.wadcfgx** file.

```

<SinksConfig>
  <Sink name="HotPath">
    <EventHub Url="https://diags-mycompany-ns.servicebus.windows.net/diageventhub"
      SharedAccessKeyName="SendRule" />
  </Sink>
</SinksConfig>

```

```

"SinksConfig": {
  "Sink": [
    {
      "name": "HotPath",
      "EventHub": {
        "Url": "https://diags-mycompany-ns.servicebus.windows.net/diageventhub",
        "SharedAccessKeyName": "SendRule"
      }
    }
  ]
}

```

In this example, the event hub URL is set to the fully qualified namespace of the event hub: Event Hubs namespace + "/" + event hub name.

The event hub URL is displayed in the [Azure portal](#) on the Event Hubs dashboard.

The **Sink** name can be set to any valid string as long as the same value is used consistently throughout the config file.

NOTE

There may be additional sinks, such as *applicationInsights* configured in this section. Azure Diagnostics allows one or more sinks to be defined if each sink is also declared in the **PrivateConfig** section.

The Event Hubs sink must also be declared and defined in the **PrivateConfig** section of the *.wadcfgx* config file.

```

<PrivateConfig xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration">
  <StorageAccount name="{account name}" key="{account key}" endpoint="{optional storage endpoint}" />
  <EventHub Url="https://diags-mycompany-ns.servicebus.windows.net/diageventhub" SharedAccessKeyName="SendRule"
    SharedAccessKey="{base64 encoded key}" />
</PrivateConfig>

```

```

{
  "storageAccountName": "{account name}",
  "storageAccountKey": "{account key}",
  "storageAccountEndPoint": "{optional storage endpoint}",
  "EventHub": {
    "Url": "https://diags-mycompany-ns.servicebus.windows.net/diageventhub",
    "SharedAccessKeyName": "SendRule",
    "SharedAccessKey": "{base64 encoded key}"
  }
}

```

The **SharedAccessKeyName** value must match a Shared Access Signature (SAS) key and policy that has been defined in the **Event Hubs** namespace. Browse to the Event Hubs dashboard in the [Azure portal](#), click the **Configure** tab, and set up a named policy (for example, "SendRule") that has *Send* permissions. The **StorageAccount** is also declared in **PrivateConfig**. There is no need to change values here if they are working. In this example, we leave the values empty, which is a sign that a downstream asset will set the values. For example, the

ServiceConfiguration.Cloud.cscfg environment configuration file sets the environment-appropriate names and keys.

WARNING

The Event Hubs SAS key is stored in plain text in the *.wadcfgx* file. Often, this key is checked in to source code control or is available as an asset in your build server, so you should protect it as appropriate. We recommend that you use a SAS key here with *Send only* permissions so that a malicious user can write to the event hub, but not listen to it or manage it.

Configure Azure Diagnostics to send logs and metrics to Event Hubs

As discussed previously, all default and custom diagnostics data, that is, metrics and logs, is automatically sent to Azure Storage in the configured intervals. With Event Hubs and any additional sink, you can specify any root or leaf node in the hierarchy to be sent to the event hub. This includes ETW events, performance counters, Windows event logs, and application logs.

It is important to consider how many data points should actually be transferred to Event Hubs. Typically, developers transfer low-latency hot-path data that must be consumed and interpreted quickly. Systems that monitor alerts or autoscale rules are examples. A developer might also configure an alternate analysis store or search store -- for example, Azure Stream Analytics, Elasticsearch, a custom monitoring system, or a favorite monitoring system from others.

The following are some example configurations.

```
<PerformanceCounters scheduledTransferPeriod="PT1M" sinks="HotPath">
    <PerformanceCounterConfiguration counterSpecifier="\Memory\Available MBytes" sampleRate="PT3M" />
    <PerformanceCounterConfiguration counterSpecifier="\Web Service(_Total)\ISAPI Extension Requests/sec"
sampleRate="PT3M" />
    <PerformanceCounterConfiguration counterSpecifier="\Web Service(_Total)\Bytes Total/Sec" sampleRate="PT3M" />
</PerformanceCounters>
```

```
"PerformanceCounters": {
    "scheduledTransferPeriod": "PT1M",
    "sinks": "HotPath",
    "PerformanceCounterConfiguration": [
        {
            "counterSpecifier": "\Processor(_Total)\% Processor Time",
            "sampleRate": "PT3M"
        },
        {
            "counterSpecifier": "\Memory\Available MBytes",
            "sampleRate": "PT3M"
        },
        {
            "counterSpecifier": "\Web Service(_Total)\ISAPI Extension Requests/sec",
            "sampleRate": "PT3M"
        }
    ]
}
```

In the above example, the sink is applied to the parent **PerformanceCounters** node in the hierarchy, which means all child **PerformanceCounters** will be sent to Event Hubs.

```

<PerformanceCounters scheduledTransferPeriod="PT1M">
    <PerformanceCounterConfiguration counterSpecifier="\Memory\Available MBytes" sampleRate="PT3M" />
    <PerformanceCounterConfiguration counterSpecifier="\Web Service(_Total)\ISAPI Extension Requests/sec"
sampleRate="PT3M" />
    <PerformanceCounterConfiguration counterSpecifier="\ASP.NET\Requests Queued" sampleRate="PT3M"
sinks="HotPath" />
    <PerformanceCounterConfiguration counterSpecifier="\ASP.NET\Requests Rejected" sampleRate="PT3M"
sinks="HotPath"/>
    <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Processor Time" sampleRate="PT3M"
sinks="HotPath"/>
</PerformanceCounters>

```

```

"PerformanceCounters": {
    "scheduledTransferPeriod": "PT1M",
    "PerformanceCounterConfiguration": [
        {
            "counterSpecifier": "\Processor(_Total)\% Processor Time",
            "sampleRate": "PT3M",
            "sinks": "HotPath"
        },
        {
            "counterSpecifier": "\Memory\Available MBytes",
            "sampleRate": "PT3M"
        },
        {
            "counterSpecifier": "\Web Service(_Total)\ISAPI Extension Requests/sec",
            "sampleRate": "PT3M"
        },
        {
            "counterSpecifier": "\ASP.NET\Requests Rejected",
            "sampleRate": "PT3M",
            "sinks": "HotPath"
        },
        {
            "counterSpecifier": "\ASP.NET\Requests Queued",
            "sampleRate": "PT3M",
            "sinks": "HotPath"
        }
    ]
}

```

In the previous example, the sink is applied to only three counters: **Requests Queued**, **Requests Rejected**, and **% Processor time**.

The following example shows how a developer can limit the amount of sent data to be the critical metrics that are used for this service's health.

```
<Logs scheduledTransferPeriod="PT1M" sinks="HotPath" scheduledTransferLogLevelFilter="Error" />
```

```

"Logs": {
    "scheduledTransferPeriod": "PT1M",
    "scheduledTransferLogLevelFilter": "Error",
    "sinks": "HotPath"
}

```

In this example, the sink is applied to logs and is filtered only to error level trace.

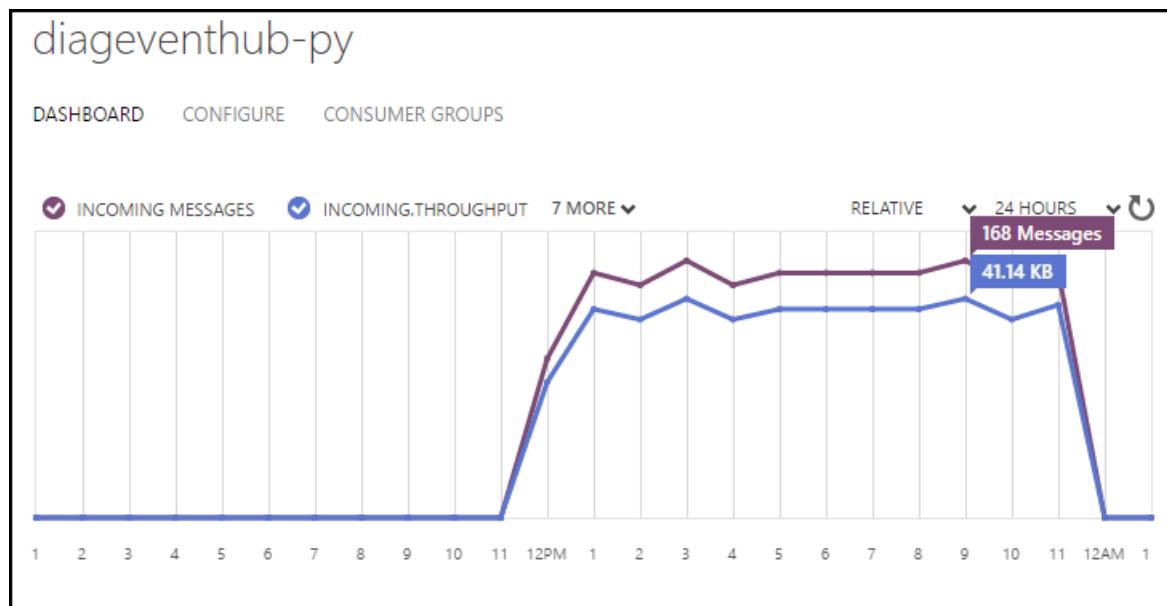
Deploy and update a Cloud Services application and diagnostics config

Visual Studio provides the easiest path to deploy the application and Event Hubs sink configuration. To view and edit the file, open the `.wadcfgx` file in Visual Studio, edit it, and save it. The path is **Cloud Service Project > Roles > (RoleName) > diagnostics.wadcfgx**.

At this point, all deployment and deployment update actions in Visual Studio, Visual Studio Team System, and all commands or scripts that are based on MSBuild and use the `/t:publish` target include the `.wadcfgx` in the packaging process. In addition, deployments and updates deploy the file to Azure by using the appropriate Azure Diagnostics agent extension on your VMs.

After you deploy the application and Azure Diagnostics configuration, you will immediately see activity in the dashboard of the event hub. This indicates that you're ready to move on to viewing the hot-path data in the listener client or analysis tool of your choice.

In the following figure, the Event Hubs dashboard shows healthy sending of diagnostics data to the event hub starting sometime after 11 PM. That's when the application was deployed with an updated `.wadcfgx` file, and the sink was configured properly.



NOTE

When you make updates to the Azure Diagnostics config file (`.wadcfgx`), it's recommended that you push the updates to the entire application as well as the configuration by using either Visual Studio publishing, or a Windows PowerShell script.

View hot-path data

As discussed previously, there are many use cases for listening to and processing Event Hubs data.

One simple approach is to create a small test console application to listen to the event hub and print the output stream. You can place the following code, which is explained in more detail in [Get started with Event Hubs](#), in a console application.

Note that the console application must include the [Event Processor Host NuGet package](#).

Remember to replace the values in angle brackets in the **Main** function with values for your resources.

```
//Console application code for EventHub test client
using System;
using System.Collections.Generic;
using System.Diagnostics;
using System.Linq;
using System.Text;
```

```

using System.Threading.Tasks;
using Microsoft.ServiceBus.Messaging;

namespace EventHubListener
{
    class SimpleEventProcessor : IEventProcessor
    {
        Stopwatch checkpointStopWatch;

        async Task IEventProcessor.CloseAsync(PartitionContext context, CloseReason reason)
        {
            Console.WriteLine("Processor Shutting Down. Partition '{0}', Reason: '{1}'.",
                context.Lease.PartitionId, reason);
            if (reason == CloseReason.Shutdown)
            {
                await context.CheckpointAsync();
            }
        }

        Task IEventProcessor.OpenAsync(PartitionContext context)
        {
            Console.WriteLine("SimpleEventProcessor initialized. Partition: '{0}', Offset: '{1}'",
                context.Lease.PartitionId, context.Lease.Offset);
            this.checkpointStopWatch = new Stopwatch();
            this.checkpointStopWatch.Start();
            return Task.FromResult<object>(null);
        }

        async Task IEventProcessor.ProcessEventsAsync(PartitionContext context, IEnumerable<EventData>
messages)
        {
            foreach (EventData eventData in messages)
            {
                string data = Encoding.UTF8.GetString(eventData.GetBytes());
                Console.WriteLine(string.Format("Message received. Partition: '{0}', Data: '{1}'",
                    context.Lease.PartitionId, data));

                foreach (var x in eventData.Properties)
                {
                    Console.WriteLine(string.Format("    {0} = {1}", x.Key, x.Value));
                }
            }

            //Call checkpoint every 5 minutes, so that worker can resume processing from 5 minutes back if it
            restarts.
            if (this.checkpointStopWatch.Elapsed > TimeSpan.FromMinutes(5))
            {
                await context.CheckpointAsync();
                this.checkpointStopWatch.Restart();
            }
        }
    }

    class Program
    {
        static void Main(string[] args)
        {
            string eventHubConnectionString = "Endpoint= <your connection string>";
            string eventHubName = "<Event hub name>";
            string storageAccountName = "<Storage account name>";
            string storageAccountKey = "<Storage account key>";
            string storageConnectionString = string.Format("DefaultEndpointsProtocol=https;AccountName=
{0};AccountKey={1}", storageAccountName, storageAccountKey);

            string eventProcessorHostName = Guid.NewGuid().ToString();
            EventProcessorHost eventProcessorHost = new EventProcessorHost(eventProcessorHostName,
                eventHubName, EventHubConsumerGroup.DefaultGroupName, eventHubConnectionString, storageConnectionString);
            Console.WriteLine("Registering EventProcessor...");
            var options = new EventProcessorOptions();
        }
    }
}

```

```

        options.ExceptionReceived += (sender, e) => { Console.WriteLine(e.Exception); };
        eventProcessorHost.RegisterEventProcessorAsync<SimpleEventProcessor>(options).Wait();

        Console.WriteLine("Receiving. Press enter key to stop worker.");
        Console.ReadLine();
        eventProcessorHost.UnregisterEventProcessorAsync().Wait();
    }
}
}

```

Troubleshoot Event Hubs sinks

- The event hub does not show incoming or outgoing event activity as expected.

Check that your event hub is successfully provisioned. All connection info in the **PrivateConfig** section of *.wadcfgx* must match the values of your resource as seen in the portal. Make sure that you have a SAS policy defined ("SendRule" in the example) in the portal and that *Send* permission is granted.

- After an update, the event hub no longer shows incoming or outgoing event activity.

First, make sure that the event hub and configuration info is correct as explained previously. Sometimes the **PrivateConfig** is reset in a deployment update. The recommended fix is to make all changes to *.wadcfgx* in the project and then push a complete application update. If this is not possible, make sure that the diagnostics update pushes a complete **PrivateConfig** that includes the SAS key.

- I tried the suggestions, and the event hub is still not working.

Try looking in the Azure Storage table that contains logs and errors for Azure Diagnostics itself:

WADDiagnosticInfrastructureLogsTable. One option is to use a tool such as [Azure Storage Explorer](#) to connect to this storage account, view this table, and add a query for TimeStamp in the last 24 hours. You can use the tool to export a .csv file and open it in an application such as Microsoft Excel. Excel makes it easy to search for calling-card strings, such as **EventHubs**, to see what error is reported.

Next steps

- [Learn more about Event Hubs](#)

Appendix: Complete Azure Diagnostics configuration file (.wadcfgx) example

```

<?xml version="1.0" encoding="utf-8"?>
<DiagnosticsConfiguration xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration">
  <PublicConfig xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration">
    <WadCfg>
      <DiagnosticMonitorConfiguration overallQuotaInMB="4096" sinks="applicationInsights.errors">
        <DiagnosticInfrastructureLogs scheduledTransferLogLevelFilter="Error" />
        <Directories scheduledTransferPeriod="PT1M">
          <IISLogs containerName="wad-iis-logfiles" />
          <FailedRequestLogs containerName="wad-failedrequestlogs" />
        </Directories>
        <PerformanceCounters scheduledTransferPeriod="PT1M" sinks="HotPath">
          <PerformanceCounterConfiguration counterSpecifier="\Memory\Available MBytes" sampleRate="PT3M" />
          <PerformanceCounterConfiguration counterSpecifier="\Web Service(_Total)\ISAPI Extension Requests/sec" sampleRate="PT3M" />
          <PerformanceCounterConfiguration counterSpecifier="\Web Service(_Total)\Bytes Total/Sec" sampleRate="PT3M" />
          <PerformanceCounterConfiguration counterSpecifier="\ASP.NET Applications(__Total__)\Requests/Sec" sampleRate="PT3M" />
          <PerformanceCounterConfiguration counterSpecifier="\ASP.NET Applications(__Total__)\Errors Total/Sec" sampleRate="PT3M" />
          <PerformanceCounterConfiguration counterSpecifier="\ASP.NET\Requests Queued" sampleRate="PT3M" />
          <PerformanceCounterConfiguration counterSpecifier="\ASP.NET\Requests Rejected" sampleRate="PT3M" />
          <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Processor Time" sampleRate="PT3M" />
        </PerformanceCounters>
        <WindowsEventLog scheduledTransferPeriod="PT1M">
          <DataSource name="Application!*" />
        </WindowsEventLog>
        <CrashDumps>
          <CrashDumpConfiguration processName="WaIISHost.exe" />
          <CrashDumpConfiguration processName="WaWorkerHost.exe" />
          <CrashDumpConfiguration processName="w3wp.exe" />
        </CrashDumps>
        <Logs scheduledTransferPeriod="PT1M" sinks="HotPath" scheduledTransferLogLevelFilter="Error" />
      </DiagnosticMonitorConfiguration>
      <SinksConfig>
        <Sink name="HotPath">
          <EventHub Url="https://diageventhub-py-ns.servicebus.windows.net/diageventhub-py" SharedAccessKeyName="SendRule" />
        </Sink>
        <Sink name="applicationInsights">
          <ApplicationInsights />
          <Channels>
            <Channel logLevel="Error" name="errors" />
          </Channels>
        </Sink>
      </SinksConfig>
    </WadCfg>
    <StorageAccount>ACCOUNT_NAME</StorageAccount>
  </PublicConfig>
  <PrivateConfig xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration">
    <StorageAccount name="{account name}" key="{account key}" endpoint="{storage endpoint}" />
    <EventHub Url="https://diageventhub-py-ns.servicebus.windows.net/diageventhub-py" SharedAccessKeyName="SendRule" SharedAccessKey="YOUR_KEY_HERE" />
  </PrivateConfig>
  <IsEnabled>true</IsEnabled>
</DiagnosticsConfiguration>

```

The complementary *ServiceConfiguration.Cloud.cscfg* for this example looks like the following.

```

<?xml version="1.0" encoding="utf-8"?>
<ServiceConfiguration serviceName="MyFixItCloudService"
xmlns="http://schemas.microsoft.com/ServiceHosting/2008/10/ServiceConfiguration" osFamily="3" osVersion="*"
schemaVersion="2015-04-2.6">
  <Role name="MyFixIt.WorkerRole">
    <Instances count="1" />
    <ConfigurationSettings>
      <Setting name="Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString"
value="YOUR_CONNECTION_STRING" />
    </ConfigurationSettings>
  </Role>
</ServiceConfiguration>

```

Equivalent Json based settings for virtual machines is as follows:

```

"settings": {
  "WadCfg": {
    "DiagnosticMonitorConfiguration": {
      "overallQuotaInMB": 4096,
      "sinks": "applicationInsights.errors",
      "DiagnosticInfrastructureLogs": {
        "scheduledTransferLogLevelFilter": "Error"
      },
      "Directories": {
        "scheduledTransferPeriod": "PT1M",
        "IISLogs": {
          "containerName": "wad-iis-logfiles"
        },
        "FailedRequestLogs": {
          "containerName": "wad-failedrequestlogs"
        }
      },
      "PerformanceCounters": {
        "scheduledTransferPeriod": "PT1M",
        "sinks": "HotPath",
        "PerformanceCounterConfiguration": [
          {
            "counterSpecifier": "\Memory\Available MBytes",
            "sampleRate": "PT3M"
          },
          {
            "counterSpecifier": "\Web Service(_Total)\ISAPI Extension Requests/sec",
            "sampleRate": "PT3M"
          },
          {
            "counterSpecifier": "\Web Service(_Total)\Bytes Total/Sec",
            "sampleRate": "PT3M"
          },
          {
            "counterSpecifier": "\ASP.NET Applications(__Total__)\Requests/Sec",
            "sampleRate": "PT3M"
          },
          {
            "counterSpecifier": "\ASP.NET Applications(__Total__)\Errors Total/Sec",
            "sampleRate": "PT3M"
          },
          {
            "counterSpecifier": "\ASP.NET\Requests Queued",
            "sampleRate": "PT3M"
          },
          {
            "counterSpecifier": "\ASP.NET\Requests Rejected",
            "sampleRate": "PT3M"
          },
          {
            "counterSpecifier": "\Processor(_Total)\% Processor Time",
            "sampleRate": "PT3M"
          }
        ]
      }
    }
  }
}

```

```

        "sampleRate": "PT3M"
    }
]
},
"WindowsEventLog": {
    "scheduledTransferPeriod": "PT1M",
    "DataSource": [
        {
            "name": "Application!*"
        }
    ]
},
"Logs": {
    "scheduledTransferPeriod": "PT1M",
    "scheduledTransferLogLevelFilter": "Error",
    "sinks": "HotPath"
}
},
"SinksConfig": {
    "Sink": [
        {
            "name": "HotPath",
            "EventHub": {
                "Url": "https://diageventhub-py-ns.servicebus.windows.net/diageventhub-py",
                "SharedAccessKeyName": "SendRule"
            }
        },
        {
            "name": "applicationInsights",
            "ApplicationInsights": "",
            "Channels": {
                "Channel": [
                    {
                        "logLevel": "Error",
                        "name": "errors"
                    }
                ]
            }
        }
    ]
}
},
"StorageAccount": "{account name}"
}

"protectedSettings": {
    "storageAccountName": "{account name}",
    "storageAccountKey": "{account key}",
    "storageAccountEndPoint": "{storage endpoint}",
    "EventHub": {
        "Url": "https://diageventhub-py-ns.servicebus.windows.net/diageventhub-py",
        "SharedAccessKeyName": "SendRule",
        "SharedAccessKey": "YOUR_KEY_HERE"
    }
}
}

```

Next steps

You can learn more about Event Hubs by visiting the following links:

- [Event Hubs overview](#)
- [Create an event hub](#)
- [Event Hubs FAQ](#)

Azure Diagnostics Troubleshooting

7/13/2017 • 11 min to read • [Edit Online](#)

Troubleshooting information relevant to using Azure Diagnostics. For more information on Azure diagnostics, see [Azure Diagnostics Overview](#).

Logical Components

Diagnostics Plugin Launcher (DiagnosticsPluginLauncher.exe): Launches the Azure Diagnostics extension. Serves as the entry point process.

Diagnostics Plugin (DiagnosticsPlugin.exe): Main process that is launched by the launcher above and configures the Monitoring Agent, launches it and manages its lifetime.

Monitoring Agent (MonAgent*.exe processes): These processes do the bulk of the work; i.e., monitoring, collection and transfer of the diagnostics data.

Log/Artifact Paths

Here are the paths to some important logs and artifacts. We keep referring to these throughout the rest of the document:

Cloud Services

ARTIFACT	PATH
Azure Diagnostics configuration file	%SystemDrive%\Packages\Plugins\Microsoft.Azure.Diagnostics.PaaSDiagnostics<version>\Config.txt
Log files	C:\Logs\Plugins\Microsoft.Azure.Diagnostics.PaaSDiagnostics<version>\
Local store for diagnostics data	C:\Resources\Directory<CloudServiceDeploymentID>. <RoleName>.DiagnosticStore\WAD0107\Tables
Monitoring Agent configuration file	C:\Resources\Directory<CloudServiceDeploymentID>. <RoleName>.DiagnosticStore\WAD0107\Configuration\MonAgentHost\config.xml
Azure diagnostics extension package	%SystemDrive%\Packages\Plugins\Microsoft.Azure.Diagnostics.PaaSDiagnostics<version>
Log collection utility path	%SystemDrive%\Packages\GuestAgent\
MonAgentHost log file	C:\Resources\Directory<CloudServiceDeploymentID>. <RoleName>.DiagnosticStore\WAD0107\Configuration\MonAgentHost..log

Virtual Machines

ARTIFACT	PATH
Azure Diagnostics configuration file	C:\Packages\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnosics<version>\RuntimeSettings
Log files	C:\Packages\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnosics<version>\Logs\
Local store for diagnostics data	C:\WindowsAzure\Logs\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnostics<DiagnosticsVersion>\WAD0107\Tables
Monitoring Agent configuration file	C:\WindowsAzure\Logs\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnostics<DiagnosticsVersion>\WAD0107\Configuration\MaConfig.xml
Status file	C:\Packages\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnosics<version>\Status
Azure diagnostics extension package	C:\Packages\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnosics<DiagnosticsVersion>
Log collection utility path	C:\WindowsAzure\Packages
MonAgentHost log file	C:\WindowsAzure\Logs\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnostics<DiagnosticsVersion>\WAD0107\Configuration\MonAgentHost..log

Azure Diagnostics is not Starting

Look at **DiagnosticsPluginLauncher.log** and **DiagnosticsPlugin.log** files from the location of the log files provided above for information on why diagnostics failed to start.

If these logs indicate `Monitoring Agent not reporting success after launch`, it means there was a failure launching MonAgentHost.exe. Look at the logs for that in the location indicated for **MonAgentHost log file** in the section above.

The last line of the log files contains the exit code.

```
DiagnosticsPluginLauncher.exe Information: 0 : [4/16/2016 6:24:15 AM] DiagnosticPlugin exited with code 0
```

If you find a **negative** exit code, refer to the [exit code table](#) in the [References](#).

Diagnostics Data is Not Logged to Azure Storage

Determine if no data is showing up or only some of the data is not showing up.

Diagnostics Infrastructure Logs

Diagnostics Infrastructure Logs is where azure diagnostics logs any errors that it runs into. Make sure you have enabled ([how to?](#)) capturing of Diagnositcs Infrastructure logs in your configuration and quickly look for any relevant errors that show up in the `DiagnosticInfrastructureLogsTable` table in your configured storage account.

No data is showing up

The most common cause of event data entirely missing is incorrectly defined storage account information.

Solution: Correct your Diagnostics configuration and reinstall Diagnostics.

If the storage account is configured correctly, remote desktop into the machine and make sure DiagnosticsPlugin.exe and MonAgentCore.exe are running. If they are not running follow [Azure Diagnostics is not Starting](#). If the processes are running, jump to [Is data getting captured locally](#) and follow this guide from there on.

Part of the data is missing

If you are getting some data but not other. This means the data collection / transfer pipeline is set correctly. Follow the sub-sections here to narrow down what the issue is:

Is Collection Configured:

Diagnostics Configuration contains the part that instructs for a particular type of data to be collected. [Review your configuration](#) to make sure you are not looking for data you have not configured for collection.

Is the host generating data:

- **Performance Counters:** open perfmon and check the counter.
- **Trace Logs:** Remote desktop into the VM and add a TextWriterTraceListener to the app's config file. See <http://msdn.microsoft.com/en-us/library/sk36c28t.aspx> to setup the text listener. Make sure the `<trace>` element has `<trace autoflush="true">`.
If you don't see trace logs getting generated, follow [More About Trace Logs Missing](#).
- **ETW traces:** Remote desktop into the VM and install PerfView. In PerfView run File -> User Command -> Listen etwprovider1,etwprovider2,etc. Note that the Listen command is case sensitive and there cannot be spaces between the comma separated list of ETW providers. If the command fails to run you can click the 'Log' button in the bottom-right of the Perfview tool to see what was attempted to run and what the result was. Assuming the input is correct then a new window will pop up and in a few seconds you will begin seeing ETW traces.
- **Event Logs:** Remote desktop into the VM. Open `Event Viewer` and ensure the events exist. ##### Is data getting captured locally: Next make sure the data is getting captured locally. The data is locally stored in `*.tsf` files in [the local store for diagnostics data](#). Different kinds of logs get collected in different `.tsf` files. The names are similar to the table names in azure storage. For example `Performance Counters` get collected in `PerformanceCountersTable.tsf`, Event Logs get collected in `WindowsEventLogsTable.tsf`. Use the instructions in [Local Log Extraction](#) section to open the local collection files and make sure you see them getting collected on disk.

If you don't see logs getting collected locally and are have already verified that the host is generating data, you likely have a configuration issue. Review your configuration carefully for the appropriate section. Also review the configuration generated for MonitoringAgent [MaConfig.xml](#) and make sure there is some section describing the relevant log source and that it is not lost in translation between azure diagnostics configuration and monitoring agent configuration.

Is data getting transferred:

If you have verified the data is getting captured locally but you still don't see it in your storage account:

- First and foremost, make sure that you have provided a correct storage account and that you have not rolled over keys etc. for the given storage account. For cloud services, sometimes we see that people don't update their `useDevelopmentStorage=true`.
- If provided storage account is correct. Make sure you do not have some network restrictions that do not allow the components to reach public storage endpoints. One way to do that is to remote desktop into the machine and try to write something to the same storage account yourself.
- Finally, you can try and see what failure are being reported by Monitoring Agent. Monitoring agent writes its logs in `maeventtable.tsf` located in [the local store for diagnostics data](#). Follow instructions in [Local Log Extraction](#) section to open this file and try and figure out if there are `errors` indicating failures to read local files or write to storage.

Capturing / Archiving logs

You went through the above steps but could not figure out what was wrong and are thinking about contacting support. The first thing they might ask you is to collect logs from you machine. You can save time by doing that

yourself. Run the `CollectGuestLogs.exe` utility at [Log Collection Utility path](#) and it will generate a zip file with all relevant azure logs in the same folder.

Diagnostics data Tables not found

The tables in Azure storage holding ETW events are named using the following code:

```
if (String.IsNullOrEmpty(eventDestination)) {
    if (e == "DefaultEvents")
        tableName = "WADDefault" + MD5(provider);
    else
        tableName = "WADEvent" + MD5(provider) + eventId;
}
else
    tableName = "WAD" + eventDestination;
```

Here is an example:

```
<EtwEventSourceProviderConfiguration provider="prov1">
    <Event id="1" />
    <Event id="2" eventDestination="dest1" />
    <DefaultEvents />
</EtwEventSourceProviderConfiguration>
<EtwEventSourceProviderConfiguration provider="prov2">
    <DefaultEvents eventDestination="dest2" />
</EtwEventSourceProviderConfiguration>
```

```
"EtwEventSourceProviderConfiguration": [
    {
        "provider": "prov1",
        "Event": [
            {
                "id": 1
            },
            {
                "id": 2,
                "eventDestination": "dest1"
            }
        ],
        "DefaultEvents": {
            "eventDestination": "DefaultEventDestination",
            "sinks": ""
        }
    },
    {
        "provider": "prov2",
        "DefaultEvents": {
            "eventDestination": "dest2"
        }
    }
]
```

That generates 4 tables:

EVENT	TABLE NAME
provider="prov1" <Event id="1" />	WADEvent+MD5("prov1")+"1"
provider="prov1" <Event id="2" eventDestination="dest1" />	WADdest1

EVENT	TABLE NAME
provider="prov1" <DefaultEvents />	WADDefault+MD5("prov1")
provider="prov2" <DefaultEvents eventDestination="dest2" />	WADdest2

References

How to check Diagnostics Extension Configuration

The easiest way to check your extension configuration is to navigate to <http://resources.azure.com>, navigate to the virtual machine or cloud service on which the Azure Diagnostics extension (IaaSDiagnostics / PaaDiagnostics) is in question.

Alternatively, remote desktop into the machine and look at the Azure Diagnostics Configuration file described in the appropriate section [here](#).

In either case search for **Microsoft.Azure.Diagnostics** then for the **xmlCfg** or **WadCfg** field.

In case of virtual machines, if the WadCfg field is present, it means the config is in JSON. If the xmlCfg field is present, it means the config is in XML and is base64 encoded. You need to [decode it](#) to see the XML that was loaded by Diagnostics.

For Cloud Service role, if you pick the configuration from disk, the data is base64 encoded so you'll need to [decode it](#) to see the XML that was loaded by Diagnostics.

Azure Diagnostics Plugin Exit Codes

The plugin returns the following exit codes:

EXIT CODE	DESCRIPTION
0	Success.
-1	Generic Error.
-2	Unable to load the rcf file. This internal error should only happen if the guest agent plugin launcher is manually invoked, incorrectly, on the VM.
-3	Cannot load the Diagnostics configuration file. Solution: Caused by a configuration file not passing schema validation. The solution is to provide a configuration file that complies with the schema.
-4	Another instance of the monitoring agent Diagnostics is already using the local resource directory. Solution: Specify a different value for LocalResourceDirectory .

EXIT CODE	DESCRIPTION
-6	<p>The guest agent plugin launcher attempted to launch Diagnostics with an invalid command line.</p> <p>This internal error should only happen if the guest agent plugin launcher is manually invoked, incorrectly, on the VM.</p>
-10	The Diagnostics plugin exited with an unhandled exception.
-11	<p>The guest agent was unable to create the process responsible for launching and monitoring the monitoring agent.</p> <p>Solution: Verify that sufficient system resources are available to launch new processes.</p>
-101	<p>Invalid arguments when calling the Diagnostics plugin.</p> <p>This internal error should only happen if the guest agent plugin launcher is manually invoked, incorrectly, on the VM.</p>
-102	<p>The plugin process is unable to initialize itself.</p> <p>Solution: Verify that sufficient system resources are available to launch new processes.</p>
-103	<p>The plugin process is unable to initialize itself. Specifically it is unable to create the logger object.</p> <p>Solution: Verify that sufficient system resources are available to launch new processes.</p>
-104	<p>Unable to load the rcf file provided by the guest agent.</p> <p>This internal error should only happen if the guest agent plugin launcher is manually invoked, incorrectly, on the VM.</p>
-105	<p>The Diagnostics plugin cannot open the Diagnostics configuration file.</p> <p>This internal error should only happen if the Diagnostics plugin is manually invoked, incorrectly, on the VM.</p>
-106	<p>Cannot read the Diagnostics configuration file.</p> <p>Solution: Caused by a configuration file not passing schema validation. So the solution is to provide a configuration file that complies with the schema. See How to check Diagnostics Extension Configuration.</p>
-107	<p>The resource directory pass to the monitoring agent is invalid.</p> <p>This internal error should only happen if the monitoring agent is manually invoked, incorrectly, on the VM.</p>

EXIT CODE	DESCRIPTION
-108	<p>Unable to convert the Diagnostics configuration file into the monitoring agent configuration file.</p> <p>This internal error should only happen if the Diagnostics plugin is manually invoked with an invalid configuration file.</p>
-110	<p>General Diagnostics configuration error.</p> <p>This internal error should only happen if the Diagnostics plugin is manually invoked with an invalid configuration file.</p>
-111	<p>Unable to start the monitoring agent.</p> <p>Solution: Verify that sufficient system resources are available.</p>
-112	General error

Local Log Extraction

The monitoring agent collects logs and artifacts as `.tsf` files. `.tsf` file is not readable but you can convert it into a `.csv` as follows:

```
<Azure diagnostics extension package>\Monitor\x64\table2csv.exe <relevantLogFile>.tsf
```

A new file called `<relevantLogFile>.csv` will be created in the same path as the corresponding `.tsf` file.

NOTE: You only need to run this utility against the main tsf file (e.g., PerformanceCountersTable.tsf). The accompanying files (e.g., PerformanceCountersTables_**001.tsf, PerformanceCountersTables_**002.tsf etc.) will automatically be processed.

More About Trace Logs Missing

Note: This mostly applies to cloud services only unless you have configured the `DiagnosticsMonitorTraceListener` on an application running on your IaaS VM.

- Make sure the `DiagnosticMonitorTraceListener` is configured in the `web.config` or `app.config`. This is configured by default in cloud service projects, but some customers comment it out which will cause the trace statements to not be collected by diagnostics.
- If logs are not getting written from the `OnStart` or `Run` method make sure the `DiagnosticMonitorTraceListener` is in the `app.config`. By default it is in the `web.config`, but that only applies to code running within `w3wp.exe`; so you need it in `app.config` to capture traces running in `W3SCHost.exe`.
- Make sure you are using `Diagnostics.Trace.TraceXXX` instead of `Diagnostics.Debug.WriteLineXXX`. The `Debug` statements will be removed from a Release build.
- Make sure the compiled code actually has the `Diagnostics.Trace` lines (use Reflector, ildasm or ILSpy to verify). `Diagnostics.Trace` commands are removed from the compiled binary unless you use the `TRACE` conditional compilation symbol. If using msbuild to build the project then this is a common problem to run into.

Known Issues and Mitigations

Here is a list of known issues with known mitigations:

1. .NET 4.5 dependency:

WAD has a runtime dependency on .NET 4.5 framework or above. At the time of writing this, all machines provisioned for cloud services as well as all official azure Virtual Machine base images have .NET 4.5 or above installed. It is still however possible to land in a situation where you try to run WAD on a machine which does not have .NET 4.5 or above. This happens when you create your machine off of an old image or snapshot; or bring your own custom disk.

This generally manifests as an exit code **255** when running `DiagnosticsPluginLauncher.exe`. Failure happens due to the unhandled exception:

```
System.IO.FileLoadException: Could not load file or assembly 'System.Threading.Tasks, Version=1.5.11.0, Culture=neutral, PublicKeyToken=b03f5f7f11d50a3a' or one of its dependencies
```

Mitigation: Install .NET 4.5 or higher on your machine.

2. Performance Counters data available in storage but not showing in portal

Virtual machines portal experience shows certain performance counters by default. If you don't see them and you know the data is getting generated because it is available in storage. Check:

- If the data in storage has english counter names. If the counter names are not in english, portal metric chart will not be able to recognize it.
- If you are using wild cards (*) in your performance counter names, the portal will not be able to correlate the configured and collected counter.

Mitigation: Change the machine's language to english for system accounts. Control Panel -> Region -> Administrative -> Copy Settings -> uncheck "Welcome screen and system accounts" so that the custom language is not applied to system account. Also make sure you do not use wild cards if you want portal to be your primary consumption experience.

Consume monitoring data from Azure

7/18/2017 • 1 min to read • [Edit Online](#)

Across the Azure platform, we are bringing together monitoring data in a single place with the Azure Monitor pipeline, but practically acknowledge that today not all monitoring data is available in that pipeline yet. In this article, we will summarize the various ways you can programmatically access monitoring data from Azure services.

Options for data consumption

DATA TYPE	CATEGORY	SUPPORTED SERVICES	METHODS OF ACCESS
Azure Monitor platform-level metrics	Metrics	See list here	<ul style="list-style-type: none">• REST API: Azure Monitor Metric API• Storage blob or event hub: Diagnostic Settings
Compute guest OS metrics (eg. perf counters)	Metrics	Windows and Linux Virtual Machines (v2), Cloud Services , Service Fabric	<ul style="list-style-type: none">• Storage table or blob: Windows or Linux Azure diagnostics• Event hub: Windows Azure diagnostics
Custom or application metrics	Metrics	Any application instrumented with Application Insights	<ul style="list-style-type: none">• REST API: Application Insights REST API
Storage metrics	Metrics	Azure Storage	<ul style="list-style-type: none">• Storage table: Storage Analytics
Billing data	Metrics	All Azure services	<ul style="list-style-type: none">• REST API: Azure Resource Usage and RateCard APIs
Activity Log	Events	All Azure services	<ul style="list-style-type: none">• REST API: Azure Monitor Events API• Storage blob or event hub: Log Profile
Azure Monitor Diagnostic Logs	Events	See list here	<ul style="list-style-type: none">• Storage blob or event hub: Diagnostic Settings

Data Type	Category	Supported Services	Methods of Access
Compute guest OS logs (eg. IIS, ETW, syslogs)	Events	Windows and Linux Virtual Machines (v2), Cloud Services, Service Fabric	<ul style="list-style-type: none"> • Storage table or blob: Windows or Linux Azure diagnostics • Event hub: Windows Azure diagnostics
App Service logs	Events	App services	<ul style="list-style-type: none"> • File, table, or blob storage: Web app diagnostics
Storage logs	Events	Azure Storage	<ul style="list-style-type: none"> • Storage table: Storage Analytics
Security Center alerts	Events	Azure Security Center	<ul style="list-style-type: none"> • REST API: Security Alerts
Active Directory reporting	Events	Azure Active Directory	<ul style="list-style-type: none"> • REST API: Azure Active Directory graph API
Security Center resource status	Status	All supported resources	<ul style="list-style-type: none"> • REST API: Security Statuses
Resource Health	Status	Supported services	<ul style="list-style-type: none"> • REST API: Resource health REST API
Azure Monitor metric alerts	Notifications	See list here	<ul style="list-style-type: none"> • Webhook: Azure metric alerts
Azure Monitor Activity Log alerts	Notifications	All Azure services	<ul style="list-style-type: none"> • Webhook: Azure Activity Log alerts
Autoscale notifications	Notifications	See list here	<ul style="list-style-type: none"> • Webhook: Autoscale notification webhook payload schema
OMS Log Search Query alerts	Notifications	OMS Log Analytics	<ul style="list-style-type: none"> • Webhook: Log Analytics alerts
Application Insights metric alerts	Notifications	Application Insights	<ul style="list-style-type: none"> • Webhook: Application Insights alerts

DATA TYPE	CATEGORY	SUPPORTED SERVICES	METHODS OF ACCESS
Application Insights web tests	Notifications	Application Insights	<ul style="list-style-type: none">• Webhook: Application Insights alerts

Next steps

- Learn more about [Azure Monitor metrics](#)
- Learn more about [the Azure Activity Log](#)
- Learn more about [Azure Diagnostic Logs](#)

Supported metrics with Azure Monitor

7/18/2017 • 38 min to read • [Edit Online](#)

Azure Monitor provides several ways to interact with metrics, including charting them in the portal, accessing them through the REST API, or querying them using PowerShell or CLI. Below is a complete list of all metrics currently available with Azure Monitor's metric pipeline.

NOTE

Other metrics may be available in the portal or using legacy APIs. This list only includes public preview metrics available using the public preview of the consolidated Azure Monitor metric pipeline.

Microsoft.AnalysisServices/servers

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
qpu_metric	QPU	Count	Average	QPU. Range 0-100 for S1, 0-200 for S2 and 0-400 for S4
memory_metric	Memory	Bytes	Average	Memory. Range 0-25 GB for S1, 0-50 GB for S2 and 0-100 GB for S4
TotalConnectionRequests	Total Connection Requests	Count	Average	Total connection requests. These are arrivals.
SuccessfulConnectionsPerSec	Successful Connections Per Sec	CountPerSecond	Average	Rate of successful connection completions.
TotalConnectionFailures	Total Connection Failures	Count	Average	Total failed connection attempts.
CurrentUserSessions	Current User Sessions	Count	Average	Current number of user sessions established.
QueryPoolBusyThreads	Query Pool Busy Threads	Count	Average	Number of busy threads in the query thread pool.
CommandPoolJobQueueLength	Command Pool Job Queue Length	Count	Average	Number of jobs in the queue of the command thread pool.

Metric	Metric Display Name	Unit	Aggregation Type	Description
ProcessingPoolJobQueueLength	Processing Pool Job Queue Length	Count	Average	Number of non-I/O jobs in the queue of the processing thread pool.
CurrentConnections	Connection: Current connections	Count	Average	Current number of client connections established.
CleanerCurrentPrice	Memory: Cleaner Current Price	Count	Average	Current price of memory, \$/byte/time, normalized to 1000.
CleanerMemoryShrinkable	Memory: Cleaner Memory shrinkable	Bytes	Average	Amount of memory, in bytes, subject to purging by the background cleaner.
CleanerMemoryNonshrinkable	Memory: Cleaner Memory nonshrinkable	Bytes	Average	Amount of memory, in bytes, not subject to purging by the background cleaner.
MemoryUsage	Memory: Memory Usage	Bytes	Average	Memory usage of the server process as used in calculating cleaner memory price. Equal to counter Process\PrivateBytes plus the size of memory-mapped data, ignoring any memory which was mapped or allocated by the xVelocity in-memory analytics engine (VertiPaq) in excess of the xVelocity engine Memory Limit.
MemoryLimitHard	Memory: Memory Limit Hard	Bytes	Average	Hard memory limit, from configuration file.
MemoryLimitHigh	Memory: Memory Limit High	Bytes	Average	High memory limit, from configuration file.
MemoryLimitLow	Memory: Memory Limit Low	Bytes	Average	Low memory limit, from configuration file.
MemoryLimitVertiPaq	Memory: Memory Limit VertiPaq	Bytes	Average	In-memory limit, from configuration file.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
Quota	Memory: Quota	Bytes	Average	Current memory quota, in bytes. Memory quota is also known as a memory grant or memory reservation.
QuotaBlocked	Memory: Quota Blocked	Count	Average	Current number of quota requests that are blocked until other memory quotas are freed.
VertiPaqNonpaged	Memory: VertiPaq Nonpaged	Bytes	Average	Bytes of memory locked in the working set for use by the in-memory engine.
VertiPaqPaged	Memory: VertiPaq Paged	Bytes	Average	Bytes of paged memory in use for in-memory data.
RowsReadPerSec	Processing: Rows read per sec	CountPerSecond	Average	Rate of rows read from all relational databases.
RowsConvertedPerSec	Processing: Rows converted per sec	CountPerSecond	Average	Rate of rows converted during processing.
RowsWrittenPerSec	Processing: Rows written per sec	CountPerSecond	Average	Rate of rows written during processing.
CommandPoolBusyThreads	Threads: Command pool busy threads	Count	Average	Number of busy threads in the command thread pool.
CommandPoolIdleThreads	Threads: Command pool idle threads	Count	Average	Number of idle threads in the command thread pool.
LongParsingBusyThreads	Threads: Long parsing busy threads	Count	Average	Number of busy threads in the long parsing thread pool.
LongParsingIdleThreads	Threads: Long parsing idle threads	Count	Average	Number of idle threads in the long parsing thread pool.
LongParsingJobQueueLength	Threads: Long parsing job queue length	Count	Average	Number of jobs in the queue of the long parsing thread pool.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
ProcessingPoolBusyI/OJobThreads	Threads: Processing pool busy I/O job threads	Count	Average	Number of threads running I/O jobs in the processing thread pool.
ProcessingPoolBusyNonIOTreads	Threads: Processing pool busy non-I/O threads	Count	Average	Number of threads running non-I/O jobs in the processing thread pool.
ProcessingPoolIOJobQueueLength	Threads: Processing pool I/O job queue length	Count	Average	Number of I/O jobs in the queue of the processing thread pool.
ProcessingPoolIdleI/OJobThreads	Threads: Processing pool idle I/O job threads	Count	Average	Number of idle threads for I/O jobs in the processing thread pool.
ProcessingPoolIdleNonIOTreads	Threads: Processing pool idle non-I/O threads	Count	Average	Number of idle threads in the processing thread pool dedicated to non-I/O jobs.
QueryPoolIdleThreads	Threads: Query pool idle threads	Count	Average	Number of idle threads for I/O jobs in the processing thread pool.
QueryPoolJobQueueLength	Threads: Query pool job queue length	Count	Average	Number of jobs in the queue of the query thread pool.
ShortParsingBusyThreads	Threads: Short parsing busy threads	Count	Average	Number of busy threads in the short parsing thread pool.
ShortParsingIdleThreads	Threads: Short parsing idle threads	Count	Average	Number of idle threads in the short parsing thread pool.
ShortParsingJobQueueLength	Threads: Short parsing job queue length	Count	Average	Number of jobs in the queue of the short parsing thread pool.
memory_thrashing_metric	Memory Thrashing	Percent	Average	Average memory thrashing.

Microsoft.ApiManagement/service

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
--------	---------------------	------	------------------	-------------

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
TotalRequests	Total Gateway Requests	Count	Total	Number of gateway requests
SuccessfulRequests	Successful Gateway Requests	Count	Total	Number of successful gateway requests
UnauthorizedRequests	Unauthorized Gateway Requests	Count	Total	Number of unauthorized gateway requests
FailedRequests	Failed Gateway Requests	Count	Total	Number of failures in gateway requests
OtherRequests	Other Gateway Requests	Count	Total	Number of other gateway requests

Microsoft.Batch/batchAccounts

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
CoreCount	Dedicated Core Count	Count	Total	Total number of dedicated cores in the batch account
TotalNodeCount	Dedicated Node Count	Count	Total	Total number of dedicated nodes in the batch account
LowPriorityCoreCount	LowPriority Core Count	Count	Total	Total number of low-priority cores in the batch account
TotalLowPriorityNodeCount	Low-Priority Node Count	Count	Total	Total number of low-priority nodes in the batch account
CreatingNodeCount	Creating Node Count	Count	Total	Number of nodes being created
StartingNodeCount	Starting Node Count	Count	Total	Number of nodes starting
WaitingForStartTaskNodeCount	Waiting For Start Task Node Count	Count	Total	Number of nodes waiting for the Start Task to complete
StartTaskFailedNodeCount	Start Task Failed Node Count	Count	Total	Number of nodes where the Start Task has failed
IdleNodeCount	Idle Node Count	Count	Total	Number of idle nodes

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
OfflineNodeCount	Offline Node Count	Count	Total	Number of offline nodes
RebootingNodeCount	Rebooting Node Count	Count	Total	Number of rebooting nodes
ReimagingNodeCount	Reimaging Node Count	Count	Total	Number of reimaging nodes
RunningNodeCount	Running Node Count	Count	Total	Number of running nodes
LeavingPoolNodeCount	Leaving Pool Node Count	Count	Total	Number of nodes leaving the Pool
UnusableNodeCount	Unusable Node Count	Count	Total	Number of unusable nodes
PreemptedNodeCount	Preempted Node Count	Count	Total	Number of preempted nodes
TaskStartEvent	Task Start Events	Count	Total	Total number of tasks that have started
TaskCompleteEvent	Task Complete Events	Count	Total	Total number of tasks that have completed
TaskFailEvent	Task Fail Events	Count	Total	Total number of tasks that have completed in a failed state
PoolCreateEvent	Pool Create Events	Count	Total	Total number of pools that have been created
PoolResizeStartEvent	Pool Resize Start Events	Count	Total	Total number of pool resizes that have started
PoolResizeCompleteEvent	Pool Resize Complete Events	Count	Total	Total number of pool resizes that have completed
PoolDeleteStartEvent	Pool Delete Start Events	Count	Total	Total number of pool deletes that have started
PoolDeleteCompleteEvent	Pool Delete Complete Events	Count	Total	Total number of pool deletes that have completed

Microsoft.Cache/redis

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
connectedclients	Connected Clients	Count	Maximum	
totalcommandsprocessed	Total Operations	Count	Total	
cachehits	Cache Hits	Count	Total	
cachemisses	Cache Misses	Count	Total	
getcommands	Gets	Count	Total	
setcommands	Sets	Count	Total	
evictedkeys	Evicted Keys	Count	Total	
totalkeys	Total Keys	Count	Maximum	
expiredkeys	Expired Keys	Count	Total	
usedmemory	Used Memory	Bytes	Maximum	
usedmemoryRss	Used Memory RSS	Bytes	Maximum	
serverLoad	Server Load	Percent	Maximum	
cacheWrite	Cache Write	BytesPerSecond	Maximum	
cacheRead	Cache Read	BytesPerSecond	Maximum	
percentProcessorTime	CPU	Percent	Maximum	
connectedclients0	Connected Clients (Shard 0)	Count	Maximum	
totalcommandsprocessed0	Total Operations (Shard 0)	Count	Total	
cachehits0	Cache Hits (Shard 0)	Count	Total	
cachemisses0	Cache Misses (Shard 0)	Count	Total	
getcommands0	Gets (Shard 0)	Count	Total	
setcommands0	Sets (Shard 0)	Count	Total	
evictedkeys0	Evicted Keys (Shard 0)	Count	Total	
totalkeys0	Total Keys (Shard 0)	Count	Maximum	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
expiredkeys0	Expired Keys (Shard 0)	Count	Total	
usedmemory0	Used Memory (Shard 0)	Bytes	Maximum	
usedmemoryRss0	Used Memory RSS (Shard 0)	Bytes	Maximum	
serverLoad0	Server Load (Shard 0)	Percent	Maximum	
cacheWrite0	Cache Write (Shard 0)	BytesPerSecond	Maximum	
cacheRead0	Cache Read (Shard 0)	BytesPerSecond	Maximum	
percentProcessorTime0	CPU (Shard 0)	Percent	Maximum	
connectedclients1	Connected Clients (Shard 1)	Count	Maximum	
totalcommandsprocessed1	Total Operations (Shard 1)	Count	Total	
cachehits1	Cache Hits (Shard 1)	Count	Total	
cachemisses1	Cache Misses (Shard 1)	Count	Total	
getcommands1	Gets (Shard 1)	Count	Total	
setcommands1	Sets (Shard 1)	Count	Total	
evictedkeys1	Evicted Keys (Shard 1)	Count	Total	
totalkeys1	Total Keys (Shard 1)	Count	Maximum	
expiredkeys1	Expired Keys (Shard 1)	Count	Total	
usedmemory1	Used Memory (Shard 1)	Bytes	Maximum	
usedmemoryRss1	Used Memory RSS (Shard 1)	Bytes	Maximum	
serverLoad1	Server Load (Shard 1)	Percent	Maximum	
cacheWrite1	Cache Write (Shard 1)	BytesPerSecond	Maximum	
cacheRead1	Cache Read (Shard 1)	BytesPerSecond	Maximum	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
percentProcessorTime1	CPU (Shard 1)	Percent	Maximum	
connectedclients2	Connected Clients (Shard 2)	Count	Maximum	
totalcommandsprocessed2	Total Operations (Shard 2)	Count	Total	
cachehits2	Cache Hits (Shard 2)	Count	Total	
cachemisses2	Cache Misses (Shard 2)	Count	Total	
getcommands2	Gets (Shard 2)	Count	Total	
setcommands2	Sets (Shard 2)	Count	Total	
evictedkeys2	Evicted Keys (Shard 2)	Count	Total	
totalkeys2	Total Keys (Shard 2)	Count	Maximum	
expiredkeys2	Expired Keys (Shard 2)	Count	Total	
usedmemory2	Used Memory (Shard 2)	Bytes	Maximum	
usedmemoryRss2	Used Memory RSS (Shard 2)	Bytes	Maximum	
serverLoad2	Server Load (Shard 2)	Percent	Maximum	
cacheWrite2	Cache Write (Shard 2)	BytesPerSecond	Maximum	
cacheRead2	Cache Read (Shard 2)	BytesPerSecond	Maximum	
percentProcessorTime2	CPU (Shard 2)	Percent	Maximum	
connectedclients3	Connected Clients (Shard 3)	Count	Maximum	
totalcommandsprocessed3	Total Operations (Shard 3)	Count	Total	
cachehits3	Cache Hits (Shard 3)	Count	Total	
cachemisses3	Cache Misses (Shard 3)	Count	Total	
getcommands3	Gets (Shard 3)	Count	Total	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
setcommands3	Sets (Shard 3)	Count	Total	
evictedkeys3	Evicted Keys (Shard 3)	Count	Total	
totalkeys3	Total Keys (Shard 3)	Count	Maximum	
expiredkeys3	Expired Keys (Shard 3)	Count	Total	
usedmemory3	Used Memory (Shard 3)	Bytes	Maximum	
usedmemoryRss3	Used Memory RSS (Shard 3)	Bytes	Maximum	
serverLoad3	Server Load (Shard 3)	Percent	Maximum	
cacheWrite3	Cache Write (Shard 3)	BytesPerSecond	Maximum	
cacheRead3	Cache Read (Shard 3)	BytesPerSecond	Maximum	
percentProcessorTime3	CPU (Shard 3)	Percent	Maximum	
connectedclients4	Connected Clients (Shard 4)	Count	Maximum	
totalcommandsprocessed4	Total Operations (Shard 4)	Count	Total	
cachehits4	Cache Hits (Shard 4)	Count	Total	
cachemisses4	Cache Misses (Shard 4)	Count	Total	
getcommands4	Gets (Shard 4)	Count	Total	
setcommands4	Sets (Shard 4)	Count	Total	
evictedkeys4	Evicted Keys (Shard 4)	Count	Total	
totalkeys4	Total Keys (Shard 4)	Count	Maximum	
expiredkeys4	Expired Keys (Shard 4)	Count	Total	
usedmemory4	Used Memory (Shard 4)	Bytes	Maximum	
usedmemoryRss4	Used Memory RSS (Shard 4)	Bytes	Maximum	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
serverLoad4	Server Load (Shard 4)	Percent	Maximum	
cacheWrite4	Cache Write (Shard 4)	BytesPerSecond	Maximum	
cacheRead4	Cache Read (Shard 4)	BytesPerSecond	Maximum	
percentProcessorTime4	CPU (Shard 4)	Percent	Maximum	
connectedclients5	Connected Clients (Shard 5)	Count	Maximum	
totalcommandsprocessed5	Total Operations (Shard 5)	Count	Total	
cachehits5	Cache Hits (Shard 5)	Count	Total	
cachemisses5	Cache Misses (Shard 5)	Count	Total	
getcommands5	Gets (Shard 5)	Count	Total	
setcommands5	Sets (Shard 5)	Count	Total	
evictedkeys5	Evicted Keys (Shard 5)	Count	Total	
totalkeys5	Total Keys (Shard 5)	Count	Maximum	
expiredkeys5	Expired Keys (Shard 5)	Count	Total	
usedmemory5	Used Memory (Shard 5)	Bytes	Maximum	
usedmemoryRss5	Used Memory RSS (Shard 5)	Bytes	Maximum	
serverLoad5	Server Load (Shard 5)	Percent	Maximum	
cacheWrite5	Cache Write (Shard 5)	BytesPerSecond	Maximum	
cacheRead5	Cache Read (Shard 5)	BytesPerSecond	Maximum	
percentProcessorTime5	CPU (Shard 5)	Percent	Maximum	
connectedclients6	Connected Clients (Shard 6)	Count	Maximum	
totalcommandsprocessed6	Total Operations (Shard 6)	Count	Total	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
cachehits6	Cache Hits (Shard 6)	Count	Total	
cachemisses6	Cache Misses (Shard 6)	Count	Total	
getcommands6	Gets (Shard 6)	Count	Total	
setcommands6	Sets (Shard 6)	Count	Total	
evictedkeys6	Evicted Keys (Shard 6)	Count	Total	
totalkeys6	Total Keys (Shard 6)	Count	Maximum	
expiredkeys6	Expired Keys (Shard 6)	Count	Total	
usedmemory6	Used Memory (Shard 6)	Bytes	Maximum	
usedmemoryRss6	Used Memory RSS (Shard 6)	Bytes	Maximum	
serverLoad6	Server Load (Shard 6)	Percent	Maximum	
cacheWrite6	Cache Write (Shard 6)	BytesPerSecond	Maximum	
cacheRead6	Cache Read (Shard 6)	BytesPerSecond	Maximum	
percentProcessorTime6	CPU (Shard 6)	Percent	Maximum	
connectedclients7	Connected Clients (Shard 7)	Count	Maximum	
totalcommandsprocessed7	Total Operations (Shard 7)	Count	Total	
cachehits7	Cache Hits (Shard 7)	Count	Total	
cachemisses7	Cache Misses (Shard 7)	Count	Total	
getcommands7	Gets (Shard 7)	Count	Total	
setcommands7	Sets (Shard 7)	Count	Total	
evictedkeys7	Evicted Keys (Shard 7)	Count	Total	
totalkeys7	Total Keys (Shard 7)	Count	Maximum	
expiredkeys7	Expired Keys (Shard 7)	Count	Total	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
usedmemory7	Used Memory (Shard 7)	Bytes	Maximum	
usedmemoryRss7	Used Memory RSS (Shard 7)	Bytes	Maximum	
serverLoad7	Server Load (Shard 7)	Percent	Maximum	
cacheWrite7	Cache Write (Shard 7)	BytesPerSecond	Maximum	
cacheRead7	Cache Read (Shard 7)	BytesPerSecond	Maximum	
percentProcessorTime7	CPU (Shard 7)	Percent	Maximum	
connectedclients8	Connected Clients (Shard 8)	Count	Maximum	
totalcommandsprocessed8	Total Operations (Shard 8)	Count	Total	
cachehits8	Cache Hits (Shard 8)	Count	Total	
cachemisses8	Cache Misses (Shard 8)	Count	Total	
getcommands8	Gets (Shard 8)	Count	Total	
setcommands8	Sets (Shard 8)	Count	Total	
evictedkeys8	Evicted Keys (Shard 8)	Count	Total	
totalkeys8	Total Keys (Shard 8)	Count	Maximum	
expiredkeys8	Expired Keys (Shard 8)	Count	Total	
usedmemory8	Used Memory (Shard 8)	Bytes	Maximum	
usedmemoryRss8	Used Memory RSS (Shard 8)	Bytes	Maximum	
serverLoad8	Server Load (Shard 8)	Percent	Maximum	
cacheWrite8	Cache Write (Shard 8)	BytesPerSecond	Maximum	
cacheRead8	Cache Read (Shard 8)	BytesPerSecond	Maximum	
percentProcessorTime8	CPU (Shard 8)	Percent	Maximum	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
connectedclients9	Connected Clients (Shard 9)	Count	Maximum	
totalcommandsprocessed9	Total Operations (Shard 9)	Count	Total	
cachehits9	Cache Hits (Shard 9)	Count	Total	
cachemisses9	Cache Misses (Shard 9)	Count	Total	
getcommands9	Gets (Shard 9)	Count	Total	
setcommands9	Sets (Shard 9)	Count	Total	
evictedkeys9	Evicted Keys (Shard 9)	Count	Total	
totalkeys9	Total Keys (Shard 9)	Count	Maximum	
expiredkeys9	Expired Keys (Shard 9)	Count	Total	
usedmemory9	Used Memory (Shard 9)	Bytes	Maximum	
usedmemoryRss9	Used Memory RSS (Shard 9)	Bytes	Maximum	
serverLoad9	Server Load (Shard 9)	Percent	Maximum	
cacheWrite9	Cache Write (Shard 9)	BytesPerSecond	Maximum	
cacheRead9	Cache Read (Shard 9)	BytesPerSecond	Maximum	
percentProcessorTime9	CPU (Shard 9)	Percent	Maximum	

Microsoft.CognitiveServices/accounts

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
TotalCalls	Total Calls	Count	Total	Total number of calls.
SuccessfulCalls	Successful Calls	Count	Total	Number of successful calls.
TotalErrors	Total Errors	Count	Total	Total number of calls with error response (HTTP response code 4xx or 5xx).

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
BlockedCalls	Blocked Calls	Count	Total	Number of calls that exceeded rate or quota limit.
ServerErrors	Server Errors	Count	Total	Number of calls with service internal error (HTTP response code 5xx).
ClientErrors	Client Errors	Count	Total	Number of calls with client side error (HTTP response code 4xx).
DataIn	Data In	Bytes	Total	Size of incoming data in bytes.
DataOut	Data Out	Bytes	Total	Size of outgoing data in bytes.
Latency	Latency	MilliSeconds	Average	Latency in milliseconds.

Microsoft.Compute/virtualMachines

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
Percentage CPU	Percentage CPU	Percent	Average	The percentage of allocated compute units that are currently in use by the Virtual Machine(s)
Network In	Network In	Bytes	Total	The number of bytes received on all network interfaces by the Virtual Machine(s) (Incoming Traffic)
Network Out	Network Out	Bytes	Total	The number of bytes out on all network interfaces by the Virtual Machine(s) (Outgoing Traffic)
Disk Read Bytes	Disk Read Bytes	Bytes	Total	Total bytes read from disk during monitoring period
Disk Write Bytes	Disk Write Bytes	Bytes	Total	Total bytes written to disk during monitoring period
Disk Read Operations/Sec	Disk Read Operations/Sec	CountPerSecond	Average	Disk Read IOPS

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
Disk Write Operations/Sec	Disk Write Operations/Sec	CountPerSecond	Average	Disk Write IOPS

Microsoft.Compute/virtualMachineScaleSets

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
Percentage CPU	Percentage CPU	Percent	Average	The percentage of allocated compute units that are currently in use by the Virtual Machine(s)
Network In	Network In	Bytes	Total	The number of bytes received on all network interfaces by the Virtual Machine(s) (Incoming Traffic)
Network Out	Network Out	Bytes	Total	The number of bytes out on all network interfaces by the Virtual Machine(s) (Outgoing Traffic)
Disk Read Bytes	Disk Read Bytes	Bytes	Total	Total bytes read from disk during monitoring period
Disk Write Bytes	Disk Write Bytes	Bytes	Total	Total bytes written to disk during monitoring period
Disk Read Operations/Sec	Disk Read Operations/Sec	CountPerSecond	Average	Disk Read IOPS
Disk Write Operations/Sec	Disk Write Operations/Sec	CountPerSecond	Average	Disk Write IOPS

Microsoft.Compute/virtualMachineScaleSets/virtualMachines

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
Percentage CPU	Percentage CPU	Percent	Average	The percentage of allocated compute units that are currently in use by the Virtual Machine(s)
Network In	Network In	Bytes	Total	The number of bytes received on all network interfaces by the Virtual Machine(s) (Incoming Traffic)

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
Network Out	Network Out	Bytes	Total	The number of bytes out on all network interfaces by the Virtual Machine(s) (Outgoing Traffic)
Disk Read Bytes	Disk Read Bytes	Bytes	Total	Total bytes read from disk during monitoring period
Disk Write Bytes	Disk Write Bytes	Bytes	Total	Total bytes written to disk during monitoring period
Disk Read Operations/Sec	Disk Read Operations/Sec	CountPerSecond	Average	Disk Read IOPS
Disk Write Operations/Sec	Disk Write Operations/Sec	CountPerSecond	Average	Disk Write IOPS

Microsoft.CustomerInsights/hubs

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
DCIApiCalls	Customer Insights API Calls	Count	Total	
DCIMappingImportOperationSuccessfulLines	Mapping Import Operation Successful Lines	Count	Total	
DCIMappingImportOperationFailedLines	Mapping Import Operation Failed Lines	Count	Total	
DCIMappingImportOperationTotalLines	Mapping Import Operation Total Lines	Count	Total	
DCIMappingImportOperationRuntimeInSeconds	Mapping Import Operation Runtime In Seconds	Seconds	Total	
DCIOutboundProfileExportSucceeded	Outbound Profile Export Succeeded	Count	Total	
DCIOutboundProfileExportFailed	Outbound Profile Export Failed	Count	Total	
DCIOutboundProfileExportDuration	Outbound Profile Export Duration	Seconds	Total	
DCIOutboundKpiExportSucceeded	Outbound Kpi Export Succeeded	Count	Total	

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
DCIOutboundKpiExportFailed	Outbound Kpi Export Failed	Count	Total	
DCIOutboundKpiExportDuration	Outbound Kpi Export Duration	Seconds	Total	
DCIOutboundKpiExportStarted	Outbound Kpi Export Started	Seconds	Total	
DCIOutboundKpiRecordCount	Outbound Kpi Record Count	Seconds	Total	
DCIOutboundProfileExportCount	Outbound Profile Export Count	Seconds	Total	
DCIOutboundInitialProfileExportFailed	Outbound Initial Profile Export Failed	Seconds	Total	
DCIOutboundInitialProfileExportSucceeded	Outbound Initial Profile Export Succeeded	Seconds	Total	
DCIOutboundInitialKpiExportFailed	Outbound Initial Kpi Export Failed	Seconds	Total	
DCIOutboundInitialKpiExportSucceeded	Outbound Initial Kpi Export Succeeded	Seconds	Total	
DCIOutboundInitialProfileExportDurationInSeconds	Outbound Initial Profile Export Duration In Seconds	Seconds	Total	
AdlaJobForStandardKpiFailed	Adla Job For Standard Kpi Failed In Seconds	Seconds	Total	
AdlaJobForStandardKpiTimeOut	Adla Job For Standard Kpi TimeOut In Seconds	Seconds	Total	
AdlaJobForStandardKpiCompleted	Adla Job For Standard Kpi Completed In Seconds	Seconds	Total	
ImportASAValuesFailed	Import ASA Values Failed Count	Count	Total	
ImportASAValuesSucceeded	Import ASA Values Succeeded Count	Count	Total	

Microsoft.DataLakeAnalytics/accounts

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
JobEndedSuccess	Successful Jobs	Count	Total	Count of successful jobs.
JobEndedFailure	Failed Jobs	Count	Total	Count of failed jobs.
JobEndedCancelled	Cancelled Jobs	Count	Total	Count of cancelled jobs.
JobAUEndedSuccess	Successful AU Time	Seconds	Total	Total AU time for successful jobs.
JobAUEndedFailure	Failed AU Time	Seconds	Total	Total AU time for failed jobs.
JobAUEndedCancelled	Cancelled AU Time	Seconds	Total	Total AU time for cancelled jobs.

Microsoft.DBforMySQL/servers

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
cpu_percent	CPU percent	Percent	Average	CPU percent
compute_limit	Compute Unit limit	Count	Average	Compute Unit limit
compute_consumption_percent	Compute Unit percentage	Percent	Average	Compute Unit percentage
memory_percent	Memory percent	Percent	Average	Memory percent
io_consumption_percent	IO percent	Percent	Average	IO percent
storage_percent	Storage percentage	Percent	Average	Storage percentage
storage_used	Storage used	Bytes	Average	Storage used
storage_limit	Storage limit	Bytes	Average	Storage limit
active_connections	Total active connections	Count	Average	Total active connections
connections_failed	Total failed connections	Count	Average	Total failed connections

Microsoft.DBforPostgreSQL/servers

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
cpu_percent	CPU percent	Percent	Average	CPU percent

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
compute_limit	Compute Unit limit	Count	Average	Compute Unit limit
compute_consumption_percent	Compute Unit percentage	Percent	Average	Compute Unit percentage
memory_percent	Memory percent	Percent	Average	Memory percent
io_consumption_percent	IO percent	Percent	Average	IO percent
storage_percent	Storage percentage	Percent	Average	Storage percentage
storage_used	Storage used	Bytes	Average	Storage used
storage_limit	Storage limit	Bytes	Average	Storage limit
active_connections	Total active connections	Count	Average	Total active connections
connections_failed	Total failed connections	Count	Average	Total failed connections

Microsoft.Devices/IotHubs

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
d2c.telemetry.ingress.allProtocol	Telemetry message send attempts	Count	Total	Number of device-to-cloud telemetry messages attempted to be sent to your IoT hub
d2c.telemetry.ingress.success	Telemetry messages sent	Count	Total	Number of device-to-cloud telemetry messages sent successfully to your IoT hub
c2d.commands.egress.complete.success	Commands completed	Count	Total	Number of cloud-to-device commands completed successfully by the device
c2d.commands.egress.abandon.success	Commands abandoned	Count	Total	Number of cloud-to-device commands abandoned by the device
c2d.commands.egress.reject.success	Commands rejected	Count	Total	Number of cloud-to-device commands rejected by the device

Metric	Metric display name	Unit	Aggregation type	Description
devices.totalDevices	Total devices	Count	Total	Number of devices registered to your IoT hub
devices.connectedDevices.allProtocol	Connected devices	Count	Total	Number of devices connected to your IoT hub
d2c.telemetry.egress.success	Telemetry messages delivered	Count	Total	Number of times messages were successfully written to endpoints (total)
d2c.telemetry.egress.dropped	Dropped messages	Count	Total	Number of messages dropped because the delivery endpoint was dead
d2c.telemetry.egress.orphaned	Orphaned messages	Count	Total	The count of messages not matching any routes including the fallback route
d2c.telemetry.egress.invalid	Invalid messages	Count	Total	The count of messages not delivered due to incompatibility with the endpoint
d2c.telemetry.egress.fallback	Messages matching fallback condition	Count	Total	Number of messages written to the fallback endpoint
d2c.endpoints.egress.eventHubs	Messages delivered to Event Hub endpoints	Count	Total	Number of times messages were successfully written to Event Hub endpoints
d2c.endpoints.latency.eventHubs	Message latency for Event Hub endpoints	Milliseconds	Average	The average latency between message ingress to the IoT hub and message ingress into an Event Hub endpoint, in milliseconds
d2c.endpoints.egress.serviceBusQueues	Messages delivered to Service Bus Queue endpoints	Count	Total	Number of times messages were successfully written to Service Bus Queue endpoints

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
d2c.endpoints.latency.serviceBusQueues	Message latency for Service Bus Queue endpoints	Milliseconds	Average	The average latency between message ingress to the IoT hub and message ingress into a Service Bus Queue endpoint, in milliseconds
d2c.endpoints.egress.serviceBusTopics	Messages delivered to Service Bus Topic endpoints	Count	Total	Number of times messages were successfully written to Service Bus Topic endpoints
d2c.endpoints.latency.serviceBusTopics	Message latency for Service Bus Topic endpoints	Milliseconds	Average	The average latency between message ingress to the IoT hub and message ingress into a Service Bus Topic endpoint, in milliseconds
d2c.endpoints.egress.builtIn.events	Messages delivered to the built-in endpoint (messages/events)	Count	Total	Number of times messages were successfully written to the built-in endpoint (messages/events)
d2c.endpoints.latency.builtIn.events	Message latency for the built-in endpoint (messages/events)	Milliseconds	Average	The average latency between message ingress to the IoT hub and message ingress into the built-in endpoint (messages/events), in milliseconds
d2c.twin.read.success	Successful twin reads from devices	Count	Total	The count of all successful device-initiated twin reads.
d2c.twin.read.failure	Failed twin reads from devices	Count	Total	The count of all failed device-initiated twin reads.
d2c.twin.read.size	Response size of twin reads from devices	Bytes	Average	The average, min, and max of all successful device-initiated twin reads.
d2c.twin.update.success	Successful twin updates from devices	Count	Total	The count of all successful device-initiated twin updates.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
d2c.twin.update.failure	Failed twin updates from devices	Count	Total	The count of all failed device-initiated twin updates.
d2c.twin.update.size	Size of twin updates from devices	Bytes	Average	The average, min, and max size of all successful device-initiated twin updates.
c2d.methods.success	Successful direct method invocations	Count	Total	The count of all successful direct method calls.
c2d.methods.failure	Failed direct method invocations	Count	Total	The count of all failed direct method calls.
c2d.methods.requestSize	Request size of direct method invocations	Bytes	Average	The average, min, and max of all successful direct method requests.
c2d.methods.responseSize	Response size of direct method invocations	Bytes	Average	The average, min, and max of all successful direct method responses.
c2d.twin.read.success	Successful twin reads from back end	Count	Total	The count of all successful back-end-initiated twin reads.
c2d.twin.read.failure	Failed twin reads from back end	Count	Total	The count of all failed back-end-initiated twin reads.
c2d.twin.read.size	Response size of twin reads from back end	Bytes	Average	The average, min, and max of all successful back-end-initiated twin reads.
c2d.twin.update.success	Successful twin updates from back end	Count	Total	The count of all successful back-end-initiated twin updates.
c2d.twin.update.failure	Failed twin updates from back end	Count	Total	The count of all failed back-end-initiated twin updates.
c2d.twin.update.size	Size of twin updates from back end	Bytes	Average	The average, min, and max size of all successful back-end-initiated twin updates.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
twinQueries.success	Successful twin queries	Count	Total	The count of all successful twin queries.
twinQueries.failure	Failed twin queries	Count	Total	The count of all failed twin queries.
twinQueries.resultSize	Twin queries result size	Bytes	Average	The average, min, and max of the result size of all successful twin queries.
jobs.createTwinUpdateJob.success	Successful creations of twin update jobs	Count	Total	The count of all successful creation of twin update jobs.
jobs.createTwinUpdateJob.failure	Failed creations of twin update jobs	Count	Total	The count of all failed creation of twin update jobs.
jobs.createDirectMethodJob.success	Successful creations of method invocation jobs	Count	Total	The count of all successful creation of direct method invocation jobs.
jobs.createDirectMethodJob.failure	Failed creations of method invocation jobs	Count	Total	The count of all failed creation of direct method invocation jobs.
jobs.listJobs.success	Successful calls to list jobs	Count	Total	The count of all successful calls to list jobs.
jobs.listJobs.failure	Failed calls to list jobs	Count	Total	The count of all failed calls to list jobs.
jobs.cancelJob.success	Successful job cancellations	Count	Total	The count of all successful calls to cancel a job.
jobs.cancelJob.failure	Failed job cancellations	Count	Total	The count of all failed calls to cancel a job.
jobs.queryJobs.success	Successful job queries	Count	Total	The count of all successful calls to query jobs.
jobs.queryJobs.failure	Failed job queries	Count	Total	The count of all failed calls to query jobs.
jobs.completed	Completed jobs	Count	Total	The count of all completed jobs.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
jobs.failed	Failed jobs	Count	Total	The count of all failed jobs.
d2c.telemetry.ingress.sendThrottle	Number of throttling errors	Count	Total	Number of throttling errors due to device throughput throttles
dailyMessageQuotaUsed	Total number of messages used	Count	Average	Number of total messages used today

Microsoft.EventHub/namespaces

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
INREQS	Incoming Send Requests	Count	Total	Total incoming send requests for a notification hub
SUCCREQ	Successful Requests	Count	Total	Total successful requests for a namespace
FAILREQ	Failed Requests	Count	Total	Total failed requests for a namespace
SVRBSY	Server Busy Errors	Count	Total	Total server busy errors for a namespace
INTERR	Internal Server Errors	Count	Total	Total internal server errors for a namespace
MISCERR	Other Errors	Count	Total	Total failed requests for a namespace
INMSGSS	Incoming Messages	Count	Total	Total incoming messages for a namespace
OUTMSGSS	Outgoing Messages	Count	Total	Total outgoing messages for a namespace
EHINMBS	Incoming bytes	Bytes	Total	Event Hub incoming message throughput for a namespace
EHOUTMBS	Outgoing bytes	Bytes	Total	Total outgoing messages for a namespace

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
EHABL	Archive backlog messages	Count	Total	Event Hub archive messages in backlog for a namespace
EHAMSGS	Archive messages	Count	Total	Event Hub archived messages in a namespace
EHAMBS	Archive message throughput	Bytes	Total	Event Hub archived message throughput in a namespace

Microsoft.Logic/workflows

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
RunsStarted	Runs Started	Count	Total	Number of workflow runs started.
RunsCompleted	Runs Completed	Count	Total	Number of workflow runs completed.
RunsSucceeded	Runs Succeeded	Count	Total	Number of workflow runs succeeded.
RunsFailed	Runs Failed	Count	Total	Number of workflow runs failed.
RunsCancelled	Runs Cancelled	Count	Total	Number of workflow runs cancelled.
RunLatency	Run Latency	Seconds	Average	Latency of completed workflow runs.
RunSuccessLatency	Run Success Latency	Seconds	Average	Latency of succeeded workflow runs.
RunThrottledEvents	Run Throttled Events	Count	Total	Number of workflow action or trigger throttled events.
RunFailurePercentage	Run Failure Percentage	Percent	Total	Percentage of workflow runs failed.
ActionsStarted	Actions Started	Count	Total	Number of workflow actions started.
ActionsCompleted	Actions Completed	Count	Total	Number of workflow actions completed.
ActionsSucceeded	Actions Succeeded	Count	Total	Number of workflow actions succeeded.

Metric	Metric Display Name	Unit	Aggregation Type	Description
ActionsFailed	Actions Failed	Count	Total	Number of workflow actions failed.
ActionsSkipped	Actions Skipped	Count	Total	Number of workflow actions skipped.
ActionLatency	Action Latency	Seconds	Average	Latency of completed workflow actions.
ActionSuccessLatency	Action Success Latency	Seconds	Average	Latency of succeeded workflow actions.
ActionThrottledEvents	Action Throttled Events	Count	Total	Number of workflow action throttled events..
TriggersStarted	Triggers Started	Count	Total	Number of workflow triggers started.
TriggersCompleted	Triggers Completed	Count	Total	Number of workflow triggers completed.
TriggersSucceeded	Triggers Succeeded	Count	Total	Number of workflow triggers succeeded.
TriggersFailed	Triggers Failed	Count	Total	Number of workflow triggers failed.
TriggersSkipped	Triggers Skipped	Count	Total	Number of workflow triggers skipped.
TriggersFired	Triggers Fired	Count	Total	Number of workflow triggers fired.
TriggerLatency	Trigger Latency	Seconds	Average	Latency of completed workflow triggers.
TriggerFireLatency	Trigger Fire Latency	Seconds	Average	Latency of fired workflow triggers.
TriggerSuccessLatency	Trigger Success Latency	Seconds	Average	Latency of succeeded workflow triggers.
TriggerThrottledEvents	Trigger Throttled Events	Count	Total	Number of workflow trigger throttled events.
BillableActionExecutions	Billable Action Executions	Count	Total	Number of workflow action executions getting billed.
BillableTriggerExecutions	Billable Trigger Executions	Count	Total	Number of workflow trigger executions getting billed.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
TotalBillableExecutions	Total Billable Executions	Count	Total	Number of workflow executions getting billed.

Microsoft.Network/applicationGateways

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
Throughput	Throughput	BytesPerSecond	Average	

Microsoft.Network/expressRouteCircuits

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
BytesIn	BytesIn	Count	Total	
BytesOut	BytesOut	Count	Total	

Microsoft.NotificationHubs/Namespace/NotificationHubs

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
registration.all	Registration Operation	Count	Total	The count of all successful registration operations (creations updates queries and deletions).
registration.create	Registration Create Operations	Count	Total	The count of all successful registration creations.
registration.update	Registration Update Operations	Count	Total	The count of all successful registration updates.
registration.get	Registration Read Operations	Count	Total	The count of all successful registration queries.
registration.delete	Registration Delete Operations	Count	Total	The count of all successful registration deletions.
incoming	Incoming Messages	Count	Total	The count of all successful send API calls.
incoming.scheduled	Scheduled Push Notifications Sent	Count	Total	Scheduled Push Notifications Cancelled

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
incoming.scheduled.cancel	Scheduled Push Notifications Cancelled	Count	Total	Scheduled Push Notifications Cancelled
scheduled.pending	Pending Scheduled Notifications	Count	Total	Pending Scheduled Notifications
installation.all	Installation Management Operations	Count	Total	Installation Management Operations
installation.get	Get Installation Operations	Count	Total	Get Installation Operations
installation.upsert	Create or Update Installation Operations	Count	Total	Create or Update Installation Operations
installation.patch	Patch Installation Operations	Count	Total	Patch Installation Operations
installation.delete	Delete Installation Operations	Count	Total	Delete Installation Operations
outgoing.allpns.success	Successful notifications	Count	Total	The count of all successful notifications.
outgoing.allpns.invalidpayload	Payload Errors	Count	Total	The count of pushes that failed because the PNS returned a bad payload error.
outgoing.allpns.pnserror	External Notification System Errors	Count	Total	The count of pushes that failed because there was a problem communicating with the PNS (excludes authentication problems).
outgoing.allpns.channelerror	Channel Errors	Count	Total	The count of pushes that failed because the channel was invalid not associated with the correct app throttled or expired.
outgoing.allpns.badorexpiredchannel	Bad or Expired Channel Errors	Count	Total	The count of pushes that failed because the channel/token/registrationId in the registration was expired or invalid.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
outgoing.wns.success	WNS Successful Notifications	Count	Total	The count of all successful notifications.
outgoing.wns.invalidcredentials	WNS Authorization Errors (Invalid Credentials)	Count	Total	The count of pushes that failed because the PNS did not accept the provided credentials or the credentials are blocked. (Windows Live does not recognize the credentials).
outgoing.wns.badchannel	WNS Bad Channel Error	Count	Total	The count of pushes that failed because the ChannelURI in the registration was not recognized (WNS status: 404 not found).
outgoing.wns.expiredchannel	WNS Expired Channel Error	Count	Total	The count of pushes that failed because the ChannelURI is expired (WNS status: 410 Gone).
outgoing.wns.throttled	WNS Throttled Notifications	Count	Total	The count of pushes that failed because WNS is throttling this app (WNS status: 406 Not Acceptable).
outgoing.wns.tokenproviderunreachable	WNS Authorization Errors (Unreachable)	Count	Total	Windows Live is not reachable.
outgoing.wns.invalidtoken	WNS Authorization Errors (Invalid Token)	Count	Total	The token provided to WNS is not valid (WNS status: 401 Unauthorized).

Metric	Metric Display Name	Unit	Aggregation Type	Description
outgoing.wns.wrongtoken	WNS Authorization Errors (Wrong Token)	Count	Total	The token provided to WNS is valid but for another application (WNS status: 403 Forbidden). This can happen if the ChannelURI in the registration is associated with another app. Check that the client app is associated with the same app whose credentials are in the notification hub.
outgoing.wns.invalidnotificationformat	WNS Invalid Notification Format	Count	Total	The format of the notification is invalid (WNS status: 400). Note that WNS does not reject all invalid payloads.
outgoing.wns.invalidnotificationsize	WNS Invalid Notification Size Error	Count	Total	The notification payload is too large (WNS status: 413).
outgoing.wns.channelthrottled	WNS Channel Throttled	Count	Total	The notification was dropped because the ChannelURI in the registration is throttled (WNS response header: X-WNS-NotificationStatus:channelThrottled).
outgoing.wns.channeldisconnected	WNS Channel Disconnected	Count	Total	The notification was dropped because the ChannelURI in the registration is throttled (WNS response header: X-WNS-DeviceConnectionStatus: disconnected).
outgoing.wns.dropped	WNS Dropped Notifications	Count	Total	The notification was dropped because the ChannelURI in the registration is throttled (X-WNS-NotificationStatus: dropped but not X-WNS-DeviceConnectionStatus: disconnected).

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
outgoing.wns.pnserro r	WNS Errors	Count	Total	Notification not delivered because of errors communicating with WNS.
outgoing.wns.authen ticationerror	WNS Authentication Errors	Count	Total	Notification not delivered because of errors communicating with Windows Live invalid credentials or wrong token.
outgoing.apns.succes s	APNS Successful Notifications	Count	Total	The count of all successful notifications.
outgoing.apns.invalid credentials	APNS Authorization Errors	Count	Total	The count of pushes that failed because the PNS did not accept the provided credentials or the credentials are blocked.
outgoing.apns.badch annel	APNS Bad Channel Error	Count	Total	The count of pushes that failed because the token is invalid (APNS status code: 8).
outgoing.apns.expire dchannel	APNS Expired Channel Error	Count	Total	The count of token that were invalidated by the APNS feedback channel.
outgoing.apns.invalid notificationsize	APNS Invalid Notification Size Error	Count	Total	The count of pushes that failed because the payload was too large (APNS status code: 7).
outgoing.apns.pnserr or	APNS Errors	Count	Total	The count of pushes that failed because of errors communicating with APNS.
outgoing.gcm.success	GCM Successful Notifications	Count	Total	The count of all successful notifications.
outgoing.gcm.invalidc redentials	GCM Authorization Errors (Invalid Credentials)	Count	Total	The count of pushes that failed because the PNS did not accept the provided credentials or the credentials are blocked.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
outgoing.gcm.badchannel	GCM Bad Channel Error	Count	Total	The count of pushes that failed because the registrationId in the registration was not recognized (GCM result: Invalid Registration).
outgoing.gcm.expiredchannel	GCM Expired Channel Error	Count	Total	The count of pushes that failed because the registrationId in the registration was expired (GCM result: NotRegistered).
outgoing.gcm.throttled	GCM Throttled Notifications	Count	Total	The count of pushes that failed because GCM throttled this app (GCM status code: 501-599 or result:Unavailable).
outgoing.gcm.invalidnotificationformat	GCM Invalid Notification Format	Count	Total	The count of pushes that failed because the payload was not formatted correctly (GCM result: InvalidDataKey or InvalidTtl).
outgoing.gcm.invalidnotificationsize	GCM Invalid Notification Size Error	Count	Total	The count of pushes that failed because the payload was too large (GCM result: MessageTooBig).
outgoing.gcm.wrongchannel	GCM Wrong Channel Error	Count	Total	The count of pushes that failed because the registrationId in the registration is not associated to the current app (GCM result: InvalidPackageName).
outgoing.gcm.pnserror	GCM Errors	Count	Total	The count of pushes that failed because of errors communicating with GCM.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
outgoing.gcm.authenticationerror	GCM Authentication Errors	Count	Total	The count of pushes that failed because the PNS did not accept the provided credentials the credentials are blocked or the SenderId is not correctly configured in the app (GCM result: MismatchedSenderId).
outgoing.mpns.success	MPNS Successful Notifications	Count	Total	The count of all successful notifications.
outgoing.mpns.invalidcredentials	MPNS Invalid Credentials	Count	Total	The count of pushes that failed because the PNS did not accept the provided credentials or the credentials are blocked.
outgoing.mpns.badchannel	MPNS Bad Channel Error	Count	Total	The count of pushes that failed because the ChannelURI in the registration was not recognized (MPNS status: 404 not found).
outgoing.mpns.throttled	MPNS Throttled Notifications	Count	Total	The count of pushes that failed because MPNS is throttling this app (WNS MPNS: 406 Not Acceptable).
outgoing.mpns.invalidnotificationformat	MPNS Invalid Notification Format	Count	Total	The count of pushes that failed because the payload of the notification was too large.
outgoing.mpns.channeldisconnected	MPNS Channel Disconnected	Count	Total	The count of pushes that failed because the ChannelURI in the registration was disconnected (MPNS status: 412 not found).

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
outgoing.mpns.dropped	MPNS Dropped Notifications	Count	Total	The count of pushes that were dropped by MPNS (MPNS response header: X-NotificationStatus: QueueFull or Suppressed).
outgoing.mpns.pnserror	MPNS Errors	Count	Total	The count of pushes that failed because of errors communicating with MPNS.
outgoing.mpns.authenticationerror	MPNS Authentication Errors	Count	Total	The count of pushes that failed because the PNS did not accept the provided credentials or the credentials are blocked.
notificationhub.pushes	All Outgoing Notifications	Count	Total	All outgoing notifications of the notification hub
incoming.all.requests	All Incoming Requests	Count	Total	Total incoming requests for a notification hub
incoming.all.failedrequests	All Incoming Failed Requests	Count	Total	Total incoming failed requests for a notification hub

Microsoft.PowerBIDedicated/capacities

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
qpu_metric	QPU	Count	Average	QPU. Range 0-100 for S1, 0-200 for S2 and 0-400 for S4
memory_metric	Memory	Bytes	Average	Memory. Range 0-25 GB for S1, 0-50 GB for S2 and 0-100 GB for S4
TotalConnectionRequests	Total Connection Requests	Count	Average	Total connection requests. These are arrivals.
SuccessfullConnectionsPerSec	Successful Connections Per Sec	CountPerSecond	Average	Rate of successful connection completions.

Metric	Metric Display Name	Unit	Aggregation Type	Description
TotalConnectionFailures	Total Connection Failures	Count	Average	Total failed connection attempts.
CurrentUserSessions	Current User Sessions	Count	Average	Current number of user sessions established.
QueryPoolBusyThreads	Query Pool Busy Threads	Count	Average	Number of busy threads in the query thread pool.
CommandPoolJobQueueLength	Command Pool Job Queue Length	Count	Average	Number of jobs in the queue of the command thread pool.
ProcessingPoolJobQueueLength	Processing Pool Job Queue Length	Count	Average	Number of non-I/O jobs in the queue of the processing thread pool.
CurrentConnections	Connection: Current connections	Count	Average	Current number of client connections established.
CleanerCurrentPrice	Memory: Cleaner Current Price	Count	Average	Current price of memory, \$/byte/time, normalized to 1000.
CleanerMemoryShrinkable	Memory: Cleaner Memory shrinkable	Bytes	Average	Amount of memory, in bytes, subject to purging by the background cleaner.
CleanerMemoryNonshrinkable	Memory: Cleaner Memory nonshrinkable	Bytes	Average	Amount of memory, in bytes, not subject to purging by the background cleaner.
MemoryUsage	Memory: Memory Usage	Bytes	Average	Memory usage of the server process as used in calculating cleaner memory price. Equal to counter Process\PrivateBytes plus the size of memory-mapped data, ignoring any memory which was mapped or allocated by the xVelocity in-memory analytics engine (VertiPaq) in excess of the xVelocity engine Memory Limit.

Metric	Metric Display Name	Unit	Aggregation Type	Description
MemoryLimitHard	Memory: Memory Limit Hard	Bytes	Average	Hard memory limit, from configuration file.
MemoryLimitHigh	Memory: Memory Limit High	Bytes	Average	High memory limit, from configuration file.
MemoryLimitLow	Memory: Memory Limit Low	Bytes	Average	Low memory limit, from configuration file.
MemoryLimitVertiPaq	Memory: Memory Limit VertiPaq	Bytes	Average	In-memory limit, from configuration file.
Quota	Memory: Quota	Bytes	Average	Current memory quota, in bytes. Memory quota is also known as a memory grant or memory reservation.
QuotaBlocked	Memory: Quota Blocked	Count	Average	Current number of quota requests that are blocked until other memory quotas are freed.
VertiPaqNonpaged	Memory: VertiPaq Nonpaged	Bytes	Average	Bytes of memory locked in the working set for use by the in-memory engine.
VertiPaqPaged	Memory: VertiPaq Paged	Bytes	Average	Bytes of paged memory in use for in-memory data.
RowsReadPerSec	Processing: Rows read per sec	CountPerSecond	Average	Rate of rows read from all relational databases.
RowsConvertedPerSec	Processing: Rows converted per sec	CountPerSecond	Average	Rate of rows converted during processing.
RowsWrittenPerSec	Processing: Rows written per sec	CountPerSecond	Average	Rate of rows written during processing.
CommandPoolBusyThreads	Threads: Command pool busy threads	Count	Average	Number of busy threads in the command thread pool.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
CommandPoolIdleThreads	Threads: Command pool idle threads	Count	Average	Number of idle threads in the command thread pool.
LongParsingBusyThreads	Threads: Long parsing busy threads	Count	Average	Number of busy threads in the long parsing thread pool.
LongParsingIdleThreads	Threads: Long parsing idle threads	Count	Average	Number of idle threads in the long parsing thread pool.
LongParsingJobQueueLength	Threads: Long parsing job queue length	Count	Average	Number of jobs in the queue of the long parsing thread pool.
ProcessingPoolBusyIOWJobThreads	Threads: Processing pool busy I/O job threads	Count	Average	Number of threads running I/O jobs in the processing thread pool.
ProcessingPoolBusyNonIOTThreads	Threads: Processing pool busy non-I/O threads	Count	Average	Number of threads running non-I/O jobs in the processing thread pool.
ProcessingPoolIOJobQueueLength	Threads: Processing pool I/O job queue length	Count	Average	Number of I/O jobs in the queue of the processing thread pool.
ProcessingPoolIdleIOWJobThreads	Threads: Processing pool idle I/O job threads	Count	Average	Number of idle threads for I/O jobs in the processing thread pool.
ProcessingPoolIdleNonIOTThreads	Threads: Processing pool idle non-I/O threads	Count	Average	Number of idle threads in the processing thread pool dedicated to non-I/O jobs.
QueryPoolIdleThreads	Threads: Query pool idle threads	Count	Average	Number of idle threads for I/O jobs in the processing thread pool.
QueryPoolJobQueueLength	Threads: Query pool job queue length	Count	Average	Number of jobs in the queue of the query thread pool.
ShortParsingBusyThreads	Threads: Short parsing busy threads	Count	Average	Number of busy threads in the short parsing thread pool.

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
ShortParsingIdleThreads	Threads: Short parsing idle threads	Count	Average	Number of idle threads in the short parsing thread pool.
ShortParsingJobQueueLength	Threads: Short parsing job queue length	Count	Average	Number of jobs in the queue of the short parsing thread pool.
memory_thrashing_metric	Memory Thrashing	Percent	Average	Average memory thrashing.

Microsoft.Search/searchServices

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
SearchLatency	Search Latency	Seconds	Average	Average search latency for the search service
SearchQueriesPerSecond	Search queries per second	CountPerSecond	Average	Search queries per second for the search service
ThrottledSearchQueriesPercentage	Throttled search queries percentage	Percent	Average	Percentage of search queries that were throttled for the search service

Microsoft.ServiceBus/namespaces

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
CPUXNS	CPU usage per namespace	Percent	Maximum	Service bus premium namespace CPU usage metric
WSXNS	Memory size usage per namespace	Percent	Maximum	Service bus premium namespace memory usage metric

Microsoft.Sql/servers/databases

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
cpu_percent	CPU percentage	Percent	Average	CPU percentage
physical_data_read_percent	Data IO percentage	Percent	Average	Data IO percentage
log_write_percent	Log IO percentage	Percent	Average	Log IO percentage

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
dtu_consumption_percent	DTU percentage	Percent	Average	DTU percentage
storage	Total database size	Bytes	Maximum	Total database size
connection_successful	Successful Connections	Count	Total	Successful Connections
connection_failed	Failed Connections	Count	Total	Failed Connections
blocked_by_firewall	Blocked by Firewall	Count	Total	Blocked by Firewall
deadlock	Deadlocks	Count	Total	Deadlocks
storage_percent	Database size percentage	Percent	Maximum	Database size percentage
xtp_storage_percent	In-Memory OLTP storage percent	Percent	Average	In-Memory OLTP storage percent
workers_percent	Workers percentage	Percent	Average	Workers percentage
sessions_percent	Sessions percentage	Percent	Average	Sessions percentage
dtu_limit	DTU Limit	Count	Average	DTU Limit
dtu_used	DTU used	Count	Average	DTU used
dwu_limit	DWU limit	Count	Maximum	DWU limit
dwu_consumption_percent	DWU percentage	Percent	Maximum	DWU percentage
dwu_used	DWU used	Count	Maximum	DWU used

Microsoft.Sql/servers/elasticPools

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
cpu_percent	CPU percentage	Percent	Average	CPU percentage
database_cpu_percent	CPU percentage	Percent	Average	CPU percentage
physical_data_read_percent	Data IO percentage	Percent	Average	Data IO percentage
database_physical_data_read_percent	Data IO percentage	Percent	Average	Data IO percentage
log_write_percent	Log IO percentage	Percent	Average	Log IO percentage

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
database_log_write_percent	Log IO percentage	Percent	Average	Log IO percentage
dtu_consumption_percent	DTU percentage	Percent	Average	DTU percentage
database_dtu_consumption_percent	DTU percentage	Percent	Average	DTU percentage
storage_percent	Storage percentage	Percent	Average	Storage percentage
workers_percent	Workers percentage	Percent	Average	Workers percentage
database_workers_percent	Workers percentage	Percent	Average	Workers percentage
sessions_percent	Sessions percentage	Percent	Average	Sessions percentage
database_sessions_percent	Sessions percentage	Percent	Average	Sessions percentage
eDTU_limit	eDTU limit	Count	Average	eDTU limit
storage_limit	Storage limit	Bytes	Average	Storage limit
eDTU_used	eDTU used	Count	Average	eDTU used
storage_used	Storage used	Bytes	Average	Storage used
database_storage_used	Storage used	Bytes	Average	Storage used
xtp_storage_percent	In-Memory OLTP storage percent	Percent	Average	In-Memory OLTP storage percent

Microsoft.Sql/servers

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
dtu_consumption_percent	DTU percentage	Percent	Average	DTU percentage
database_dtu_consumption_percent	DTU percentage	Percent	Average	DTU percentage
storage_used	Storage used	Bytes	Average	Storage used
database_storage_used	Storage used	Bytes	Average	Storage used

Microsoft.StreamAnalytics/streamingjobs

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
ResourceUtilization	SU % Utilization	Percent	Maximum	SU % Utilization
InputEvents	Input Events	Count	Total	Input Events
InputEventBytes	Input Event Bytes	Bytes	Total	Input Event Bytes
LateInputEvents	Late Input Events	Count	Total	Late Input Events
OutputEvents	Output Events	Count	Total	Output Events
ConversionErrors	Data Conversion Errors	Count	Total	Data Conversion Errors
Errors	Runtime Errors	Count	Total	Runtime Errors
DroppedOrAdjustedEvents	Out of order Events	Count	Total	Out of order Events
AMLCalloutRequests	Function Requests	Count	Total	Function Requests
AMLCalloutFailedRequests	Failed Function Requests	Count	Total	Failed Function Requests
AMLCalloutInputEvents	Function Events	Count	Total	Function Events

Microsoft.Web/serverfarms

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
CpuPercentage	CPU Percentage	Percent	Average	CPU Percentage
MemoryPercentage	Memory Percentage	Percent	Average	Memory Percentage
DiskQueueLength	Disk Queue Length	Count	Total	Disk Queue Length
HttpQueueLength	Http Queue Length	Count	Total	Http Queue Length
BytesReceived	Data In	Bytes	Total	Data In
BytesSent	Data Out	Bytes	Total	Data Out

Microsoft.Web/sites (excluding functions)

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
CpuTime	CPU Time	Seconds	Total	CPU Time
Requests	Requests	Count	Total	Requests

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
BytesReceived	Data In	Bytes	Total	Data In
BytesSent	Data Out	Bytes	Total	Data Out
Http101	Http 101	Count	Total	Http 101
Http2xx	Http 2xx	Count	Total	Http 2xx
Http3xx	Http 3xx	Count	Total	Http 3xx
Http401	Http 401	Count	Total	Http 401
Http403	Http 403	Count	Total	Http 403
Http404	Http 404	Count	Total	Http 404
Http406	Http 406	Count	Total	Http 406
Http4xx	Http 4xx	Count	Total	Http 4xx
Http5xx	Http Server Errors	Count	Total	Http Server Errors
MemoryWorkingSet	Memory working set	Bytes	Average	Memory working set
AverageMemoryWorkingSet	Average memory working set	Bytes	Average	Average memory working set
AverageResponseTime	Average Response Time	Seconds	Average	Average Response Time

Microsoft.Web/sites (functions)

METRIC	METRIC DISPLAY NAME	UNIT	AGGREGATION TYPE	DESCRIPTION
BytesReceived	Data In	Bytes	Total	Data In
BytesSent	Data Out	Bytes	Total	Data Out
Http5xx	Http Server Errors	Count	Total	Http Server Errors
MemoryWorkingSet	Memory working set	Bytes	Average	Memory working set
AverageMemoryWorkingSet	Average memory working set	Bytes	Average	Average memory working set
FunctionExecutionUnits	Function Execution Units	Count	Average	Function Execution Units
FunctionExecutionCount	Function Execution Count	Count	Average	Function Execution Count

Microsoft.Web/sites/slots

Metric	Metric Display Name	Unit	Aggregation Type	Description
CpuTime	CPU Time	Seconds	Total	CPU Time
Requests	Requests	Count	Total	Requests
BytesReceived	Data In	Bytes	Total	Data In
BytesSent	Data Out	Bytes	Total	Data Out
Http101	Http 101	Count	Total	Http 101
Http2xx	Http 2xx	Count	Total	Http 2xx
Http3xx	Http 3xx	Count	Total	Http 3xx
Http401	Http 401	Count	Total	Http 401
Http403	Http 403	Count	Total	Http 403
Http404	Http 404	Count	Total	Http 404
Http406	Http 406	Count	Total	Http 406
Http4xx	Http 4xx	Count	Total	Http 4xx
Http5xx	Http Server Errors	Count	Total	Http Server Errors
MemoryWorkingSet	Memory working set	Bytes	Average	Memory working set
AverageMemoryWorkingSet	Average memory working set	Bytes	Average	Average memory working set
AverageResponseTime	Average Response Time	Seconds	Average	Average Response Time
FunctionExecutionUnits	Function Execution Units	Count	Average	Function Execution Units
FunctionExecutionCount	Function Execution Count	Count	Average	Function Execution Count

Next steps

- [Read about metrics in Azure Monitor](#)
- [Create alerts on metrics](#)
- [Export metrics to storage, Event Hub, or Log Analytics](#)

Azure Activity Log event schema

7/25/2017 • 12 min to read • [Edit Online](#)

The **Azure Activity Log** is a log that provides insight into any subscription-level events that have occurred in Azure. This article describes the event schema per category of data.

Administrative

This category contains the record of all create, update, delete, and action operations performed through Resource Manager. Examples of the types of events you would see in this category include "create virtual machine" and "delete network security group". Every action taken by a user or application using Resource Manager is modeled as an operation on a particular resource type. If the operation type is Write, Delete, or Action, the records of both the start and success or fail of that operation are recorded in the Administrative category. The Administrative category also includes any changes to role-based access control in a subscription.

Sample event

```
{  
  "authorization": {  
    "action": "microsoft.support/supporttickets/write",  
    "role": "Subscription Admin",  
    "scope":  
      "/subscriptions/s1/resourceGroups/MSSupportGroup/providers/microsoft.support/supporttickets/115012112305841"  
  },  
  "caller": "admin@contoso.com",  
  "channels": "Operation",  
  "claims": {  
    "aud": "https://management.core.windows.net/",  
    "iss": "https://sts.windows.net/72f988bf-86f1-41af-91ab-2d7cd011db47/",  
    "iat": "1421876371",  
    "nbf": "1421876371",  
    "exp": "1421880271",  
    "ver": "1.0",  
    "http://schemas.microsoft.com/identity/claims/tenantid": "1e8d8218-c5e7-4578-9acc-9abbd5d23315 ",  
    "http://schemas.microsoft.com/claims/authnmethodsreferences": "pwd",  
    "http://schemas.microsoft.com/identity/claims/objectidentifier": "2468adf0-8211-44e3-95xq-85137af64708",  
    "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn": "admin@contoso.com",  
    "puid": "20030000801A118C",  
    "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/nameidentifier":  
      "9vcckmEGF7zDKk1YzIY8k0t1_EAPaXoeHyPRn6f413zM",  
    "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/givenname": "John",  
    "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/surname": "Smith",  
    "name": "John Smith",  
    "groups": "cacfef77c-e058-4712-83qw-f9b08849fd60,7f71d11d-4c41-4b23-99d2-d32ce7aa621c,31522864-0578-4ea0-  
      9gdc-e66cc564d18c",  
    "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name": " admin@contoso.com",  
    "appid": "c44b4083-3bq0-49c1-b47d-974e53cbdf3c",  
    "appidacr": "2",  
    "http://schemas.microsoft.com/identity/claims/scoped": "user_impersonation",  
    "http://schemas.microsoft.com/claims/authnclassreference": "1"  
},  
  "correlationId": "1e121103-0ba6-4300-ac9d-952bb5d0c80f",  
  "description": "",  
  "eventDataId": "44ade6b4-3813-45e6-ae27-7420a95fa2f8",  
  "eventName": {  

```

```

"clientRequestId": "27005025-9105-418T-8601-29e557ac0249",
"clientIpAddress": "192.168.35.115",
"method": "PUT"
},
"id":
"/subscriptions/s1/resourceGroups/MSSupportGroup/providers/microsoft.support/supporttickets/115012112305841/events/44ade6b4-3813-45e6-ae27-7420a95fa2f8/ticks/635574752669792776",
"level": "Informational",
"resourceGroupName": "MSSupportGroup",
"resourceProviderName": {
  "value": "microsoft.support",
  "localizedValue": "microsoft.support"
},
"resourceUri":
"/subscriptions/s1/resourceGroups/MSSupportGroup/providers/microsoft.support/supporttickets/115012112305841",
"operationId": "1e121103-0ba6-4300-ac9d-952bb5d0c80f",
"operationName": {
  "value": "microsoft.support/supporttickets/write",
  "localizedValue": "microsoft.support/supporttickets/write"
},
"properties": {
  "statusCode": "Created"
},
"status": {
  "value": "Succeeded",
  "localizedValue": "Succeeded"
},
"subStatus": {
  "value": "Created",
  "localizedValue": "Created (HTTP Status Code: 201)"
},
"eventTimestamp": "2015-01-21T22:14:26.9792776Z",
"submissionTimestamp": "2015-01-21T22:14:39.9936304Z",
"subscriptionId": "s1"
}

```

Property descriptions

ELEMENT NAME	DESCRIPTION
authorization	Blob of RBAC properties of the event. Usually includes the "action", "role" and "scope" properties.
caller	Email address of the user who has performed the operation, UPN claim, or SPN claim based on availability.
channels	One of the following values: "Admin", "Operation"
claims	The JWT token used by Active Directory to authenticate the user or application to perform this operation in resource manager.
correlationId	Usually a GUID in the string format. Events that share a correlationId belong to the same uber action.
description	Static text description of an event.
eventDataId	Unique identifier of an event.
httpRequest	Blob describing the Http Request. Usually includes the "clientRequestId", "clientIpAddress" and "method" (HTTP method. For example, PUT).

ELEMENT NAME	DESCRIPTION
level	Level of the event. One of the following values: "Critical", "Error", "Warning", "Informational" and "Verbose"
resourceGroupName	Name of the resource group for the impacted resource.
resourceProviderName	Name of the resource provider for the impacted resource
resourceId	Resource id of the impacted resource.
operationId	A GUID shared among the events that correspond to a single operation.
operationName	Name of the operation.
properties	Set of <Key, Value> pairs (that is, a Dictionary) describing the details of the event.
status	String describing the status of the operation. Some common values are: Started, In Progress, Succeeded, Failed, Active, Resolved.
subStatus	Usually the HTTP status code of the corresponding REST call, but can also include other strings describing a substatus, such as these common values: OK (HTTP Status Code: 200), Created (HTTP Status Code: 201), Accepted (HTTP Status Code: 202), No Content (HTTP Status Code: 204), Bad Request (HTTP Status Code: 400), Not Found (HTTP Status Code: 404), Conflict (HTTP Status Code: 409), Internal Server Error (HTTP Status Code: 500), Service Unavailable (HTTP Status Code: 503), Gateway Timeout (HTTP Status Code: 504).
eventTimestamp	Timestamp when the event was generated by the Azure service processing the request corresponding the event.
submissionTimestamp	Timestamp when the event became available for querying.
subscriptionId	Azure Subscription Id.

Service health

This category contains the record of any service health incidents that have occurred in Azure. An example of the type of event you would see in this category is "SQL Azure in East US is experiencing downtime." Service health events come in five varieties: Action Required, Assisted Recovery, Incident, Maintenance, Information, or Security, and only appear if you have a resource in the subscription that would be impacted by the event.

Sample event

```
{
  "channels": "Admin",
  "correlationId": "c550176b-8f52-4380-bdc5-36c1b59d3a44",
  "description": "Active: Network Infrastructure - UK South",
  "eventDataId": "c5bc4514-6642-2be3-453e-c6a67841b073",
  "eventName": {
    "value": null
  },
  "category": {
    "value": "ServiceHealth",
    "localizedValue": "Service Health"
  },
  "eventTimestamp": "2017-07-20T23:30:14.8022297Z",
  "id": "/subscriptions/mySubscriptionID/events/c5bc4514-6642-2be3-453e-c6a67841b073/ticks/636361902148022297",
  "level": "Warning",
  "operationName": {
    "value": "Microsoft.ServiceHealth/incident/action",
    "localizedValue": "Microsoft.ServiceHealth/incident/action"
  },
  "resourceProviderName": {
    "value": null
  },
  "resourceType": {
    "value": null,
    "localizedValue": ""
  },
  "resourceId": "/subscriptions/mySubscriptionID",
  "status": {
    "value": "Active",
    "localizedValue": "Active"
  },
  "subStatus": {
    "value": null
  },
  "submissionTimestamp": "2017-07-20T23:30:34.7431946Z",
  "subscriptionId": "mySubscriptionID",
  "properties": {
    "title": "Network Infrastructure - UK South",
    "service": "Service Fabric",
    "region": "UK South",
    "communication": "Starting at approximately 21:41 UTC on 20 Jul 2017, a subset of customers in UK South may experience degraded performance, connectivity drops or timeouts when accessing their Azure resources hosted in this region. Engineers are investigating underlying Network Infrastructure issues in this region. Impacted services may include, but are not limited to App Services, Automation, Service Bus, Log Analytics, Key Vault, SQL Database, Service Fabric, Event Hubs, Stream Analytics, Azure Data Movement, API Management, and Azure Search. Multiple engineering teams are engaged in multiple workflows to mitigate the impact. The next update will be provided in 60 minutes, or as events warrant.",
    "incidentType": "Incident",
    "trackingId": "NA0F-BJG",
    "impactStartTime": "2017-07-20T21:41:00.000000Z",
    "impactedServices": "[{\\"ImpactedRegions\\": [{\\"RegionName\\": \"UK South\\"}], \\"ServiceName\\": \"Service Fabric\\\"}]",
    "defaultLanguageTitle": "Network Infrastructure - UK South",
    "defaultLanguageContent": "Starting at approximately 21:41 UTC on 20 Jul 2017, a subset of customers in UK South may experience degraded performance, connectivity drops or timeouts when accessing their Azure resources hosted in this region. Engineers are investigating underlying Network Infrastructure issues in this region. Impacted services may include, but are not limited to App Services, Automation, Service Bus, Log Analytics, Key Vault, SQL Database, Service Fabric, Event Hubs, Stream Analytics, Azure Data Movement, API Management, and Azure Search. Multiple engineering teams are engaged in multiple workflows to mitigate the impact. The next update will be provided in 60 minutes, or as events warrant.",
    "stage": "Active",
    "communicationId": "636361902146035247",
    "version": "0.1.1"
  }
}
```

Property descriptions

ELEMENT NAME	DESCRIPTION
channels	Is one of the following values: "Admin", "Operation"
correlationId	Is usually a GUID in the string format. Events with that belong to the same uber action usually share the same correlationId.
description	Description of the event.
eventDataId	The unique identifier of an event.
eventName	The title of the event.
level	Level of the event. One of the following values: "Critical", "Error", "Warning", "Informational" and "Verbose"
resourceProviderName	Name of the resource provider for the impacted resource. If not known, this will be null.
resourceType	The type of resource of the impacted resource. If not known, this will be null.
subStatus	Usually null for Service Health events.
eventTimestamp	Timestamp when the log event was generated and submitted to the Activity Log.
submissionTimestamp	Timestamp when the event became available in the Activity Log.
subscriptionId	The Azure subscription in which this event was logged.
status	String describing the status of the operation. Some common values are: Active, Resolved.
operationName	Name of the operation. Usually Microsoft.ServiceHealth/incident/action.
category	"ServiceHealth"
resourceId	Resource id of the impacted resource, if known. Subscription ID is provided otherwise.
Properties.title	The localized title for this communication. English is the default language.
Properties.communication	The localized details of the communication with HTML markup. English is the default.
Properties.incidentType	Possible values: AssistedRecovery, ActionRequired, Information, Incident, Maintenance, Security

ELEMENT NAME	DESCRIPTION
Properties.trackingId	Identifies the incident this event is associated with. Use this to correlate the events related to an incident.
Properties.impactedServices	An escaped JSON blob which describes the services and regions that are impacted by the incident. A list of Services, each of which has a ServiceName and a list of ImpactedRegions, each of which has a RegionName.
Properties.defaultLanguageTitle	The communication in English
Properties.defaultLanguageContent	The communication in English as either html markup or plain text
Properties.stage	Possible values for AssistedRecovery, ActionRequired, Information, Incident, Security: are Active, Resolved. For Maintenance they are: Active, Planned, InProgress, Canceled, Rescheduled, Resolved, Complete
Properties.communicationId	The communication this event is associated.

Alert

This category contains the record of all activations of Azure alerts. An example of the type of event you would see in this category is "CPU % on myVM has been over 80 for the past 5 minutes." A variety of Azure systems have an alerting concept -- you can define a rule of some sort and receive a notification when conditions match that rule. Each time a supported Azure alert type 'activates,' or the conditions are met to generate a notification, a record of the activation is also pushed to this category of the Activity Log.

Sample event

```
{
  "caller": "Microsoft.Insights/alertRules",
  "channels": "Admin, Operation",
  "claims": {
    "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/spn": "Microsoft.Insights/alertRules"
  },
  "correlationId": "/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/microsoft.insights/alertrules/myaler
t/incidents/L3N1YnNjcmldG1vbnMvZGY2MDJjOWMtN2FhMC00MDdkLWE2ZmItZWIyMGM4YmQxMTkyL3Jlc291cmNlR3JvdXBzL0NzbUV2ZW
50RE9HRk9PRC1XZXN0VVVmvcHJvdmlkZXJzL21pY3Jvc29mdC5pbnnPpZ2h0cy9hbGVydHJ1bGVzL215YWxlcnQwNjM2MzYyMjU4NTM1MjIxOTIw
",
  "description": "'Disk read LessThan 100000 ([Count]) in the last 5 minutes' has been resolved for
CloudService: myResourceGroup/Production/Event.BackgroundJobsWorker.razzle (myResourceGroup)",
  "eventDataId": "149d4baf-53dc-4cf4-9e29-17de37405cd9",
  "eventName": {
    "value": "Alert",
    "localizedValue": "Alert"
  },
  "category": {
    "value": "Alert",
    "localizedValue": "Alert"
  },
  "id": "/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/Microsoft.ClassicCompute/domainNames
/myResourceGroup/slots/Production/roles/Event.BackgroundJobsWorker.razzle/events/149d4baf-53dc-4cf4-9e29-
17de37405cd9/ticks/636362258535221920",
  "level": "Informational",
  "resourceGroupName": "myResourceGroup",
  "resourceProviderName": {
    "value": "Microsoft.Insights"
  }
}
```

```

    "value": "Microsoft.ClassicCompute",
    "localizedValue": "Microsoft.ClassicCompute"
},
"resourceId":
"/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/Microsoft.ClassicCompute/domainNames
/myResourceGroup/slots/Production/roles/Event.BackgroundJobsWorker.razzle",
"resourceType": {
    "value": "Microsoft.ClassicCompute/domainNames/slots/roles",
    "localizedValue": "Microsoft.ClassicCompute/domainNames/slots/roles"
},
"operationId":
"/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/microsoft.insights/alertrules/myaler
t/incidents/L3N1YnNjcm1wdG1vbnMvZGY2MDJjOWMtN2FhMC00MDdkLWE2ZmItZWIyMGM4YmQxMTkyL3J1c291cmNlR3JvdXBzL0NzbUV2ZW
50RE9HRk9PRC1XZXN0VVMvcHJvdmlkZXJzL21pY3Jvc29mdC5pbnNpZ2h0cy9hbGVydHJ1bGVzL215YwxlcnQwNjM2MzYyMju4NTM1MjIxOTIw
",
"operationName": {
    "value": "Microsoft.Insights/AlertRules/Resolved/Action",
    "localizedValue": "Microsoft.Insights/AlertRules/Resolved/Action"
},
"properties": {
    "RuleUri": "
        "RuleName": "myalert",
        "RuleDescription": "",
        "Threshold": "100000",
        "WindowSizeInMinutes": "5",
        "Aggregation": "Average",
        "Operator": "LessThan",
        "MetricName": "Disk read",
        "MetricUnit": "Count"
    },
    "status": {
        "value": "Resolved",
        "localizedValue": "Resolved"
    },
    "subStatus": {
        "value": null
    },
    "eventTimestamp": "2017-07-21T09:24:13.522192Z",
    "submissionTimestamp": "2017-07-21T09:24:15.6578651Z",
    "subscriptionId": "mySubscriptionID"
}

```

Property descriptions

ELEMENT NAME	DESCRIPTION
caller	Always Microsoft.Insights/alertRules
channels	Always "Admin, Operation"
claims	JSON blob with the SPN (service principal name), or resource type, of the alert engine.
correlationId	A GUID in the string format.
description	Static text description of the alert event.
eventDataId	Unique identifier of the alert event.

ELEMENT NAME	DESCRIPTION
level	Level of the event. One of the following values: "Critical", "Error", "Warning", "Informational" and "Verbose"
resourceGroupName	Name of the resource group for the impacted resource if it is a metric alert. For other alert types, this is the name of the resource group that contains the alert itself.
resourceProviderName	Name of the resource provider for the impacted resource if it is a metric alert. For other alert types, this is the name of the resource provider for the alert itself.
resourceId	Name of the resource ID for the impacted resource if it is a metric alert. For other alert types, this is the resource ID of the alert resource itself.
operationId	A GUID shared among the events that correspond to a single operation.
operationName	Name of the operation.
properties	Set of <code><Key, Value></code> pairs (that is, a Dictionary) describing the details of the event.
status	String describing the status of the operation. Some common values are: Started, In Progress, Succeeded, Failed, Active, Resolved.
subStatus	Usually null for alerts.
eventTimestamp	Timestamp when the event was generated by the Azure service processing the request corresponding the event.
submissionTimestamp	Timestamp when the event became available for querying.
subscriptionId	Azure Subscription Id.

Properties field per alert type

The properties field will contain different values depending on the source of the alert event. Two common alert event providers are Activity Log alerts and metric alerts.

Properties for Activity Log alerts

ELEMENT NAME	DESCRIPTION
properties.subscriptionId	The subscription ID from the activity log event which caused this activity log alert rule to be activated.
properties.eventDataId	The event data ID from the activity log event which caused this activity log alert rule to be activated.
properties.resourceGroup	The resource group from the activity log event which caused this activity log alert rule to be activated.

ELEMENT NAME	DESCRIPTION
properties.resourceId	The resource ID from the activity log event which caused this activity log alert rule to be activated.
properties.eventTimestamp	The event timestamp of the activity log event which caused this activity log alert rule to be activated.
properties.operationName	The operation name from the activity log event which caused this activity log alert rule to be activated.
properties.status	The status from the activity log event which caused this activity log alert rule to be activated.

Properties for metric alerts

ELEMENT NAME	DESCRIPTION
properties.RuleUri	Resource ID of the metric alert rule itself.
properties.RuleName	The name of the metric alert rule.
properties.RuleDescription	The description of the metric alert rule (as defined in the alert rule).
properties.Threshold	The threshold value used in the evaluation of the metric alert rule.
properties.WindowSizeInMinutes	The window size used in the evaluation of the metric alert rule.
properties.Aggregation	The aggregation type defined in the metric alert rule.
properties.Operator	The conditional operator used in the evaluation of the metric alert rule.
properties.MetricName	The metric name of the metric used in the evaluation of the metric alert rule.
properties.MetricUnit	The metric unit for the metric used in the evaluation of the metric alert rule.

Autoscale

This category contains the record of any events related to the operation of the autoscale engine based on any autoscale settings you have defined in your subscription. An example of the type of event you would see in this category is "Autoscale scale up action failed." Using autoscale, you can automatically scale out or scale in the number of instances in a supported resource type based on time of day and/or load (metric) data using an autoscale setting. When the conditions are met to scale up or down, the start and succeeded or failed events will be recorded in this category.

Sample event

```
{
  "caller": "Microsoft.Insights/autoscaleSettings",
  "channels": "Admin, Operation",
  "claims": {
    "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/spn": "Microsoft.Insights/autoscaleSettings"
  },
  "correlationId": "fc6a7ff5-ff68-4bb7-81b4-3629212d03d0",
  "description": "The autoscale engine attempting to scale resource
'/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/Microsoft.ClassicCompute/domainNames
/myResourceGroup/slots/Production/roles/myResource' from 3 instances count to 2 instances count.",
  "eventDataId": "a5b92075-1de9-42f1-b52e-6f3e4945a7c7",
  "eventName": {
    "value": "AutoscaleAction",
    "localizedValue": "AutoscaleAction"
  },
  "category": {
    "value": "Autoscale",
    "localizedValue": "Autoscale"
  },
  "id": {
    "/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/microsoft.insights/autoscalesettings
/myResourceGroup-Production-myResource-myResourceGroup/events/a5b92075-1de9-42f1-b52e-
6f3e4945a7c7/ticks/636361956518681572",
      "level": "Informational",
      "resourceGroupName": "myResourceGroup",
      "resourceProviderName": {
        "value": "microsoft.insights",
        "localizedValue": "microsoft.insights"
      },
      "resourceId": {
        "/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/microsoft.insights/autoscalesettings
/myResourceGroup-Production-myResource-myResourceGroup",
          "resourceType": {
            "value": "microsoft.insights/autoscalesettings",
            "localizedValue": "microsoft.insights/autoscalesettings"
          },
          "operationId": "fc6a7ff5-ff68-4bb7-81b4-3629212d03d0",
          "operationName": {
            "value": "Microsoft.Insights/AutoscaleSettings/Scaledown/Action",
            "localizedValue": "Microsoft.Insights/AutoscaleSettings/Scaledown/Action"
          },
          "properties": {
            "Description": "The autoscale engine attempting to scale resource
'/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/Microsoft.ClassicCompute/domainNames
/myResourceGroup/slots/Production/roles/myResource' from 3 instances count to 2 instances count.",
            "ResourceName": {
              "/subscriptions/mySubscriptionID/resourceGroups/myResourceGroup/providers/Microsoft.ClassicCompute/domainNames
/myResourceGroup/slots/Production/roles/myResource",
                "OldInstancesCount": "3",
                "NewInstancesCount": "2",
                "LastScaleActionTime": "Fri, 21 Jul 2017 01:00:51 GMT"
              },
              "status": {
                "value": "Succeeded",
                "localizedValue": "Succeeded"
              },
              "subStatus": {
                "value": null
              },
              "eventTimestamp": "2017-07-21T01:00:51.8681572Z",
              "submissionTimestamp": "2017-07-21T01:00:52.3008754Z",
              "subscriptionId": "mySubscriptionID"
            }
          }
        }
      }
    }
  }
}
```

Property descriptions

ELEMENT NAME	DESCRIPTION
caller	Always Microsoft.Insights/autoscaleSettings
channels	Always "Admin, Operation"
claims	JSON blob with the SPN (service principal name), or resource type, of the autoscale engine.
correlationId	A GUID in the string format.
description	Static text description of the autoscale event.
eventDataId	Unique identifier of the autoscale event.
level	Level of the event. One of the following values: "Critical", "Error", "Warning", "Informational" and "Verbose"
resourceGroupName	Name of the resource group for the autoscale setting.
resourceProviderName	Name of the resource provider for the autoscale setting.
resourceId	Resource id of the autoscale setting.
operationId	A GUID shared among the events that correspond to a single operation.
operationName	Name of the operation.
properties	Set of <Key, Value> pairs (that is, a Dictionary) describing the details of the event.
properties.Description	Detailed description of what the autoscale engine was doing.
properties.ResourceName	Resource ID of the impacted resource (the resource on which the scale action was being performed)
properties.OldInstancesCount	The number of instances before the autoscale action took effect.
properties.NewInstancesCount	The number of instances after the autoscale action took effect.
properties.LastScaleActionTime	The timestamp of when the autoscale action occurred.
status	String describing the status of the operation. Some common values are: Started, In Progress, Succeeded, Failed, Active, Resolved.
subStatus	Usually null for autoscale.
eventTimestamp	Timestamp when the event was generated by the Azure service processing the request corresponding the event.
submissionTimestamp	Timestamp when the event became available for querying.

ELEMENT NAME	DESCRIPTION
subscriptionId	Azure Subscription Id.

Next steps

- [Learn more about the Activity Log \(formerly Audit Logs\)](#)
- [Stream the Azure Activity Log to Event Hubs](#)

Azure Diagnostics extention configuration schema versions and history

6/27/2017 • 6 min to read • [Edit Online](#)

This page indexes Azure Diagnostics extension schema versions shipped as part of the Microsoft Azure SDK.

NOTE

The Azure Diagnostics extension is the component used to collect performance counters and other statistics from:

- Azure Virtual Machines
- Virtual Machine Scale Sets
- Service Fabric
- Cloud Services
- Network Security Groups

This page is only relevant if you are using one of these services.

The Azure Diagnostics extension is used with other Microsoft diagnostics products like Azure Monitor, Application Insights, and Log Analytics. For more information see [Microsoft Monitoring Tools Overview](#).

Azure SDK and diagnostics versions shipping chart

AZURE SDK VERSION	DIAGNOSTICS EXTENSION VERSION	MODEL
1.x	1.0	plug-in
2.0 - 2.4	1.0	plug-in
2.5	1.2	extension
2.6	1.3	"
2.7	1.4	"
2.8	1.5	"
2.9	1.6	"
2.96	1.7	"
2.96	1.8	"
2.96	1.8.1	"
2.96	1.9	"

Azure Diagnostics version 1.0 first shipped in a plug-in model -- meaning that when you installed the Azure SDK, you got the version of Azure diagnostics shipped with it.

Starting with SDK 2.5 (diagnostics version 1.2), Azure diagnostics went to an extension model. The tools to utilize new features were only available in newer Azure SDKs, but any service using Azure diagnostics would pick up the latest shipping version directly from Azure. For example, anyone still using SDK 2.5 would be loading the latest version shown in the previous table, regardless if they are using the newer features.

Schemas index

Different versions of Azure diagnostics use different configuration schemas.

[Diagnostics 1.0 Configuration Schema](#)

[Diagnostics 1.2 Configuration Schema](#)

[Diagnostics 1.3 and later Configuration Schema](#)

Version history

Diagnostics extension 1.9

Added Docker support.

Diagnostics extension 1.8.1

Can specify a SAS token instead of a storage account key in the private config. If a SAS token is provided, the storage account key is ignored.

```
{  
    "storageAccountName": "diagstorageaccount",  
    "storageAccountEndPoint": "https://core.windows.net",  
    "storageAccountSasToken": "{sas token}",  
    "SecondaryStorageAccounts": {  
        "StorageAccount": [  
            {  
                "name": "secondarydiagstorageaccount",  
                "endpoint": "https://core.windows.net",  
                "sasToken": "{sas token}"  
            }  
        ]  
    }  
}
```

```
<PrivateConfig>  
    <StorageAccount name="diagstorageaccount" endpoint="https://core.windows.net" sasToken="{sas token}" />  
    <SecondaryStorageAccounts>  
        <StorageAccount name="secondarydiagstorageaccount" endpoint="https://core.windows.net" sasToken="{sas token}" />  
    </SecondaryStorageAccounts>  
</PrivateConfig>
```

Diagnostics extension 1.8

Added Storage Type to PublicConfig. StorageType can be *Table*, *Blob*, *TableAndBlob*. *Table* is the default.

```
{  
    "WadCfg": {  
    },  
    "StorageAccount": "diagstorageaccount",  
    "StorageType": "TableAndBlob"  
}
```

```
<PublicConfig>
  <WadCfg />
  <StorageAccount>diagstorageaccount</StorageAccount>
  <StorageType>TableAndBlob</StorageType>
</PublicConfig>
```

Diagnostics extension 1.7

Added the ability to route to EventHub.

Diagnostics extension 1.5

Added the sinks element and the ability to send diagnostics data to [Application Insights](#) making it easier to diagnose issues across your application as well as the system and infrastructure level.

Azure SDK 2.6 and diagnostics extension 1.3

For Cloud Service projects in Visual Studio, the following changes were made. (These changes also apply to later versions of Azure SDK.)

- The local emulator now supports diagnostics. This means you can collect diagnostics data and ensure your application is creating the right traces while you're developing and testing in Visual Studio. The connection string `UseDevelopmentStorage=true` enables diagnostics data collection while you're running your cloud service project in Visual Studio by using the Azure storage emulator. All diagnostics data is collected in the (Development Storage) storage account.
- The diagnostics storage account connection string (`Microsoft.WindowsAzure.Plugins.Diagnostics.ConnectionString`) is stored once again in the service configuration (.cscfg) file. In Azure SDK 2.5 the diagnostics storage account was specified in the `diagnostics.wadcfgx` file.

There are some notable differences between how the connection string worked in Azure SDK 2.4 and earlier and how it works in Azure SDK 2.6 and later.

- In Azure SDK 2.4 and earlier, the connection string was used as a runtime by the diagnostics plugin to get the storage account information for transferring diagnostics logs.
- In Azure SDK 2.6 and later, the diagnostics connection string is used by Visual Studio to configure the diagnostics extension with the appropriate storage account information during publishing. The connection string lets you define different storage accounts for different service configurations that Visual Studio will use when publishing. However, because the diagnostics plugin is no longer available (after Azure SDK 2.5), the .cscfg file by itself can't enable the Diagnostics Extension. You have to enable the extension separately through tools such as Visual Studio or PowerShell.
- To simplify the process of configuring the diagnostics extension with PowerShell, the package output from Visual Studio also contains the public configuration XML for the diagnostics extension for each role. Visual Studio uses the diagnostics connection string to populate the storage account information present in the public configuration. The public config files are created in the Extensions folder and follow the pattern `PaaS.Diagnostics..PubConfig.xml`. Any PowerShell based deployments can use this pattern to map each configuration to a Role.
- The connection string in the .cscfg file is also used by the Azure portal to access the diagnostics data so it can appear in the **Monitoring** tab. The connection string is needed to configure the service to show verbose monitoring data in the portal.

Migrating projects to Azure SDK 2.6 and later

When migrating from Azure SDK 2.5 to Azure SDK 2.6 or later, if you had a diagnostics storage account specified in the `.wadcfgx` file, then it will stay there. To take advantage of the flexibility of using different storage accounts for different storage configurations, you'll have to manually add the connection string to your project. If you're migrating a project from Azure SDK 2.4 or earlier to Azure SDK 2.6, then the diagnostics connection strings are

preserved. However, please note the changes in how connection strings are treated in Azure SDK 2.6 as specified in the previous section.

How Visual Studio determines the diagnostics storage account

- If a diagnostics connection string is specified in the .cscfg file, Visual Studio uses it to configure the diagnostics extension when publishing, and when generating the public configuration xml files during packaging.
- If no diagnostics connection string is specified in the .cscfg file, then Visual Studio falls back to using the storage account specified in the .wadcfgx file to configure the diagnostics extension when publishing, and generating the public configuration xml files when packaging.
- The diagnostics connection string in the .cscfg file takes precedence over the storage account in the .wadcfgx file. If a diagnostics connection string is specified in the .cscfg file, then Visual Studio uses that and ignores the storage account in .wadcfgx.

What does the "Update development storage connection strings..." checkbox do?

The checkbox for **Update development storage connection strings for Diagnostics and Caching with Microsoft Azure storage account credentials when publishing to Microsoft Azure** gives you a convenient way to update any development storage account connection strings with the Azure storage account specified during publishing.

For example, suppose you select this checkbox and the diagnostics connection string specifies

`UseDevelopmentStorage=true`. When you publish the project to Azure, Visual Studio will automatically update the diagnostics connection string with the storage account you specified in the Publish wizard. However, if a real storage account was specified as the diagnostics connection string, then that account is used instead.

Diagnostics functionality differences between Azure SDK 2.4 and earlier and Azure SDK 2.5 and later

If you're upgrading your project from Azure SDK 2.4 to Azure SDK 2.5 or later, you should bear in mind the following diagnostics functionality differences.

- **Configuration APIs are deprecated** – Programmatic configuration of diagnostics is available in Azure SDK 2.4 or earlier versions, but is deprecated in Azure SDK 2.5 and later. If your diagnostics configuration is currently defined in code, you'll need to reconfigure those settings from scratch in the migrated project in order for diagnostics to keep working. The diagnostics configuration file for Azure SDK 2.4 is diagnostics.wadcfg, and diagnostics.wadcfgx for Azure SDK 2.5 and later.
- **Diagnostics for cloud service applications can only be configured at the role level, not at the instance level.**
- **Every time you deploy your app, the diagnostics configuration is updated** – This can cause parity issues if you change your diagnostics configuration from Server Explorer and then redeploy your app.
- **In Azure SDK 2.5 and later, crash dumps are configured in the diagnostics configuration file, not in code** – If you have crash dumps configured in code, you'll have to manually transfer the configuration from code to the configuration file, because the crash dumps aren't transferred during the migration to Azure SDK 2.6.

Azure Diagnostics 1.0 Configuration Schema

6/27/2017 • 5 min to read • [Edit Online](#)

NOTE

Azure Diagnostics is the component used to collect performance counters and other statistics from Azure Virtual Machines, Virtual Machine Scale Sets, Service Fabric, and Cloud Services. This page is only relevant if you are using one of these services.

Azure Diagnostics is used with other Microsoft diagnostics products like Azure Monitor, Application Insights, and Log Analytics.

The Azure Diagnostics configuration file defines values that are used to initialize the Diagnostics Monitor. This file is used to initialize diagnostic configuration settings when the diagnostics monitor starts.

By default, the Azure Diagnostics configuration schema file is installed to the

`C:\Program Files\Microsoft SDKs\Azure\.NET SDK\<version>\schemas` directory. Replace `<version>` with the installed version of the [Azure SDK](#).

NOTE

The diagnostics configuration file is typically used with startup tasks that require diagnostic data to be collected earlier in the startup process. For more information about using Azure Diagnostics, see [Collect Logging Data by Using Azure Diagnostics](#).

Example of the diagnostics configuration file

The following example shows a typical diagnostics configuration file:

```

<?xml version="1.0" encoding="utf-8"?>
<DiagnosticMonitorConfiguration
  xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration"
    configurationChangePollInterval="PT1M"
    overallQuotaInMB="4096">
  <DiagnosticInfrastructureLogs bufferQuotaInMB="1024"
    scheduledTransferLogLevelFilter="Verbose"
    scheduledTransferPeriod="PT1M" />
  <Logs bufferQuotaInMB="1024"
    scheduledTransferLogLevelFilter="Verbose"
    scheduledTransferPeriod="PT1M" />

  <Directories bufferQuotaInMB="1024"
    scheduledTransferPeriod="PT1M">

    <!-- These three elements specify the special directories
        that are set up for the log types -->
    <CrashDumps container="wad-crash-dumps" directoryQuotaInMB="256" />
    <FailedRequestLogs container="wad-frq" directoryQuotaInMB="256" />
    <IISLogs container="wad-iis" directoryQuotaInMB="256" />

    <!-- For regular directories the DataSources element is used -->
    <DataSources>
      <DirectoryConfiguration container="wad-panther" directoryQuotaInMB="128">
        <!-- Absolute specifies an absolute path with optional environment expansion -->
        <Absolute expandEnvironment="true" path="%SystemRoot%\system32\sysprep\Panther" />
      </DirectoryConfiguration>
      <DirectoryConfiguration container="wad-custom" directoryQuotaInMB="128">
        <!-- LocalResource specifies a path relative to a local
            resource defined in the service definition -->
        <LocalResource name="MyLoggingLocalResource" relativePath="logs" />
      </DirectoryConfiguration>
    </DataSources>
  </Directories>

  <PerformanceCounters bufferQuotaInMB="512" scheduledTransferPeriod="PT1M">
    <!-- The counter specifier is in the same format as the imperative
        diagnostics configuration API -->
    <PerformanceCounterConfiguration
      counterSpecifier="\Processor(_Total)\% Processor Time" sampleRate="PT5S" />
  </PerformanceCounters>

  <WindowsEventLog bufferQuotaInMB="512"
    scheduledTransferLogLevelFilter="Verbose"
    scheduledTransferPeriod="PT1M" />
    <!-- The event log name is in the same format as the imperative
        diagnostics configuration API -->
    <DataSource name="System!*" />
  </WindowsEventLog>
</DiagnosticMonitorConfiguration>

```

DiagnosticsConfiguration Namespace

The XML namespace for the diagnostics configuration file is:

<http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration>

Schema Elements

The diagnostics configuration file includes the following elements.

DiagnosticMonitorConfiguration Element

The top-level element of the diagnostics configuration file.

Attributes:

ATTRIBUTE	TYPE	REQUIRED	DEFAULT	DESCRIPTION
configurationChangePollInterval	duration	Optional	PT1M	Specifies the interval at which the diagnostic monitor polls for diagnostic configuration changes.
overallQuotaInMB	unsignedInt	Optional	4000 MB. If you provide a value, it must not exceed this amount	The total amount of file system storage allocated for all logging buffers.

DiagnosticInfrastructureLogs Element

Defines the buffer configuration for the logs that are generated by the underlying diagnostics infrastructure.

Parent Element: [DiagnosticMonitorConfiguration Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
bufferQuotaInMB	unsignedInt	Optional. Specifies the maximum amount of file system storage that is available for the specified data. The default is 0.
scheduledTransferLogLevelFilter	string	Optional. Specifies the minimum severity level for log entries that are transferred. The default value is Undefined . Other possible values are Verbose , Information , Warning , Error , and Critical .
scheduledTransferPeriod	duration	Optional. Specifies the interval between scheduled transfers of data, rounded up to the nearest minute. The default is PT0S.

Logs Element

Defines the buffer configuration for basic Azure logs.

Parent element: [DiagnosticMonitorConfiguration Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
bufferQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum amount of file system storage that is available for the specified data.</p> <p>The default is 0.</p>
scheduledTransferLogLevelFilter	string	<p>Optional. Specifies the minimum severity level for log entries that are transferred. The default value is Undefined. Other possible values are Verbose, Information, Warning, Error, and Critical.</p>
scheduledTransferPeriod	duration	<p>Optional. Specifies the interval between scheduled transfers of data, rounded up to the nearest minute.</p> <p>The default is PT0S.</p>

Directories Element

Defines the buffer configuration for file-based logs that you can define.

Parent element: [DiagnosticMonitorConfiguration Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
bufferQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum amount of file system storage that is available for the specified data.</p> <p>The default is 0.</p>
scheduledTransferPeriod	duration	<p>Optional. Specifies the interval between scheduled transfers of data, rounded up to the nearest minute.</p> <p>The default is PT0S.</p>

CrashDumps Element

Defines the crash dumps directory.

Parent Element: [Directories Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
container	string	<p>The name of the container where the contents of the directory is to be transferred.</p>

ATTRIBUTE	TYPE	DESCRIPTION
directoryQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum size of the directory in megabytes.</p> <p>The default is 0.</p>

FailedRequestLogs Element

Defines the failed request log directory.

Parent Element [Directories Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
container	string	The name of the container where the contents of the directory is to be transferred.
directoryQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum size of the directory in megabytes.</p> <p>The default is 0.</p>

IISLogs Element

Defines the IIS log directory.

Parent Element [Directories Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
container	string	The name of the container where the contents of the directory is to be transferred.
directoryQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum size of the directory in megabytes.</p> <p>The default is 0.</p>

DataSources Element

Defines zero or more additional log directories.

Parent Element: [Directories Element](#).

DirectoryConfiguration Element

Defines the directory of log files to monitor.

Parent Element: [DataSources Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
container	string	The name of the container where the contents of the directory is to be transferred.
directoryQuotaInMB	unsignedInt	Optional. Specifies the maximum size of the directory in megabytes. The default is 0.

Absolute Element

Defines an absolute path of the directory to monitor with optional environment expansion.

Parent Element: [DirectoryConfiguration Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
path	string	Required. The absolute path to the directory to monitor.
expandEnvironment	boolean	Required. If set to true , environment variables in the path are expanded.

LocalResource Element

Defines a path relative to a local resource defined in the service definition.

Parent Element: [DirectoryConfiguration Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
name	string	Required. The name of the local resource that contains the directory to monitor.
relativePath	string	Required. The path relative to the local resource to monitor.

PerformanceCounters Element

Defines the path to the performance counter to collect.

Parent Element: [DiagnosticMonitorConfiguration Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
bufferQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum amount of file system storage that is available for the specified data.</p> <p>The default is 0.</p>
scheduledTransferPeriod	duration	<p>Optional. Specifies the interval between scheduled transfers of data, rounded up to the nearest minute.</p> <p>The default is PT0S.</p>

PerformanceCounterConfiguration Element

Defines the performance counter to collect.

Parent Element: [PerformanceCounters Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
counterSpecifier	string	Required. The path to the performance counter to collect.
sampleRate	duration	Required. The rate at which the performance counter should be collected.

WindowsEventLog Element

Defines the event logs to monitor.

Parent Element: [DiagnosticMonitorConfiguration Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
bufferQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum amount of file system storage that is available for the specified data.</p> <p>The default is 0.</p>
scheduledTransferLogLevelFilter	string	Optional. Specifies the minimum severity level for log entries that are transferred. The default value is Undefined . Other possible values are Verbose , Information , Warning , Error , and Critical .

ATTRIBUTE	TYPE	DESCRIPTION
scheduledTransferPeriod	duration	<p>Optional. Specifies the interval between scheduled transfers of data, rounded up to the nearest minute.</p> <p>The default is PT0S.</p>

DataSource Element

Defines the event log to monitor.

Parent Element: [WindowsEventLog Element](#).

Attributes:

ATTRIBUTE	TYPE	DESCRIPTION
name	string	Required. An XPath expression specifying the log to collect.

Azure Diagnostics 1.2 configuration schema

6/27/2017 • 7 min to read • [Edit Online](#)

NOTE

Azure Diagnostics is the component used to collect performance counters and other statistics from Azure Virtual Machines, Virtual Machine Scale Sets, Service Fabric, and Cloud Services. This page is only relevant if you are using one of these services.

Azure Diagnostics is used with other Microsoft diagnostics products like Azure Monitor, Application Insights, and Log Analytics.

This schema defines the possible values you can use to initialize diagnostic configuration settings when the diagnostics monitor starts.

Download the public configuration file schema definition by executing the following PowerShell command:

```
(Get-AzureServiceAvailableExtension -ExtensionName 'PaaSDiagnostics' -ProviderNamespace 'Microsoft.Azure.Diagnostics').PublicConfigurationSchema | Out-File -Encoding utf8 -FilePath 'C:\temp\WadConfig.xsd'
```

For more information about using Azure Diagnostics, see [Enabling Diagnostics in Azure Cloud Services](#).

Example of the diagnostics configuration file

The following example shows a typical diagnostics configuration file:

```

<?xml version="1.0" encoding="utf-8"?>
<PublicConfig xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration">
  <WadCfg>
    <DiagnosticMonitorConfiguration overallQuotaInMB="10000">
      <PerformanceCounters scheduledTransferPeriod="PT1M">
        <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Processor Time" sampleRate="PT1M"
unit="percent" />
      </PerformanceCounters>
      <Directories scheduledTransferPeriod="PT5M">
        <IISLogs containerName="iislogs" />
        <FailedRequestLogs containerName="iisfailed" />
      <DataSources>
        <DirectoryConfiguration containerName="mynewprocess">
          <Absolute path="C:\MyNewProcess" expandEnvironment="false" />
        </DirectoryConfiguration>
        <DirectoryConfiguration containerName="badapp">
          <Absolute path="%SYSTEMDRIVE%\BadApp" expandEnvironment="true" />
        </DirectoryConfiguration>
        <DirectoryConfiguration containerName="goodapp">
          <LocalResource name="Skippy" relativePath="..\PeanutButter"/>
        </DirectoryConfiguration>
      </DataSources>
    </Directories>
    <EtwProviders>
      <EtwEventSourceProviderConfiguration provider="MyProviderClass" scheduledTransferPeriod="PT5M">
        <Event id="0"/>
        <Event id="1" eventDestination="errorTable"/>
        <DefaultEvents />
      </EtwEventSourceProviderConfiguration>
      <EtwManifestProviderConfiguration provider="5974b00b-84c2-44bc-9e58-3a2451b4e3ad"
scheduledTransferLogLevelFilter="Information" scheduledTransferPeriod="PT2M">
        <Event id="0"/>
        <DefaultEvents eventDestination="defaultTable"/>
      </EtwManifestProviderConfiguration>
    </EtwProviders>
    <WindowsEventLog scheduledTransferPeriod="PT5M">
      <DataSource name="System!*[System[Provider[@Name='Microsoft Antimalware']]://" />
      <DataSource name="System!*[System[Provider[@Name='NTFS'] and (EventID=55)]]" />
      <DataSource name="System!*[System[Provider[@Name='disk'] and (EventID=7 or EventID=52 or EventID=55)]]" />
    </WindowsEventLog>
    <CrashDumps containerName="wad-crashdumps" directoryQuotaPercentage="30" dumpType="Mini">
      <CrashDumpConfiguration processName="mynewprocess.exe" />
      <CrashDumpConfiguration processName="badapp.exe" />
    </CrashDumps>
  </DiagnosticMonitorConfiguration>
  </WadCfg>
</PublicConfig>

```

Diagnostics Configuration Namespace

The XML namespace for the diagnostics configuration file is:

<http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration>

PublicConfig Element

Top-level element of the diagnostics configuration file. The following table describes the elements of the configuration file.

ELEMENT NAME	DESCRIPTION
WadCfg	Required. Configuration settings for the telemetry data to be collected.
StorageAccount	The name of the Azure Storage account to store the data in. This may also be specified as a parameter when executing the Set-AzureServiceDiagnosticsExtension cmdlet.

ELEMENT NAME	DESCRIPTION
LocalResourceDirectory	<p>The directory on the virtual machine to be used by the Monitoring Agent to store event data. If not set, the default directory is used:</p> <p>For a Worker/web role: C:\Resources\<guid>\directory\<guid>. <RoleName>.DiagnosticStore\</p> <p>For a Virtual Machine: C:\WindowsAzure\Logs\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnos<WADVersion>\WAD<WADVersion></p> <p>Required attributes are:</p> <ul style="list-style-type: none"> - path - The directory on the system to be used by Azure Diagnostics. - expandEnvironment - Controls whether environment variables are expanded in the path name.

WadCFG Element

Defines configuration settings for the telemetry data to be collected. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
DiagnosticMonitorConfiguration	<p>Required. Optional attributes are:</p> <ul style="list-style-type: none"> - overallQuotaInMB - The maximum amount of local disk space that may be consumed by the various types of diagnostic data collected by Azure Diagnostics. The default setting is 5120MB. - useProxyServer - Configure Azure Diagnostics to use the proxy server settings as set in IE settings.
CrashDumps	<p>Enable collection of crash dumps. Optional attributes are:</p> <ul style="list-style-type: none"> - containerName - The name of the blob container in your Azure Storage account to be used to store crash dumps. - crashDumpType - Configures Azure Diagnostics to collect Mini or Full crash dumps. - directoryQuotaPercentage - Configures the percentage of overallQuotaInMB to be reserved for crash dumps on the VM.
DiagnosticInfrastructureLogs	<p>Enable collection of logs generated by Azure Diagnostics. The diagnostic infrastructure logs are useful for troubleshooting the diagnostics system itself. Optional attributes are:</p> <ul style="list-style-type: none"> - scheduledTransferLogLevelFilter - Configures the minimum severity level of the logs collected. - scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. The value is an XML "Duration Data Type."
Directories	<p>Enables the collection of the contents of a directory, IIS failed access request logs and/or IIS logs. Optional attribute:</p> <p>scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. The value is an XML "Duration Data Type."</p>

ELEMENT NAME	DESCRIPTION
EtwProviders	Configures collection of ETW events from EventSource and/or ETW Manifest based providers.
Metrics	This element enables you to generate a performance counter table that is optimized for fast queries. Each performance counter that is defined in the PerformanceCounters element is stored in the Metrics table in addition to the Performance Counter table. Required attribute: resourceId - This is the resource ID of the Virtual Machine you are deploying Azure Diagnostics to. Get the resourceID from the Azure portal . Select Browse -> Resource Groups -> . Click the Properties tile and copy the value from the ID field.
PerformanceCounters	Enables the collection of performance counters. Optional attribute: scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. Value is an XML "Duration Data Type" .
WindowsEventLog	Enables the collection of Windows Event Logs. Optional attribute: scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. Value is an XML "Duration Data Type" .

CrashDumps Element

Enables collection of crash dumps. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
CrashDumpConfiguration	Required. Required attribute: processName - The name of the process you want Azure Diagnostics to collect a crash dump for.
crashDumpType	Configures Azure Diagnostics to collect mini or full crash dumps.
directoryQuotaPercentage	Configures the percentage of overallQuotaInMB to be reserved for crash dumps on the VM.

Directories Element

Enables the collection of the contents of a directory, IIS failed access request logs and/or IIS logs. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
DataSources	A list of directories to monitor.
FailedRequestLogs	Including this element in the configuration enables collection of logs about failed requests to an IIS site or application. You must also enable tracing options under system.WebServer in Web.config .
IISLogs	Including this element in the configuration enables the collection of IIS logs: containerName - The name of the blob container in your Azure Storage account to be used to store the IIS logs.

DataSources Element

A list of directories to monitor. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
DirectoryConfiguration	Required. Required attribute: containerName - The name of the blob container in your Azure Storage account to be used to store the log files.

DirectoryConfiguration Element

DirectoryConfiguration may include either the **Absolute** or **LocalResource** element but not both. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
Absolute	The absolute path to the directory to monitor. The following attributes are required: - Path - The absolute path to the directory to monitor. - expandEnvironment - Configures whether environment variables in Path are expanded.
LocalResource	The path relative to a local resource to monitor. Required attributes are: - Name - The local resource that contains the directory to monitor - relativePath - The path relative to Name that contains the directory to monitor

EtwProviders Element

Configures collection of ETW events from EventSource and/or ETW Manifest based providers. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
EtwEventSourceProviderConfiguration	Configures collection of events generated from EventSource Class . Required attribute: provider - The class name of the EventSource event. Optional attributes are: - scheduledTransferLogLevelFilter - The minimum severity level to transfer to your storage account. - scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. Value is an XML Duration Data Type .

ELEMENT NAME	DESCRIPTION
EtwManifestProviderConfiguration	<p>Required attribute:</p> <p>provider - The GUID of the event provider</p> <p>Optional attributes are:</p> <ul style="list-style-type: none"> - scheduledTransferLogLevelFilter - The minimum severity level to transfer to your storage account. - scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. Value is an XML Duration Data Type.

EtwEventSourceProviderConfiguration Element

Configures collection of events generated from [EventSource Class](#). The following table describes child elements:

ELEMENT NAME	DESCRIPTION
DefaultEvents	<p>Optional attribute:</p> <p>eventDestination - The name of the table to store the events in</p>
Event	<p>Required attribute:</p> <p>id - The id of the event.</p> <p>Optional attribute:</p> <p>eventDestination - The name of the table to store the events in</p>

EtwManifestProviderConfiguration Element

The following table describes child elements:

ELEMENT NAME	DESCRIPTION
DefaultEvents	<p>Optional attribute:</p> <p>eventDestination - The name of the table to store the events in</p>
Event	<p>Required attribute:</p> <p>id - The id of the event.</p> <p>Optional attribute:</p> <p>eventDestination - The name of the table to store the events in</p>

Metrics Element

Enables you to generate a performance counter table that is optimized for fast queries. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
--------------	-------------

ELEMENT NAME	DESCRIPTION
MetricAggregation	<p>Required attribute:</p> <p>scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. Value is an XML Duration Data Type.</p>

PerformanceCounters Element

Enables the collection of performance counters. The following table describes child elements:

ELEMENT NAME	DESCRIPTION
PerformanceCounterConfiguration	<p>The following attributes are required:</p> <ul style="list-style-type: none"> - counterSpecifier - The name of the performance counter. For example, <code>\Processor(_Total)\% Processor Time</code>. To get a list of performance counters on your host run the command <code>typeperf</code>. - sampleRate - How often the counter should be sampled. <p>Optional attribute:</p> <p>unit - The unit of measure of the counter.</p>

PerformanceCounterConfiguration Element

The following table describes child elements:

ELEMENT NAME	DESCRIPTION
annotation	<p>Required attribute:</p> <p>displayName - The display name for the counter</p> <p>Optional attribute:</p> <p>locale - The locale to use when displaying the counter name</p>

WindowsEventLog Element

The following table describes child elements:

ELEMENT NAME	DESCRIPTION
DataSource	<p>The Windows Event logs to collect. Required attribute:</p> <p>name - The XPath query describing the windows events to be collected. For example:</p> <pre>Application!*[System[(Level >= 3)]], System!* [System[(Level <=3)]], System!* [System[Provider[@Name='Microsoft Antimalware']]], Security!*[System[(Level >= 3)]]</pre> <p>To collect all events, specify <code>"*"</code>.</p>

Azure Diagnostics 1.3 and later configuration schema

6/27/2017 • 13 min to read • [Edit Online](#)

NOTE

The Azure Diagnostics extension is the component used to collect performance counters and other statistics from:

- Azure Virtual Machines
- Virtual Machine Scale Sets
- Service Fabric
- Cloud Services
- Network Security Groups

This page is only relevant if you are using one of these services.

This page is valid for versions 1.3 and newer (Azure SDK 2.4 and newer). Newer configuration sections are commented to show in what version they were added.

The configuration file described here is used to set diagnostic configuration settings when the diagnostics monitor starts.

The extension is used in conjunction with other Microsoft diagnostics products like Azure Monitor, Application Insights, and Log Analytics.

Download the public configuration file schema definition by executing the following PowerShell command:

```
(Get-AzureServiceAvailableExtension -ExtensionName 'PaaSDiagnostics' -ProviderNamespace 'Microsoft.Azure.Diagnostics').PublicConfigurationSchema | Out-File -Encoding utf8 -FilePath 'C:\temp\WadConfig.xsd'
```

For more information about using Azure Diagnostics, see [Azure Diagnostics Extension](#).

Example of the diagnostics configuration file

The following example shows a typical diagnostics configuration file:

```
<?xml version="1.0" encoding="utf-8"?>
<DiagnosticsConfiguration xmlns="http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration">
    <PublicConfig>
        <WadCfg>
            <DiagnosticMonitorConfiguration overallQuotaInMB="10000">

                <PerformanceCounters scheduledTransferPeriod="PT1M">
                    <PerformanceCounterConfiguration counterSpecifier="\Processor(_Total)\% Processor Time" sampleRate="PT1M" unit="percent" />
                </PerformanceCounters>

                <Directories scheduledTransferPeriod="PT5M">
                    <IISLogs containerName="iislogs" />
                    <FailedRequestLogs containerName="iisfailed" />

                    <DataSources>
                        <DirectoryConfiguration containerName="mynewprocess">
                            <Absolute path="C:\MyNewProcess" expandEnvironment="false" />
                        </DirectoryConfiguration>
                        <DirectoryConfiguration containerName="badapp">
                            <Absolute path="%SYSTEMDRIVE%\BadApp" expandEnvironment="true" />
                        </DirectoryConfiguration>
                        <DirectoryConfiguration containerName="goodapp">
                            <LocalResource name="Skippy" relativePath="..\PeanutButter"/>
                        </DirectoryConfiguration>
                    </DataSources>
                </Directories>
            </DiagnosticMonitorConfiguration>
        </WadCfg>
    </PublicConfig>
</DiagnosticsConfiguration>
```

```

<EtwProviders>
    <EtwEventSourceProviderConfiguration
        provider="MyProviderClass"
        scheduledTransferPeriod="PT5M">
        <Event id="0"/>
        <Event id="1" eventDestination="errorTable"/>
        <DefaultEvents />
    </EtwEventSourceProviderConfiguration>
    <EtwManifestProviderConfiguration provider="5974b00b-84c2-44bc-9e58-3a2451b4e3ad"
scheduledTransferLogLevelFilter="Information" scheduledTransferPeriod="PT2M">
        <Event id="0"/>
        <DefaultEvents eventDestination="defaultTable"/>
    </EtwManifestProviderConfiguration>
</EtwProviders>

<WindowsEventLog scheduledTransferPeriod="PT5M">
    <DataSource name="System!*[System[Provider[@Name='Microsoft Antimalware']]]" />
    <DataSource name="System!*[System[Provider[@Name='NTFS'] and (EventID=55)]]" />
    <DataSource name="System!*[System[Provider[@Name='disk'] and (EventID=7 or EventID=52 or EventID=55)]]" />
</WindowsEventLog>

<Logs bufferQuotaInMB="1024"
      scheduledTransferPeriod="PT1M"
      scheduledTransferLogLevelFilter="Verbose"
      sinks="ApplicationInsights.AppLogs"/> <!-- sinks attribute added in 1.5 -->

<CrashDumps containerName="wad-crashdumps" directoryQuotaPercentage="30" dumpType="Mini">
    <CrashDumpConfiguration processName="mynewprocess.exe" />
    <CrashDumpConfiguration processName="badapp.exe"/>
</CrashDumps>

<DockerSources> <!-- Added in 1.9 -->
    <Stats enabled="true" sampleRate="PT1M" scheduledTransferPeriod="PT1M" />
</DockerSources>

</DiagnosticMonitorConfiguration>

<SinksConfig> <!-- Added in 1.5 -->
    <Sink name="ApplicationInsights">
        <ApplicationInsights>{Insert InstrumentationKey}</ApplicationInsights>
        <Channels>
            <Channel logLevel="Error" name="Errors" />
            <Channel logLevel="Verbose" name="AppLogs" />
        </Channels>
    </Sink>
    <Sink name="EventHub"> <!-- Added in 1.7 -->
        <EventHub Url="https://myeventhub-ns.servicebus.windows.net/diageventhub" SharedAccessKeyName="SendRule"
usePublisherId="false" />
    </Sink>
    <Sink name="SecondaryEventHub"> <!-- Added in 1.7 -->
        <EventHub Url="https://myeventhub-ns.servicebus.windows.net/secondarydiageventhub"
SharedAccessKeyName="SendRule" usePublisherId="false" />
    </Sink>
    <Sink name="secondaryStorageAccount"> <!-- Added in 1.7 -->
        <StorageAccount name="secondarydiagstorageaccount" endpoint="https://core.windows.net" />
    </Sink>
</SinksConfig>

</WadCfg>

<StorageAccount>diagstorageaccount</StorageAccount>
<StorageType>TableAndBlob</StorageType> <!-- Added in 1.8 -->
</PublicConfig>

<PrivateConfig> <!-- Added in 1.3 -->
    <StorageAccount name="" key="" endpoint="" sasToken="{sas token}" /> <!-- sasToken in Private config added in 1.8.1
-->
    <EventHub Url="https://myeventhub-ns.servicebus.windows.net/diageventhub" SharedAccessKeyName="SendRule"
SharedAccessKey="{base64 encoded key}" />

    <SecondaryStorageAccounts>
        <StorageAccount name="secondarydiagstorageaccount" key="{base64 encoded key}" endpoint="https://core.windows.net"
sasToken="{sas token}" />
    </SecondaryStorageAccounts>

```

```
<SecondaryEventHubs>
    <EventHub Url="https://myeventhub-ns.servicebus.windows.net/secondarydiageventhub" SharedAccessKeyName="SendRule"
SharedAccessKey="{base64 encoded key}" />
</SecondaryEventHubs>

</PrivateConfig>
<IsEnabled>true</IsEnabled>
</DiagnosticsConfiguration>
```

JSON equivalent of the previous XML configuration file.

The PublicConfig and PrivateConfig are separated because in most json usage cases, they are passed as different variables. These cases include Resource Manager templates, Virtual Machine Scale set PowerShell, and Visual Studio.

```
"PublicConfig" {
    "WadCfg": {
        "DiagnosticMonitorConfiguration": {
            "overallQuotaInMB": 10000,
            "DiagnosticInfrastructureLogs": {
                "scheduledTransferLogLevelFilter": "Error"
            },
            "PerformanceCounters": {
                "scheduledTransferPeriod": "PT1M",
                "PerformanceCounterConfiguration": [
                    {
                        "counterSpecifier": "\Processor(_Total)\% Processor Time",
                        "sampleRate": "PT1M",
                        "unit": "percent"
                    }
                ]
            },
            "Directories": {
                "scheduledTransferPeriod": "PT5M",
                "IISLogs": {
                    "containerName": "iislogs"
                },
                "FailedRequestLogs": {
                    "containerName": "iisfailed"
                },
                "DataSources": [
                    {
                        "containerName": "mynewprocess",
                        "Absolute": {
                            "path": "C:\MyNewProcess",
                            "expandEnvironment": false
                        }
                    },
                    {
                        "containerName": "badapp",
                        "Absolute": {
                            "path": "%SYSTEMDRIVE%\BadApp",
                            "expandEnvironment": true
                        }
                    },
                    {
                        "containerName": "goodapp",
                        "LocalResource": {
                            "relativePath": "..\PeanutButter",
                            "name": "Skippy"
                        }
                    }
                ]
            },
            "EtwProviders": {
                "sinks": "",
                "EtwEventSourceProviderConfiguration": [
                    {
                        "scheduledTransferPeriod": "PT5M",
                        "provider": "MyProviderClass",
                        "Event": [
                            {
                                "id": 0
                            },
                            {
                                "id": 1
                            }
                        ]
                    }
                ]
            }
        }
    }
}
```

```

        "id": 1,
        "eventDestination": "errorTable"
    }
],
"DefaultEvents": {
}
},
"EtwManifestProviderConfiguration": [
{
    "scheduledTransferPeriod": "PT2M",
    "scheduledTransferLogLevelFilter": "Information",
    "provider": "5974b00b-84c2-44bc-9e58-3a2451b4e3ad",
    "Event": [
        {
            "id": 0
        }
    ],
    "DefaultEvents": {
    }
}
]
},
"WindowsEventLog": {
    "scheduledTransferPeriod": "PT5M",
    "DataSource": [
        {
            "name": "System!*[System[Provider[@Name='Microsoft Antimalware']]}"
        },
        {
            "name": "System!*[System[Provider[@Name='NTFS'] and (EventID=55)]]"
        },
        {
            "name": "System!*[System[Provider[@Name='disk'] and (EventID=7 or EventID=52 or EventID=55)]]"
        }
    ]
},
"Logs": {
    "scheduledTransferPeriod": "PT1M",
    "scheduledTransferLogLevelFilter": "Verbose",
    "sinks": "ApplicationInsights.AppLogs"
},
"CrashDumps": {
    "directoryQuotaPercentage": 30,
    "dumpType": "Mini",
    "containerName": "wad-crashdumps",
    "CrashDumpConfiguration": [
        {
            "processName": "mynewprocess.exe"
        },
        {
            "processName": "badapp.exe"
        }
    ]
},
"SinksConfig": {
    "Sink": [
    {
        "name": "ApplicationInsights",
        "ApplicationInsights": "{Insert InstrumentationKey}",
        "Channels": {
            "Channel": [
                {
                    "logLevel": "Error",
                    "name": "Errors"
                },
                {
                    "logLevel": "Verbose",
                    "name": "AppLogs"
                }
            ]
        }
    },
    {
        "name": "File"
    }
]
}
}

```

```

        "name": "EventHub",
        "EventHub": {
            "Url": "https://myeventhub-ns.servicebus.windows.net/diageventhub",
            "SharedAccessKeyName": "SendRule",
            "usePublisherId": false
        }
    },
    {
        "name": "secondaryEventHub",
        "EventHub": {
            "Url": "https://myeventhub-ns.servicebus.windows.net/secondarydiageventhub",
            "SharedAccessKeyName": "SendRule",
            "usePublisherId": false
        }
    },
    {
        "name": "secondaryStorageAccount",
        "StorageAccount": {
            "name": "secondarydiagstorageaccount",
            "endpoint": "https://core.windows.net"
        }
    }
]
},
"StorageAccount": "diagstorageaccount",
"StorageType": "TableAndBlob"
}

```

```

"PrivateConfig" {
    "storageAccountName": "diagstorageaccount",
    "storageAccountKey": "{base64 encoded key}",
    "storageAccountEndPoint": "https://core.windows.net",
    "storageAccountSasToken": "{sas token}",
    "EventHub": {
        "Url": "https://myeventhub-ns.servicebus.windows.net/diageventhub",
        "SharedAccessKeyName": "SendRule",
        "SharedAccessKey": "{base64 encoded key}"
    },
    "SecondaryStorageAccounts": {
        "StorageAccount": [
            {
                "name": "secondarydiagstorageaccount",
                "key": "{base64 encoded key}",
                "endpoint": "https://core.windows.net",
                "sasToken": "{sas token}"
            }
        ]
    },
    "SecondaryEventHubs": {
        "EventHub": [
            {
                "Url": "https://myeventhub-ns.servicebus.windows.net/secondarydiageventhub",
                "SharedAccessKeyName": "SendRule",
                "SharedAccessKey": "{base64 encoded key}"
            }
        ]
    }
}

```

Reading this page

The tags following are roughly in order shown in the preceding example. If you don't see a full description where you expect it, search the page for the element or attribute.

Common Attribute Types

scheduledTransferPeriod attribute appears in several elements. It is the interval between scheduled transfers to storage rounded up to the nearest minute. The value is an [XML "Duration Data Type"](#).

DiagnosticsConfiguration Element

Tree: Root - *DiagnosticsConfiguration*

Added in version 1.3.

The top-level element of the diagnostics configuration file.

Attribute xmlns - The XML namespace for the diagnostics configuration file is:

<http://schemas.microsoft.com/ServiceHosting/2010/10/DiagnosticsConfiguration>

CHILD ELEMENTS	DESCRIPTION
PublicConfig	Required. See description elsewhere on this page.
PrivateConfig	Optional. See description elsewhere on this page.
.IsEnabled	Boolean. See description elsewhere on this page.

PublicConfig Element

Tree: Root - *DiagnosticsConfiguration* - *PublicConfig*

Describes the public diagnostics configuration.

CHILD ELEMENTS	DESCRIPTION
WadCfg	Required. See description elsewhere on this page.
StorageAccount	The name of the Azure Storage account to store the data in. May also be specified as a parameter when executing the Set-AzureServiceDiagnosticsExtension cmdlet.
StorageType	Can be <i>Table</i> , <i>Blob</i> , or <i>TableAndBlob</i> . Table is default. When TableAndBlob is chosen, diagnostic data is written twice -- once to each type.
LocalResourceDirectory	<p>The directory on the virtual machine where the Monitoring Agent stores event data. If not, set, the default directory is used:</p> <p>For a Worker/web role: C:\Resources\<guid>\directory\<guid>. <RoleName>.DiagnosticStore\</p> <p>For a Virtual Machine: C:\WindowsAzure\Logs\Plugins\Microsoft.Azure.Diagnostics.IaaSDiagnos <WADVersion>\WAD<WADVersion></p> <p>Required attributes are:</p> <ul style="list-style-type: none">- path - The directory on the system to be used by Azure Diagnostics.- expandEnvironment - Controls whether environment variables are expanded in the path name.

WadCFG Element

Tree: Root - *DiagnosticsConfiguration* - *PublicConfig* - *WadCFG*

Identifies and configures the telemetry data to be collected.

DiagnosticMonitorConfiguration Element

Required

ATTRIBUTES	DESCRIPTION
overallQuotaInMB	The maximum amount of local disk space that may be consumed by the various types of diagnostic data collected by Azure Diagnostics. The default setting is 5120 MB.
useProxyServer	Configure Azure Diagnostics to use the proxy server settings as set in IE settings.

CHILD ELEMENTS	DESCRIPTION
CrashDumps	See description elsewhere on this page.
DiagnosticInfrastructureLogs	Enable collection of logs generated by Azure Diagnostics. The diagnostic infrastructure logs are useful for troubleshooting the diagnostics system itself. Optional attributes are: - scheduledTransferLogLevelFilter - Configures the minimum severity level of the logs collected. - scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. The value is an XML "Duration Data Type" .
Directories	See description elsewhere on this page.
EtwProviders	See description elsewhere on this page.
Metrics	See description elsewhere on this page.
PerformanceCounters	See description elsewhere on this page.
WindowsEventLog	See description elsewhere on this page.
DockerSources	See description elsewhere on this page.

CrashDumps Element

Enable the collection of crash dumps.

ATTRIBUTES	DESCRIPTION
containerName	Optional. The name of the blob container in your Azure Storage account to be used to store crash dumps.
crashDumpType	Optional. Configures Azure Diagnostics to collect mini or full crash dumps.
directoryQuotaPercentage	Optional. Configures the percentage of overallQuotaInMB to be reserved for crash dumps on the VM.

CHILD ELEMENTS	DESCRIPTION
CrashDumpConfiguration	<p>Required. Defines configuration values for each process.</p> <p>The following attribute is also required:</p> <p>processName - The name of the process you want Azure Diagnostics to collect a crash dump for.</p>

Directories Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - Directories

Enables the collection of the contents of a directory, IIS failed access request logs and/or IIS logs.

Optional **scheduledTransferPeriod** attribute. See explanation earlier.

CHILD ELEMENTS	DESCRIPTION
IISLogs	<p>Including this element in the configuration enables the collection of IIS logs:</p> <p>containerName - The name of the blob container in your Azure Storage account to be used to store the IIS logs.</p>
FailedRequestLogs	<p>Including this element in the configuration enables collection of logs about failed requests to an IIS site or application. You must also enable tracing options under system.WebServer in Web.config.</p>
DataSources	A list of directories to monitor.

DataSources Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - Directories - DataSources

A list of directories to monitor.

CHILD ELEMENTS	DESCRIPTION
DirectoryConfiguration	<p>Required. Required attribute:</p> <p>containerName - The name of the blob container in your Azure Storage account that to be used to store the log files.</p>

DirectoryConfiguration Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - Directories - DataSources - DirectoryConfiguration

May include either the **Absolute** or **LocalResource** element but not both.

CHILD ELEMENTS	DESCRIPTION
Absolute	<p>The absolute path to the directory to monitor. The following attributes are required:</p> <ul style="list-style-type: none"> - Path - The absolute path to the directory to monitor. - expandEnvironment - Configures whether environment variables in Path are expanded.

CHILD ELEMENTS	DESCRIPTION
LocalResource	<p>The path relative to a local resource to monitor. Required attributes are:</p> <ul style="list-style-type: none"> - Name - The local resource that contains the directory to monitor - relativePath - The path relative to Name that contains the directory to monitor

EtwProviders Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - EtwProviders

Configures collection of ETW events from EventSource and/or ETW Manifest based providers.

CHILD ELEMENTS	DESCRIPTION
EtwEventSourceProviderConfiguration	<p>Configures collection of events generated from EventSource Class. Required attribute:</p> <p>provider - The class name of the EventSource event.</p> <p>Optional attributes are:</p> <ul style="list-style-type: none"> - scheduledTransferLogLevelFilter - The minimum severity level to transfer to your storage account. - scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. The value is an XML "Duration Data Type."
EtwManifestProviderConfiguration	<p>Required attribute:</p> <p>provider - The GUID of the event provider</p> <p>Optional attributes are:</p> <ul style="list-style-type: none"> - scheduledTransferLogLevelFilter - The minimum severity level to transfer to your storage account. - scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. The value is an XML "Duration Data Type."

EtwEventSourceProviderConfiguration Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - EtwProviders - EtwEventSourceProviderConfiguration

Configures collection of events generated from [EventSource Class](#).

CHILD ELEMENTS	DESCRIPTION
DefaultEvents	<p>Optional attribute:</p> <p>eventDestination - The name of the table to store the events in</p>
Event	<p>Required attribute:</p> <p>id - The id of the event.</p> <p>Optional attribute:</p> <p>eventDestination - The name of the table to store the events in</p>

EtwManifestProviderConfiguration Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - EtwProviders - EtwManifestProviderConfiguration

CHILD ELEMENTS	DESCRIPTION
DefaultEvents	Optional attribute: eventDestination - The name of the table to store the events in
Event	Required attribute: id - The id of the event. Optional attribute: eventDestination - The name of the table to store the events in

Metrics Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - Metrics

Enables you to generate a performance counter table that is optimized for fast queries. Each performance counter that is defined in the **PerformanceCounters** element is stored in the Metrics table in addition to the Performance Counter table.

The **resourceId** attribute is required. The resource ID of the Virtual Machine you are deploying Azure Diagnostics to. Get the **resourceID** from the [Azure portal](#). Select **Browse** -> **Resource Groups** -> . Click the **Properties** tile and copy the value from the **ID** field.

CHILD ELEMENTS	DESCRIPTION
MetricAggregation	Required attribute: scheduledTransferPeriod - The interval between scheduled transfers to storage rounded up to the nearest minute. The value is an XML "Duration Data Type."

PerformanceCounters Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - PerformanceCounters

Enables the collection of performance counters.

Optional attribute:

Optional **scheduledTransferPeriod** attribute. See explanation earlier.

CHILD ELEMENT	DESCRIPTION
PerformanceCounterConfiguration	The following attributes are required: - counterSpecifier - The name of the performance counter. For example, <code>\Processor(_Total)\% Processor Time</code> . To get a list of performance counters on your host, run the command <code>typeperf</code> . - sampleRate - How often the counter should be sampled. Optional attribute: unit - The unit of measure of the counter.

WindowsEventLog Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - WindowsEventLog

Enables the collection of Windows Event Logs.

Optional **scheduledTransferPeriod** attribute. See explanation earlier.

CHILD ELEMENT	DESCRIPTION
DataSource	<p>The Windows Event logs to collect. Required attribute:</p> <p>name - The XPath query describing the windows events to be collected. For example:</p> <pre>Application!*[System[(Level <=3)]], System!*[System[(Level <=3)]], System!*[System[Provider[@Name='Microsoft Antimalware']]], Security!*[System[(Level <= 3)]]</pre> <p>To collect all events, specify "*"</p>

Logs Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - Logs

Present in version 1.0 and 1.1. Missing in 1.2. Added back in 1.3.

Defines the buffer configuration for basic Azure logs.

ATTRIBUTE	TYPE	DESCRIPTION
bufferQuotaInMB	unsignedInt	<p>Optional. Specifies the maximum amount of file system storage that is available for the specified data.</p> <p>The default is 0.</p>
scheduledTransferLogLevelFilter	string	<p>Optional. Specifies the minimum severity level for log entries that are transferred. The default value is Undefined, which transfers all logs. Other possible values (in order of most to least information) are Verbose, Information, Warning, Error, and Critical.</p>
scheduledTransferPeriod	duration	<p>Optional. Specifies the interval between scheduled transfers of data, rounded up to the nearest minute.</p> <p>The default is PT0S.</p>
sinks Added in 1.5	string	<p>Optional. Points to a sink location to also send diagnostic data. For example, Application Insights.</p>

DockerSources

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - DiagnosticMonitorConfiguration - DockerSources

Added in 1.9.

ELEMENT NAME	DESCRIPTION
Stats	Tells the system to collect stats for Docker containers

SinksConfig Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - SinksConfig

A list of locations to send diagnostics data to and the configuration associated with those locations.

ELEMENT NAME	DESCRIPTION
Sink	See description elsewhere on this page.

Sink Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - SinksConfig - Sink

Added in version 1.5.

Defines locations to send diagnostic data to. For example, the Application Insights service.

ATTRIBUTE	TYPE	DESCRIPTION
name	string	A string identifying the sinkname.
ELEMENT	TYPE	DESCRIPTION
Application Insights	string	Used only when sending data to Application Insights. Contain the Instrumentation Key for an active Application Insights account that you have access to.
Channels	string	One for each additional filtering that stream that you

Channels Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - SinksConfig - Sink - Channels

Added in version 1.5.

Defines filters for streams of log data passing through a sink.

ELEMENT	TYPE	DESCRIPTION
Channel	string	See description elsewhere on this page.

Channel Element

Tree: Root - DiagnosticsConfiguration - PublicConfig - WadCFG - SinksConfig - Sink - Channels - Channel

Added in version 1.5.

Defines locations to send diagnostic data to. For example, the Application Insights service.

ATTRIBUTES	TYPE	DESCRIPTION
logLevel	string	Specifies the minimum severity level for log entries that are transferred. The default value is Undefined , which transfers all logs. Other possible values (in order of most to least information) are Verbose , Information , Warning , Error , and Critical .
name	string	A unique name of the channel to refer to

PrivateConfig Element

Tree: Root - DiagnosticsConfiguration - PrivateConfig

Added in version 1.3.

Optional

Stores the private details of the storage account (name, key, and endpoint). This information is sent to the virtual machine, but cannot be retrieved from it.

CHILD ELEMENTS	DESCRIPTION
StorageAccount	<p>The storage account to use. The following attributes are required</p> <ul style="list-style-type: none">- name - The name of the storage account.- key - The key to the storage account.- endpoint - The endpoint to access the storage account. <p>-sasToken (added 1.8.1)- You can specify an SAS token instead of a storage account key in the private config. If provided, the storage account key is ignored.</p> <p>Requirements for the SAS Token:</p> <ul style="list-style-type: none">- Supports account SAS token only- <i>b, t</i> service types are required.- <i>a, c, u, w</i> permissions are required.- <i>c, o</i> resource types are required.- Supports the HTTPS protocol only- Start and expiry time must be valid.

.IsEnabled Element

Tree: Root - DiagnosticsConfiguration - IsEnabled

Boolean. Use `true` to enable the diagnostics or `false` to disable the diagnostics.

Azure Monitor Cross-platform CLI 1.0 quick start samples

7/18/2017 • 3 min to read • [Edit Online](#)

This article shows you sample command-line interface (CLI) commands to help you access Azure Monitor features. Azure Monitor allows you to AutoScale Cloud Services, Virtual Machines, and Web Apps and to send alert notifications or call web URLs based on values of configured telemetry data.

NOTE

Azure Monitor is the new name for what was called "Azure Insights" until Sept 25th, 2016. However, the namespaces and thus the commands below still contain the "insights".

Prerequisites

If you haven't already installed the Azure CLI, see [Install the Azure CLI](#). If you're unfamiliar with Azure CLI, you can read more about it at [Use the Azure CLI for Mac, Linux, and Windows with Azure Resource Manager](#).

In Windows, install npm from the [Node.js website](#). After you complete the installation, using CMD.exe with Run As Administrator privileges, execute the following from the folder where npm is installed:

```
npm install azure-cli --global
```

Next, navigate to any folder/location you want and type at the command-line:

```
azure help
```

Log in to Azure

The first step is to login to your Azure account.

```
azure login
```

After running this command, you have to sign in via the instructions on the screen. Afterward, you see your Account, TenantId, and default Subscription Id. All commands work in the context of your default subscription.

To list the details of your current subscription, use the following command.

```
azure account show
```

To change working context to a different subscription, use the following command.

```
azure account set "subscription ID or subscription name"
```

To use Azure Resource Manager and Azure Monitor commands, you need to be in Azure Resource Manager mode.

```
azure config mode arm
```

To view a list of all supported Azure Monitor commands, perform the following.

```
azure insights
```

View activity log for a subscription

To view a list of activity log events, perform the following.

```
azure insights logs list [options]
```

Try the following to view all available options.

```
azure insights logs list -help
```

Here is an example to list logs by a resourceGroup

```
azure insights logs list --resourceGroup "myrg1"
```

Example to list logs by caller

```
azure insights logs list --caller "myname@company.com"
```

Example to list logs by caller on a resource type, within a start and end date

```
azure insights logs list --resourceProvider "Microsoft.Web" --caller "myname@company.com" --startTime 2016-03-08T00:00:00Z --endTime 2016-03-16T00:00:00Z
```

Work with alerts

You can use the information in the section to work with alerts.

Get alert rules in a resource group

```
azure insights alerts rule list abhingrgtest123
azure insights alerts rule list abhingrgtest123 --ruleName andy0323
```

Create a metric alert rule

```
azure insights alerts actions email create --customEmails foo@microsoft.com
azure insights alerts actions webhook create https://someuri.com
azure insights alerts rule metric set andy0323 eastus abhingrgtest123 PT5M GreaterThan 2
/subscriptions/df602c9c-7aa0-407d-a6fb-eb20c8bd1192/resourceGroups/Default-Web-EastUS/providers/Microsoft.Web/serverfarms/Default1 BytesReceived Total
```

Create a log alert rule

```
azure insights alerts rule log set ruleName eastus resourceName someOperationName
```

Create webtest alert rule

```
azure insights alerts rule webtest set leowebtestr1-webtestr1 eastus Default-Web-WestUS PT5M 1 GSMT_AvRaw  
/subscriptions/b67f7fec-69fc-4974-9099-a26bd6ffeda3/resourcegroups/Default-Web-  
WestUS/providers/microsoft.insights/webtests/leowebtestr1-webtestr1
```

Delete an alert rule

```
azure insights alerts rule delete abhingrgtest123 andy0323
```

Log profiles

Use the information in this section to work with log profiles.

Get a log profile

```
azure insights logprofile list  
azure insights logprofile get -n default
```

Add a log profile without retention

```
azure insights logprofile add --name default --storageId /subscriptions/1a66ce04-b633-4a0b-b2bc-  
a912ec8986a6/resourceGroups/insights-  
integration/providers/Microsoft.Storage/storageAccounts/insightsintegration7777 --locations  
global,westus,eastus,northeurope,westeurope,eastasia,southeastasia,japaneast,japanwest,northcentralus,southce-  
ntralus,eastus2,centralus,australiaeast,australiasoutheast,brazilsouth,centralindia,southindia,westindia
```

Remove a log profile

```
azure insights logprofile delete --name default
```

Add a log profile with retention

```
azure insights logprofile add --name default --storageId /subscriptions/1a66ce04-b633-4a0b-b2bc-  
a912ec8986a6/resourceGroups/insights-  
integration/providers/Microsoft.Storage/storageAccounts/insightsintegration7777 --locations  
global,westus,eastus,northeurope,westeurope,eastasia,southeastasia,japaneast,japanwest,northcentralus,southce-  
ntralus,eastus2,centralus,australiaeast,australiasoutheast,brazilsouth,centralindia,southindia,westindia --  
retentionInDays 90
```

Add a log profile with retention and EventHub

```
azure insights logprofile add --name default --storageId /subscriptions/1a66ce04-b633-4a0b-b2bc-  
a912ec8986a6/resourceGroups/insights-  
integration/providers/Microsoft.Storage/storageAccounts/insightsintegration7777 --serviceBusRuleId  
/subscriptions/1a66ce04-b633-4a0b-b2bc-a912ec8986a6/resourceGroups/Default-ServiceBus-  
EastUS/providers/Microsoft.ServiceBus/namespaces/testshoeboxeastus/authorizationrules/RootManageSharedAccessK  
ey --locations  
global,westus,eastus,northeurope,westeurope,eastasia,southeastasia,japaneast,japanwest,northcentralus,southce-  
ntralus,eastus2,centralus,australiaeast,australiasoutheast,brazilsouth,centralindia,southindia,westindia --  
retentionInDays 90
```

Diagnostics

Use the information in this section to work with diagnostic settings.

Get a diagnostic setting

```
azure insights diagnostic get --resourceId /subscriptions/df602c9c-7aa0-407d-a6fb-eb20c8bd1192/resourceGroups/andyrg/providers/Microsoft.Logic/workflows/andy0315logicapp
```

Disable a diagnostic setting

```
azure insights diagnostic set --resourceId /subscriptions/df602c9c-7aa0-407d-a6fb-eb20c8bd1192/resourceGroups/andyrg/providers/Microsoft.Logic/workflows/andy0315logicapp --storageId /subscriptions/df602c9c-7aa0-407d-a6fb-eb20c8bd1192/resourceGroups/Default-Storage-WestUS/providers/Microsoft.Storage/storageAccounts/shibanitesting --enabled false
```

Enable a diagnostic setting without retention

```
azure insights diagnostic set --resourceId /subscriptions/df602c9c-7aa0-407d-a6fb-eb20c8bd1192/resourceGroups/andyrg/providers/Microsoft.Logic/workflows/andy0315logicapp --storageId /subscriptions/df602c9c-7aa0-407d-a6fb-eb20c8bd1192/resourceGroups/Default-Storage-WestUS/providers/Microsoft.Storage/storageAccounts/shibanitesting --enabled true
```

Autoscale

Use the information in this section to work with autoscale settings. You need to modify these examples.

Get autoscale settings for a resource group

```
azure insights autoscale setting list montest2
```

Get autoscale settings by name in a resource group

```
azure insights autoscale setting list montest2 -n setting2
```

Set autoscale settings

```
azure insights autoscale setting set montest2 -n setting2 --settingSpec
```

Azure Monitor PowerShell quick start samples

7/18/2017 • 9 min to read • [Edit Online](#)

This article shows you sample PowerShell commands to help you access Azure Monitor features. Azure Monitor allows you to AutoScale Cloud Services, Virtual Machines, and Web Apps and to send alert notifications or call web URLs based on values of configured telemetry data.

NOTE

Azure Monitor is the new name for what was called "Azure Insights" until Sept 25th, 2016. However, the namespaces and thus the commands below still contain the "insights".

Set up PowerShell

If you haven't already, set up PowerShell to run on your computer. For more information, see [How to Install and Configure PowerShell](#).

Examples in this article

The examples in the article illustrate how you can use Azure Monitor cmdlets. You can also review the entire list of Azure Monitor PowerShell cmdlets at [Azure Monitor \(Insights\) Cmdlets](#).

Sign in and use subscriptions

First, log into your Azure subscription.

```
Login-AzureRmAccount
```

This requires you to sign in. Once you do, your Account, TenantId and default Subscription Id are displayed. All the Azure cmdlets work in the context of your default subscription. To view the list of subscriptions you have access to, use the following command.

```
Get-AzureRmSubscription
```

To change your working context to a different subscription, use the following command.

```
Set-AzureRmContext -SubscriptionId <subscriptionid>
```

Retrieve Activity log for a subscription

Use the `Get-AzureRmLog` cmdlet. Below are some common examples.

Get log entries from this time/date to present:

```
Get-AzureRmLog -StartTime 2016-03-01T10:30
```

Get log entries between a time/date range:

```
Get-AzureRmLog -StartTime 2015-01-01T10:30 -EndTime 2015-01-01T11:30
```

Get log entries from a specific resource group:

```
Get-AzureRmLog -ResourceGroup 'myrg1'
```

Get log entries from a specific resource provider between a time/date range:

```
Get-AzureRmLog -ResourceProvider 'Microsoft.Web' -StartTime 2015-01-01T10:30 -EndTime 2015-01-01T11:30
```

Get all log entries with a specific caller:

```
Get-AzureRmLog -Caller 'myname@company.com'
```

The following command retrieves the last 1000 events from the activity log:

```
Get-AzureRmLog -MaxEvents 1000
```

`Get-AzureRmLog` supports many other parameters. See the [Get-AzureRmLog](#) reference for more information.

NOTE

`Get-AzureRmLog` only provides 15 days of history. Using the **-MaxEvents** parameter allows you to query the last N events, beyond 15 days. To access events older than 15 days, use the REST API or SDK (C# sample using the SDK). If you do not include **StartTime**, then the default value is **EndTime** minus one hour. If you do not include **EndTime**, then the default value is current time. All times are in UTC.

Retrieve alerts history

To view all alert events, you can query the Azure Resource Manager logs using the following examples.

```
Get-AzureRmLog -Caller "Microsoft.Insights/alertRules" -DetailedOutput -StartTime 2015-03-01
```

To view the history for a specific alert rule, you can use the `Get-AzureRmAlertHistory` cmdlet, passing in the resource ID of the alert rule.

```
Get-AzureRmAlertHistory -ResourceId  
/subscriptions/s1/resourceGroups/rg1/providers/microsoft.insights/alertrules/myalert -StartTime 2016-03-1 -  
Status Activated
```

The `Get-AzureRmAlertHistory` cmdlet supports various parameters. More information, see [Get-AlertHistory](#).

Retrieve information on alert rules

All of the following commands act on a Resource Group named "montest".

View all the properties of the alert rule:

```
Get-AzureRmAlertRule -Name simpletestCPU -ResourceGroup montest -DetailedOutput
```

Retrieve all alerts on a resource group:

```
Get-AzureRmAlertRule -ResourceGroup montest
```

Retrieve all alert rules set for a target resource. For example, all alert rules set on a VM.

```
Get-AzureRmAlertRule -ResourceGroup montest -TargetResourceId  
/subscriptions/s1/resourceGroups/montest/providers/Microsoft.Compute/virtualMachines/testconfig
```

`Get-AzureRmAlertRule` supports other parameters. See [Get-AlertRule](#) for more information.

Create alert rules

You can use the `Add-AlertRule` cmdlet to create, update or disable an alert rule.

You can create email and webhook properties using `New-AzureRmAlertRuleEmail` and `New-AzureRmAlertRuleWebhook`, respectively. In the Alert rule cmdlet, assign these as actions to the **Actions** property of the Alert Rule.

The next section contains a sample that shows you how to create an Alert Rule with various parameters.

Alert rule on a metric

The following table describes the parameters and values used to create an alert using a metric.

PARAMETER	VALUE
Name	simpletestdiskwrite
Location of this alert rule	East US
ResourceGroup	montest
TargetResourceId	/subscriptions/s1/resourceGroups/montest/providers/Microsoft.Compute/virtualMachines/testconfig
MetricName of the alert that is created	\PhysicalDisk(_Total)\Disk Writes/sec. See the <code>Get-MetricDefinitions</code> cmdlet below about how to retrieve the exact metric names
operator	GreaterThan
Threshold value (count/sec in for this metric)	1
WindowSize (hh:mm:ss format)	00:05:00
aggregator (statistic of the metric, which uses Average count, in this case)	Average
custom emails (string array)	'foo@example.com','bar@example.com'
send email to owners, contributors and readers	-SendToServiceOwners

Create an Email action

```
$actionEmail = New-AzureRmAlertRuleEmail -CustomEmail myname@company.com
```

Create a Webhook action

```
$actionWebhook = New-AzureRmAlertRuleWebhook -ServiceUri https://example.com?token=mytoken
```

Create the alert rule on the CPU% metric on a classic VM

```
Add-AzureRmMetricAlertRule -Name vmcpu_gt_1 -Location "East US" -ResourceGroup myrg1 -TargetResourceId /subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.ClassicCompute/virtualMachines/my_vm1 -MetricName "Percentage CPU" -Operator GreaterThan -Threshold 1 -WindowSize 00:05:00 -TimeAggregationOperator Average -Actions $actionEmail, $actionWebhook -Description "alert on CPU > 1%"
```

Retrieve the alert rule

```
Get-AzureRmAlertRule -Name vmcpu_gt_1 -ResourceGroup myrg1 -DetailedOutput
```

The Add alert cmdlet also updates the rule if an alert rule already exists for the given properties. To disable an alert rule, include the parameter **-DisableRule**.

Alert on activity log event

NOTE

This feature is in preview and will be removed at some point in the future (it is being replaced).

In this scenario, you'll send email when a website is successfully started in my subscription in resource group *abhingrgtest123*.

Setup an email rule

```
$actionEmail = New-AzureRmAlertRuleEmail -CustomEmail myname@company.com
```

Setup a webhook rule

```
$actionWebhook = New-AzureRmAlertRuleWebhook -ServiceUri https://example.com?token=mytoken
```

Create the rule on the event

```
Add-AzureRmLogAlertRule -Name superalert1 -Location "East US" -ResourceGroup myrg1 -OperationName microsoft.web/sites/start/action -Status Succeeded -TargetResourceGroup abhingrgtest123 -Actions $actionEmail, $actionWebhook
```

Retrieve the alert rule

```
Get-AzureRmAlertRule -Name superalert1 -ResourceGroup myrg1 -DetailedOutput
```

The `Add-AlertRule` cmdlet allows various other parameters. More information, see [Add-AlertRule](#).

Get a list of available metrics for alerts

You can use the `Get-AzureRmMetricDefinition` cmdlet to view the list of all metrics for a specific resource.

```
Get-AzureRmMetricDefinition -ResourceId <resource_id>
```

The following example generates a table with the metric Name and the Unit for it.

```
Get-AzureRmMetricDefinition -ResourceId <resource_id> | Format-Table -Property Name,Unit
```

A full list of available options for `Get-AzureRmMetricDefinition` is available at [Get-MetricDefinitions](#).

Create and manage AutoScale settings

A resource, such as a Web app, VM, Cloud Service or VM Scale Set can have only one autoscale setting configured for it. However, each autoscale setting can have multiple profiles. For example, one for a performance-based scale profile and a second one for a schedule based profile. Each profile can have multiple rules configured on it. For more information about Autoscale, see [How to Autoscale an Application](#).

Here are the steps we will use:

1. Create rule(s).
2. Create profile(s) mapping the rules that you created previously to the profiles.
3. Optional: Create notifications for autoscale by configuring webhook and email properties.
4. Create an autoscale setting with a name on the target resource by mapping the profiles and notifications that you created in the previous steps.

The following examples show you how you can create an Autoscale setting for a VM scale set for a Windows operating system based by using the CPU utilization metric.

First, create a rule to scale-out, with an instance count increase .

```
$rule1 = New-AzureRmAutoscaleRule -MetricName "Percentage CPU" -MetricResourceId /subscriptions/s1/resourceGroups/big2/providers/Microsoft.Compute/virtualMachineScaleSets/big2 -Operator GreaterThan -MetricStatistic Average -Threshold 60 -TimeGrain 00:01:00 -TimeWindow 00:10:00 -ScaleActionCooldown 00:10:00 -ScaleActionDirection Increase -ScaleActionValue 1
```

Next, create a rule to scale-in, with an instance count decrease.

```
$rule2 = New-AzureRmAutoscaleRule -MetricName "Percentage CPU" -MetricResourceId /subscriptions/s1/resourceGroups/big2/providers/Microsoft.Compute/virtualMachineScaleSets/big2 -Operator GreaterThan -MetricStatistic Average -Threshold 30 -TimeGrain 00:01:00 -TimeWindow 00:10:00 -ScaleActionCooldown 00:10:00 -ScaleActionDirection Decrease -ScaleActionValue 1
```

Then, create a profile for the rules.

```
$profile1 = New-AzureRmAutoscaleProfile -DefaultCapacity 2 -MaximumCapacity 10 -MinimumCapacity 2 -Rules $rule1,$rule2 -Name "My_Profile"
```

Create a webhook property.

```
$webhook_scale = New-AzureRmAutoscaleWebhook -ServiceUri "https://example.com?mytoken=mytokenvalue"
```

Create the notification property for the autoscale setting, including email and the webhook that you created previously.

```
$notification1= New-AzureRmAutoscaleNotification -CustomEmails ashwink@microsoft.com -  
SendEmailToSubscriptionAdministrators SendEmailToSubscriptionCoAdministrators -Webhooks $webhook_scale
```

Finally, create the autoscale setting to add the profile that you created above.

```
Add-AzureRmAutoscaleSetting -Location "East US" -Name "MyScaleVMSSSetting" -ResourceGroup big2 -  
TargetResourceId  
/subscriptions/s1/resourceGroups/big2/providers/Microsoft.Compute/virtualMachineScaleSets/big2 -  
AutoscaleProfiles $profile1 -Notifications $notification1
```

For more information about managing Autoscale settings, see [Get-AutoscaleSetting](#).

Autoscale history

The following example shows you how you can view recent autoscale and alert events. Use the activity log search to view the autoscale history.

```
Get-AzureRmLog -Caller "Microsoft.Insights/autoscaleSettings" -DetailedOutput -StartTime 2015-03-01
```

You can use the `Get-AzureRmAutoScaleHistory` cmdlet to retrieve AutoScale history.

```
Get-AzureRmAutoScaleHistory -ResourceId  
/subscriptions/s1/resourceGroups/myrg1/providers/microsoft.insights/autoscalesettings/myScaleSetting -  
StartTime 2016-03-15 -DetailedOutput
```

For more information, see [Get-AutoscaleHistory](#).

View details for an autoscale setting

You can use the `Get-AutoscaleSetting` cmdlet to retrieve more information about the autoscale setting.

The following example shows details about all autoscale settings in the resource group 'myrg1'.

```
Get-AzureRmAutoscaleSetting -ResourceGroup myrg1 -DetailedOutput
```

The following example shows details about all autoscale settings in the resource group 'myrg1' and specifically the autoscale setting named 'MyScaleVMSSSetting'.

```
Get-AzureRmAutoscaleSetting -ResourceGroup myrg1 -Name MyScaleVMSSSetting -DetailedOutput
```

Remove an autoscale setting

You can use the `Remove-AutoscaleSetting` cmdlet to delete an autoscale setting.

```
Remove-AzureRmAutoscaleSetting -ResourceGroup myrg1 -Name MyScaleVMSSSetting
```

Manage log profiles for activity log

You can create a *log profile* and export data from your activity log to a storage account and you can configure data retention for it. Optionally, you can also stream the data to your Event Hub. Note that this feature is currently in Preview and you can only create one log profile per subscription. You can use the following cmdlets with your current subscription to create and manage log profiles. You can also choose a particular subscription. Although

PowerShell defaults to the current subscription, you can always change that using `Set-AzureRmContext`. You can configure activity log to route data to any storage account or Event Hub within that subscription. Data is written as blob files in JSON format.

Get a log profile

To fetch your existing log profiles, use the `Get-AzureRmLogProfile` cmdlet.

Add a log profile without data retention

```
Add-AzureRmLogProfile -Name my_log_profile_s1 -StorageAccountId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/my_storage -Locations  
global,westus,eastus,northeurope,westeurope,eastasia,southeastasia,japaneast,japanwest,northcentralus,southce  
ntralus,eastus2,centralus,australiaeast,australiasoutheast,brazilsouth,centralindia,southindia,westindia
```

Remove a log profile

```
Remove-AzureRmLogProfile -name my_log_profile_s1
```

Add a log profile with data retention

You can specify the **-RetentionInDays** property with the number of days, as a positive integer, where the data is retained.

```
Add-AzureRmLogProfile -Name my_log_profile_s1 -StorageAccountId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/my_storage -Locations  
global,westus,eastus,northeurope,westeurope,eastasia,southeastasia,japaneast,japanwest,northcentralus,southce  
ntralus,eastus2,centralus,australiaeast,australiasoutheast,brazilsouth,centralindia,southindia,westindia -  
RetentionInDays 90
```

Add log profile with retention and EventHub

In addition to routing your data to storage account, you can also stream it to an Event Hub. Note that in this preview release and the storage account configuration is mandatory but Event Hub configuration is optional.

```
Add-AzureRmLogProfile -Name my_log_profile_s1 -StorageAccountId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/my_storage -  
serviceBusRuleId /subscriptions/s1/resourceGroups/Default-ServiceBus-  
EastUS/providers/Microsoft.ServiceBus/namespaces/mytestSB/authorizationrules/RootManageSharedAccessKey -  
Locations  
global,westus,eastus,northeurope,westeurope,eastasia,southeastasia,japaneast,japanwest,northcentralus,southce  
ntralus,eastus2,centralus,australiaeast,australiasoutheast,brazilsouth,centralindia,southindia,westindia -  
RetentionInDays 90
```

Configure diagnostics logs

Many Azure services provide additional logs and telemetry that can be configured to save data in your Azure Storage account, send to Event Hubs, and/or sent to an OMS Log Analytics workspace. That operation can only be performed at a resource level and the storage account or event hub should be present in the same region as the target resource where the diagnostics setting is configured.

Get diagnostic setting

```
Get-AzureRmDiagnosticSetting -ResourceId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Logic/workflows/andy0315logicapp
```

Disable diagnostic setting

```
Set-AzureRmDiagnosticSetting -ResourceId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Logic/workflows/andy0315logicapp -StorageAccountId  
/subscriptions/s1/resourceGroups/Default-Storage-  
WestUS/providers/Microsoft.Storage/storageAccounts/mystorageaccount -Enable $false
```

Enable diagnostic setting without retention

```
Set-AzureRmDiagnosticSetting -ResourceId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Logic/workflows/andy0315logicapp -StorageAccountId  
/subscriptions/s1/resourceGroups/Default-Storage-  
WestUS/providers/Microsoft.Storage/storageAccounts/mystorageaccount -Enable $true
```

Enable diagnostic setting with retention

```
Set-AzureRmDiagnosticSetting -ResourceId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Logic/workflows/andy0315logicapp -StorageAccountId  
/subscriptions/s1/resourceGroups/Default-Storage-  
WestUS/providers/Microsoft.Storage/storageAccounts/mystorageaccount -Enable $true -RetentionEnabled $true -  
RetentionInDays 90
```

Enable diagnostic setting with retention for a specific log category

```
Set-AzureRmDiagnosticSetting -ResourceId /subscriptions/s1/resourceGroups/insights-  
integration/providers/Microsoft.Network/networkSecurityGroups/viruela1 -StorageAccountId  
/subscriptions/s1/resourceGroups/myrg1/providers/Microsoft.Storage/storageAccounts/sakteststorage -Categories  
NetworkSecurityGroupEvent -Enable $true -RetentionEnabled $true -RetentionInDays 90
```

Enable diagnostic setting for Event Hubs

```
Set-AzureRmDiagnosticSetting -ResourceId /subscriptions/s1/resourceGroups/insights-  
integration/providers/Microsoft.Network/networkSecurityGroups/viruela1 -serviceBusRuleId  
/subscriptions/s1/resourceGroups/Default-ServiceBus-  
EastUS/providers/Microsoft.ServiceBus/namespaces/mytestSB/authorizationrules/RootManageSharedAccessKey -  
Enable $true
```

Enable diagnostic setting for OMS

```
Set-AzureRmDiagnosticSetting -ResourceId /subscriptions/s1/resourceGroups/insights-  
integration/providers/Microsoft.Network/networkSecurityGroups/viruela1 -WorkspaceId 76d785fd-d1ce-4f50-8ca3-  
858fc819ca0f -Enabled $true
```