Caution

The SignalFx Instrumentation for .NET reached End of Support on February 21, 2025. The library has been archived and is no longer maintained.

New customers instrumenting the .NET ecosystem should use the Splunk Distribution of OpenTelemetry .NET. Existing customers should consider migrating to Splunk Distribution of OpenTelemetry .NET which offers similar capabilities. To learn how to migrate, see Migrate from the SignalFx .NET Instrumentation.

Metrics collected by the SignalFx Instrumentation for .NET

The SignalFx Instrumentation for .NET can collect runtime and trace metrics. To learn about the different metric types, see Metric types.

Activate metrics collection

To activate the collection of .NET runtime and trace metrics, see Metrics settings.

Note

NetRuntime metrics are always collected if memory profiling is activated.

Default metric dimensions

The following dimensions are automatically added to all metrics exported by the agent:

Dimension	Description
`deployment.environment`	Deployment environment, if present.

`service.name`	Name of the service.
`process.pid`	The .NET process identifier (PID).
`container.id`	Identifier of the container, if applicable.
`host.name`	Name of the host.
`telemetry.sdk.name`	Name of the SDK, set to `signalfx-dotnet-tracing`.
`telemetry.sdk.language`	Language of the SDK, set to `dotnet`.
`telemetry.sdk.version`	Version of the SDK.
`splunk.distro.version`	Version of the Splunk distribution.

.NET runtime metrics

The SignalFx Instrumentation for .NET can collect the following runtime metrics:

Metric	Type	Description
`process.runtime.dotnet.exceptions.count`	Gauge	Count of exceptions since the previous observation.
`process.runtime.dotnet.gc.collections.count`	Cumulative counter	Number of garbage collections since the process started.
`process.runtime.dotnet.gc.heap.size`	Gauge	Heap size, as observed during the last garbage collection.

`process.runtime.dotnet.gc.objects.size`	Gauge	Count of bytes currently in use by live objects in the GC heap.
`process.runtime.dotnet.gc.allocations.size`	Cumulative counter	Count of bytes allocated on the managed GC heap since the process started. Only available for .NET Core.
`process.runtime.dotnet.gc.committed_memory.size`	Gauge	Amount of committed virtual memory for the managed GC heap, as observed during the last garbage collection. Only available for .NET 6 and higher.
`process.runtime.dotnet.gc.pause.time`	Counter	Number of milliseconds spent in GC pause. Only available for .NET Core.
`process.runtime.dotnet.monitor.lock_contention.count`	Cumulative counter	Contentions count when trying to acquire a monitor lock since the process started.
`process.runtime.dotnet.thread_pool.threads.count`	Gauge	Number of thread pool threads, as observed during the last measurement. Only available for .NET Core.

Process metrics

The SignalFx Instrumentation for .NET can collect the following process metrics:

Metric	**Type**	**Description**
`process.memory.usage`	Gauge	The amount of physical memory allocated for this process.

`process.memory.virtual`	Gauge	The amount of committed virtual memory for this process.
`process.cpu.time`	CumulativeCounter	Total CPU seconds broken down by different states, such as user and system.
`process.cpu.utilization` (deprecated)	Gauge	Difference in `process.cpu.time` since the last measurement, divided by the elapsed time and number of CPUs available to the process.
`process.threads`	Gauge	Process threads count.

ASP.NET Core metrics

The SignalFx Instrumentation for .NET can collect the following ASP.NET Core metrics:

Metric	**Type**	**Description**
`signalfx.dotnet.aspnetcore.connections.current`	Gauge	Number of active HTTP connections to the web server. Only available for .NET Core.
`signalfx.dotnet.aspnetcore.connections.queue_length`	Gauge	Length of the HTTP connection queue. Only available for .NET Core.
`signalfx.dotnet.aspnetcore.connections.total`	Gauge	Number of HTTP connections to the web server. Only available for .NET Core.
`signalfx.dotnet.aspnetcore.requests.current`	Gauge	Number of HTTP requests that have started, but haven't stopped yet. Only available for .NET Core.

`signalfx.dotnet.aspnetcore.requests.failed`	Gauge	Number of failed HTTP requests received by the server. Only available for .NET Core.
`signalfx.dotnet.aspnetcore.requests.queue_length`	Gauge	Length of the HTTP request queue.
`signalfx.dotnet.aspnetcore.requests.total`	Gauge	Number of HTTP requests received by the server. Only available for .NET Core.

Additional permissions for IIS

The .NET Framework collects metrics using performance counters. To let service accounts and IIS application pool accounts access counter data, add them to the `Performance Monitoring Users` group.

IIS application pools use special accounts that don't appear in the list of users. To add IIS application pool accounts to the `Performance Monitoring Users` group, search for `IIS APPPOOL\<name-of-the-pool>`. For example, the user for the `DefaultAppPool` pool is `IIS APPPOOL\DefaultAppPool`.

The following example shows how to add an IIS application pool account to the `Performance Monitoring Users` group from a command prompt with Administrator permissions:

```
net localgroup "Performance Monitor Users" "IIS APPPOOL\DefaultAppPool" /add
```

Trace metrics

The SignalFx Instrumentation for .NET can collect the following trace metrics:

Metric	Type	Description
`signalfx.tracer.queue.enqueued_traces`	Counter	Number of traces pushed into the queue.

`signalfx.tracer.queue.dequeued_traces`	Counter	Number of traces pulled from the queue for flushing.
`signalfx.tracer.queue.enqueued_spans`	Counter	Number of spans pushed into the queue.
`signalfx.tracer.queue.dequeued_spans`	Counter	Number of spans pulled from the queue for flushing.
`signalfx.tracer.queue.dropped_traces`	Counter	Number of traces dropped due to a full queue.
`signalfx.tracer.queue.dropped_spans`	Counter	Number of spans dropped due to a full queue.
`signalfx.tracer.heartbeat`	Gauge	Number of tracers.