Controller System Requirements

On this page:

Related pages:

- Prepare the Controller Host

This page describes hardware and software requirements for the Controller host to help you prepare for your AppDynamics deployment.

About Controller Sizing Information

Every deployment is unique. Factors such as the nature of the application, workload, and the AppDynamics configuration can all affect the resources required for your specific scenario. Be sure to test the performance of your system in a staging environment, so that you can fully understand your requirements before deploying AppDynamics to its live operating environment.

Before installation, it's usually easiest to estimate your deployment size based on number of nodes. For Java, for example, a node corresponds to a JVM. However, the best indicator of the actual workload on your Controller is provided by the metric ingestion rate.

After initial installation, you should verify your Controller sizing using the metric upload rate. You then need to continue to monitor the Controller for changing workload brought about by changes in the monitored application, its usage patterns, or in the AppDynamics configuration.

General Hardware Requirements

The following general requirements that apply to the machine on which you install the Controller:

- The Controller should run on a dedicated machine. A production Controller *must* run on a dedicated machine. The requirements here assume that no other major processes are running on the machine where the Controller is installed, including no other Controllers.
- The Controller is not supported on machines that use Power Architecture processors, including PowerPC processors.
- The Controller host must have around 200 MB of free space available in the system temporary directory.
- Disk I/O is a key element to Controller performance, particularly low latency. See Disk I/O Requirements for more information.

Controller Sizing

The following table shows Controller installation profiles by metric ingestion rate and node count. As previously noted, the actual metrics generated by a node can vary greatly depending on the nature of the application on the node and the AppDynamics configuration. Be sure to validate your sizing against the metric ingestion rate before deploying to production.

Installer Performance Profile	Maximum Metric Count /Minute	Estimated Max Number of Nodes	Minimum Hardware Profile			
			CPU	RAM	Disk Size	Notes
Demo	10 K	5	2 Cores	3 GB	50 GB	Suitable for a laptop installation.

Small	50 K	50	4 Cores	16 GB	400 GB	
Medium	1 M	1500	8 Cores (physical host) /16 vCPUs (virtual host)	128 GB	5 TB	On physical machines, this performance profile assumes the use of 5 SAS SSDs using 2 NVMe cards (minimum 800GB each). On physical and virtual machines, if using SAN we require 10 Gb/s FCoE and an SSD-based SAN target. See Prepare Virtual Machines for the Controller and Disk I/O Requirements. On AWS machines, the recommended Controller EC2 Instance Size is i3.4xlarge. There is no Aurora DB support for this performance profile.
Large	5 M	10,000	28 Cores	512 GB	20 TB	Linux only. Not supported on virtual machines. Storage requirements are 10 SAS SSDs using 2 NVMe cards (minimum 1.6TB each). Be sure to observe the Disk I/O Requirements. Also see Tuning for Large Profile Controllers. On AWS machines, the recommended Controller EC2 Instance Size is r4.8xlarge. The recommended Aurora RDS Instance Size is db.r4.16xlarge.
Extra Large	For deployments that exceed the Large profile size, contact AppDynamics Professional Services.					

The numbers here have been last updated on May 25, 2018.

Additional Sizing Considerations

Note the following additional requirements:

- Large and extra large profile installations are not supported on virtual machines or systems that use network attached storage.
- The RAM recommendations leave room for operating system processes. However, the recommendations assume that no other memory intensive applications are running on the same machine. While the Enterprise Console can run on the same host as the Controller in small or demo profile Controllers, it is not recommended for medium and larger profiles or for high availability deployments.
- Disk sizing shown in the sizing table represents the approximate space consumption for metrics, about 7 MB for each metric per minute.
- The motherboard should not have more than 2 sockets.
- See Calculating Node Count in .NET Environments for information related to sizing a .NET environment.
- The agent counts do not reflect additional requirements for EUM or Database Visibility. See the following sections for more information.

Disk I/O Requirements

A critical factor in a machine's ability to support the performance requirements of a Controller is the machine's disk I/O performance.

There are two requirements related to I/O latency:

- This disk I/O must perform such that the maximum write latency for the Controller's primary storage must not exceed 3 milliseconds while the Controller is under sustained load. AppDynamics cannot provide support for Controller problems resulting from excessive disk latency.
- Self-monitoring must be set up for the Controller. Self-monitoring consists of a SIM agent that measures the latency of data partitions on the Controller host, and the configuration needs to include dashboard and health rule alerts that trigger when the maximum latency exceeds 3 ms. For details on Controller self-monitoring, contact your AppDynamics account representative.

Disk I/O Operations

The AppDynamics Controller performs two types of I/O operations important to Controller performance:

- The MySQL intent log is very sensitive to latency, and MySQL performs writes using varying block sizes.
- MySQL's InnoDB storage engine uses random, asynchronous, 16kB reads and writes to move database pages between storage and cache. In a properly sized Controller, most reads are satisfied from one of the software caches.

For best performance, it is important that the stripe size of the RAID configuration matches the write size. The two write sizes are 16Kb (for the database) and 128Kb (for the logs). If using a hardware-based RAID controller, be sure that it supports these stripe sizes. The stripe size can be determined by the number of data disks multiplied by the strip/segment/chunk (the portion of data stored on a single disk).

SAN Storage Limitations

While onboard disks typically satisfy I/O requirements, SAN-based storage could be hampered by poor I/O latency performance. In addition, AppDynamics discourages the use of an NFS-mounted filesystem. NFS adds latency and throughput constraints that can negatively affect Controller performance and even lead to data corruption. Similarly, you should avoid iSCSI or other SAN technologies that are subject to quality of service issues from the underlying network.

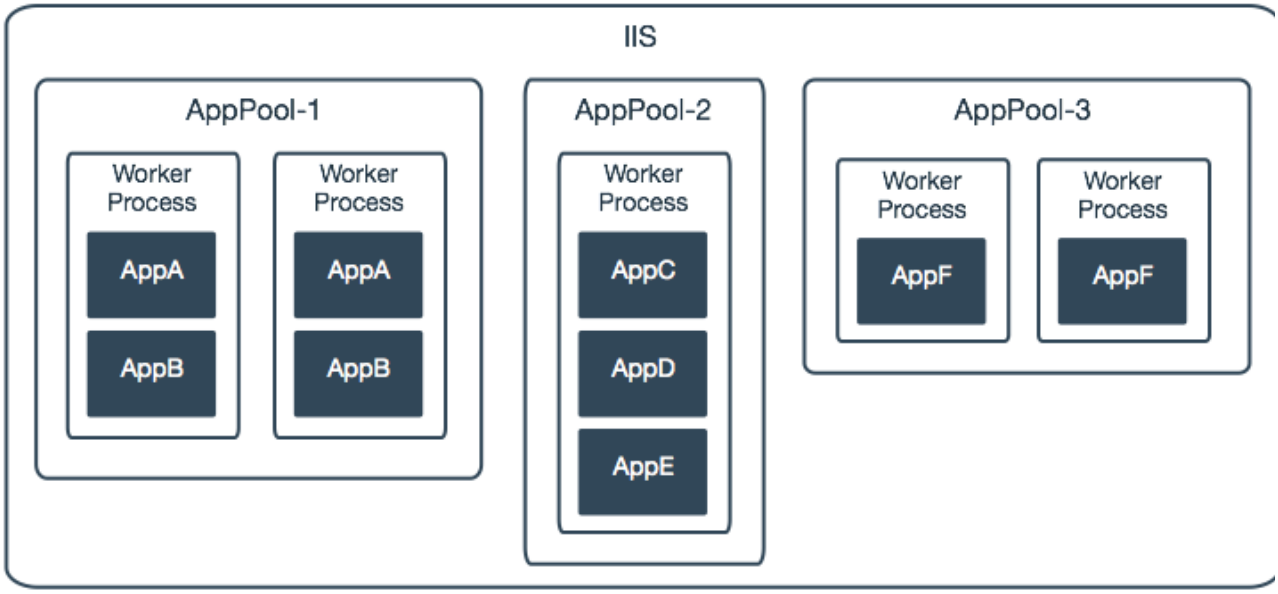
If you choose to deploy one of these latency-challenged storage technologies on a system that is expected to process 1M metrics/min or greater, a mirrored NVMe configured as a write-back cache for all storage accesses is recommended. Configuring such a device will hide some of the longer latencies that have been seen in these environments.

In all cases, be sure to thoroughly test the deployment with real-world traffic load before putting an AppDynamics Controller into a live environment.

Calculating Node Count in .NET Environments

The .NET Agent dynamically creates nodes depending on the monitored application's configuration in the IIS server. An IIS server can create multiple instances of each monitored IIS application. For every instance the .NET Agent creates a node. For example, if an IIS application has five instances, the .NET Agent will create five nodes, one for each instance.

The maximum number of instances of a particular IIS application is determined by the number of worker processes configured for its application pool, as illustrated in the following diagram:



The diagram shows three application pools — AppPool-1, AppPool-2, and AppPool-3 — with the following characteristics:

- AppPool-1 and AppPool-3 can have a maximum of two worker processes (known as a web garden), containing two applications (AppA, AppB) and one application (AppF), respectively.
- AppPool-2 can have one worker process. It has three applications.

To determine the number of nodes, for each AppPool, multiply the number of applications by the maximum number of worker processes. Add those together, as well as a node for the Windows service or standalone application processes.

The example would result in nine AppPool nodes. Adding one for a Windows service would result in a total of ten nodes, calculated as follows:

```
AppPool-1: 2 (applications) * 2 (max number of worker processes)  = 4
AppPool-2: 3 (applications) * 1 (max number of worker processes)  = 3
AppPool-3: 1 (application) * 2 (max number of worker processes)   = 2
Windows Service or standalone application process                 = 1
------
Total:                                                            = 10
```

To find the number of CLRs that will be launched for a particular .NET Application/App Pool:

1. Open the IIS manager and see the number of applications assigned to that AppPool.
2. Check if any AppPools are configured to run as a Web Garden. This would be a multiplier for the number of .NET nodes coming from this AppPool as described above.

Also see: http://technet.microsoft.com/en-us/library/cc725601(v=ws.10).aspx.

Sizing the Controller for the Events Service

The Events Service is a file-based storage facility used by EUM, Database Monitoring, and Analytics. Database Monitoring uses the Events Service instance embedded in the Controller by default. The disk space required will vary depending upon how active the databases are and how many are being monitored.

For redundancy and optimum performance, the Events Service should run on a separate machine. For details on sizing considerations, see Events Service Requirements.

Asynchronous Call Monitoring Considerations

The Small profile is not supported for installations with extensive async monitoring. A Medium profile running 40+ agents may need to upgrade to a configuration closer to a Large profile if extensive async monitoring are added.

Specifically, monitoring asynchronous calls increases the number of metrics per minute to a maximum number of 23000 per minute.

End User Monitoring (EUM) Considerations

End User Monitoring (EUM) typically increases the number of metrics collected. Accordingly, the Small profile is not supported for installations that use EUM. A Medium profile running 40+ agents should be sized at a specification closer to a Large profile for EUM.

Specifically, EUM impact metrics as follows:

- Web RUM can increase the number of individual metric data points per minute by up to 22000
- Mobile RUM can increase the number of individual metric data points per minute by as much as 15 to 25K per instrumented application, if your applications are heavily accessed. The actual number depends on how many network requests your applications receive.

The number of separate EUM metric *names* saved in the Controller database can be larger than the kinds of individual data points saved. For example a metric name for a metric for iOS 5 might still be in the database even if all your users have migrated away from iOS 5. So the metric name would no longer have an impact on resource utilization, but it would count against the default limit in the Controller for metric names per application. The default limit for *names* is 200,000 for Browser RUM and 100,000 for Mobile RUM.

Database Monitoring Considerations

The following guidelines can help you determine additional disk and RAM required for the machine hosting the Controller that is monitoring the Database Agent. For very large installations, you should work with your AppDynamics representative for additional guidelines.

For on-premises installations, the machine running the Controller and Event Service will require (for data retention period of 10 days):

- 1 – 10 collectors: 2 GB RAM, Single CPU
- 10 – 20 collectors: 4 GB RAM, 2 CPUs
- More than 20 collectors: 8 GB RAM, 4 CPUs