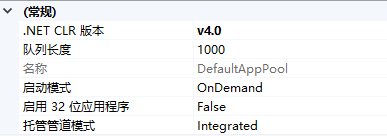
<https://www.cnblogs.com/gurenyumao/p/14149485.html>

<https://docs.microsoft.com/en-us/windows-server/administration/performance-tuning/role/web-server/tuning-iis-10>

# **参数说明**

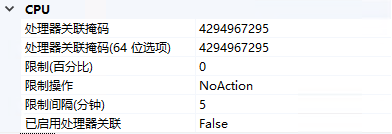
## **1.常规**



| 属性名称 | 属性详解 |
| --- | --- |
| NET CLR 版本 | 配置应用程序池，以加载特定版本的 .NET CLR。选定的 CLR版本应与应用程序所使用的相应版本的 .NET Framework 对应。选择“无托管代码”将导致所有的 ASP.NET 请求失败。 |
| 队列长度 | HTTP.sys 将针对应用程序池排队的最大请求数。如果队列已满，新请求将收到 503“服务不可用”的响应。默认队列长度设置是1000，范围在10-65535 之间。 |
| 名称 | 应用程序池名称是应用程序池的唯一标识符。 |
| 启动模式 | 将应用程序池配置为在按需运行模式或始终运行模式下运行。 |
| 启用 32 位应用程序 | 如果针对 64 位操作系统上的应用程序池将该属性设为 True,则为应用程序池提供服务的工作进程将处于 WOW64 （Windows on Windows64）模式。WOW64模式下的进程是仅加载 32 位应用程序的 32 位进程。 |
| 托管管道模式 | 将 ASP.NET 配置成作为 ISAPI 扩展并以经典模式来运行。在后一种情况下，托管代码集成到请求处理管道中。 |

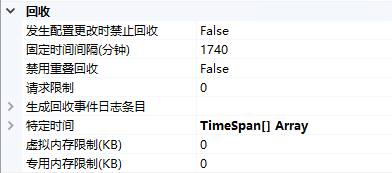
****Classic模式****：指的是与IIS 6或者之前版本保持兼容的一种模式，一个典型问题就是，在处理ASP.NET这种动态网站的时候，它是通过一个所谓的ISAPI程序，作为插件的方式来工作的。针对不同的动态应用程序（例如ASP,PHP等），会需要不同的ISAPI。  
****Integrated模式****：这种全新的模式，允许我们将ASP.NET更好地与IIS集成，甚至允许我们在ASP.NET中编写一些功能（例如Module）来改变IIS的行为（扩展）。集成的好处是，不再通过ISAPI的方式，提高了速度和稳定性。至于扩展，则可以使得我们对于IIS，以及其他类型的请求有更多的控制。

## **2.CUP**



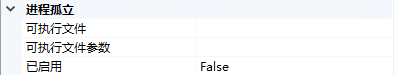
| 属性名称 | 属性详解 |
| --- | --- |
| 处理器关联掩码 | 强制此应用程序池的工作进程在特定 CPU 上运行的十六进制掩码。如果启用了处理器关联，则值 0 将导致错误。 |
| 处理器关联掩码（64位选项） | 为64位计算机制定强制此应用程序池的工作进程在特定 CPU 上运行的高顺序 DWORD 十六进制掩码。在 64 位计算机上，smpProcessorAffinityMask 特性包含处理器掩码的低顺序 DWORD ，而 smpProcessorAffinityMask2 特性包含处理器掩码的高顺序 DWORD。 |
| 限制（百分比） | 配置允许应用程序池中的工作进程在" CPU 限制间隔 "属性指示的时间段内使用的 CPU 时间的最大百分比。如果超过“ CPU 限制 ”属性设置的限制，系统将向事件日志写入一个事件，并且可能触发一组可选事件（由“CPU 限制操作”属性决定）。如果将此属性的值设为 0 ，将禁止将工作进程限制为 CPU 时间的百分比。 |
| 限制操作 | 如果设置为"NoAction"，将生成一个事件日志条目。如果设置为“KillW3WP”，则将在重设间隔期间关闭应用程序池并生成一个事件日志条目。如果设置为“ Throttle ”，则 CPU 使用率将限制为限制中设置的值。不使用限制间隔，并且生成一个事件日志条目。如果设置为“ ThrottleUnderLoad ”，则只有在争用 CPU 时，才限制 CPU 使用率。不使用限制间隔，并且生成一个事件日志条目。 |
| 限制间隔（分钟） | 指定用于应用程序池的 CPU 监视和限制的重设期限（以分钟为单位）。如果自上次进程计帐重设以来所经过的分钟数等于此属性指定的分钟数，IIS 将重设日志和限制间隔的 CPU 计时器。将此属性的值设为 0 将禁用 CPU 监视。 |
| 已启用处理器关联 | 如果设为 True ，“处理器关联掩码”属性会强制为此应用程序池提供服务的工作进程在特定的 CPU 上运行。这样便可以在多处理服务器中有效使用 CPU 缓存。 |

## **3.回收**



| 属性名称 | 属性详解 |
| --- | --- |
| 发生配置更改时禁止回收 | 如果为 True，应用程序池在发生配置更改时将不会回收。 |
| 固定时间间隔（分钟） | 一个时间段（以分钟为单位），超过该时间后，应用程序池将回收。值为 0 意味着应用程序池不会按固定间隔回收。 |
| 禁止重叠回收 | 如果为 True ,将发生应用程序池回收，以便在创建另一个工作进程之前退出现有工作进程。如果工作进程加载不支持多个实例的应用程序，请将该属性设为True。 |
| 请求限制 | 应用程序池在回收之前可以处理的最大请求数。如果值为0，则表示应用程序池可以处理的请求数没有限制。 |
| 生成回收事件日志条目 | 每发生一次指定的回收事件时便生成一个事件日志条目。 |
| ISAPI 报告了非正常状态 | 如果为True，则当应用程序池由于 ISAPI 扩展将其自身报告为非正常而进行回收时，系统将生成一个事件日志条目。 |
| 超出请求限制 | 如果为 True，则当应用程序池在超出其请求限制后进行回收时，系统将生成一个事件日志条目。 |
| 超出虚拟内存限制 | 如果为True，则当应用程序池在超出其虚拟内存限制后进行回收时，系统将生成一个事件日志条目。 |
| 固定时间间隔 | 如果为True，则当应用程序池按计划的间隔进行回收时，系统将生成一个事件日志条目。 |
| 手动回收 | 如果为True，则当手动回收应用程序池时，系统将生成一个事件日志条目。 |
| 特定时间 | 如果为True，则当应用程序池在计划的时间进行回收时，系统将生成一个事件日志条目。 |
| 已超出专用内存限制 | 如果为True，则当应用程序池在超出其专用内存限制后进行回收时，系统将生成一个事件日志条目。 |
| 应用程序池配置已更改 | 如果为True，则当应用程序池由于其配置发生更改而回收时，系统将生成一个事件日志条目。 |
| 特定时间 | 应用程序池进行回收的一组特定的本地时间（24小时制）。 |
| 虚拟内存限制（KB） | 工作进程可以使用的最大虚拟内存量（以 KB 为单位），超过此内存量，将导致应用程序池回收。如果值为 0 ，则表示没有限制。 |
| 专用内存限制（KB） | 工作进程可以使用的最大专用内存量（以 KB 为单位），超出此内存量，将导致应用程序池回收。如果值为0，则表示没有限制。 |

## **4.进程孤立**



| 属性名称 | 属性详解 |
| --- | --- |
| 可执行文件 | 当工作进程被废弃（孤立）时运行的可执行文件。例如，“C:\dbgtools\ntsd.exe”将调用 NTSD 来调试工作进程故障。 |
| 可执行文件参数 | 当工作进程被废弃（孤立）时所运行的可执行文件的参数。例如，如果 NTSD 是为调试工作进程故障而调用的可执行文件，则“-g -p %1%”适用。 |
| 已启用 | 如果设为True ，则无响应的工作进程将被废弃（孤立），而不是终止。可以使用此功能来调试工作进程故障。 |

## **5.进程模型**

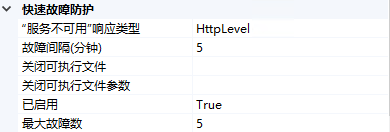


| 属性名称 | 属性详解 |
| --- | --- |
| Ping 间隔（秒） | 两次向为此应用程序池提供服务的工作进程发送健康状况监视 ping 所间隔的时间段（以秒为单位）。 |
| Ping 最大响应时间（秒） | 为工作进程指定的、响应健康状况监视 ping 的最长时间（以秒为单位）。如果工作进程不响应，将被终止。 |
| 标识 | 配置应用程序池以作为内置账户或特定的用户标识运行，内置账户也就是“应用程序池标识”（推荐）、“网络服务”、“本地系统”、“本地服务”。 |
| 关闭时间限制（秒） | 为工作进程指定的、完成处理请求并关闭的时间段（以秒为单位）。如果工作进程超过关闭的时间限制，将被终止。 |
| 加载用户配置文件 | 此设置指定 IIS 是否为应用程序池标识加载用户配置文件。当此值为 True 时，IIS为应用程序池标识加载用户配置文件。如果您需要像 IIS 6.0 那样不为应用程序池标识加载用户配置文件，则此值设置为 false。 |
| 空闲超时操作 | 达到空闲超时持续时间后要执行什么操作。 |
| 启动时间限制（秒） | 为工作进程指定的、启动并进行初始化的时间段（以秒为单位）。如果工作进程初始化时间超过启动时间限制，将被终止。 |
| 启用 Ping | 如果为 True，系统将定期对为此应用程序池提供服务的工作进程执行ping 操作，以确保这些工作进程仍及时响应。此过程称为健康状况监视。 |
| 生成进程模型时间日志条目 | 为每次发生的指定进程模型事件生成一个事件日志条目。 |
| 空闲超时已到 | 如果为 True，则当应用程序池在超出其空闲时限制后关闭时，系统将生成一个事件日志条目。 |
| 闲置超时（分钟） | 工作进程在关闭之前可以保持闲置状态的时间（以分钟为单位）。如果某个工作进程既未处理请求，也未收到任何新的请求，则将进入闲置状态。 |
| 最大工作进程数 | 可用来处理对应程序池的请求的最大工作进程数。如果此数字大于 1，则应用程序池为“Web 园”。在 NUMA 感知系统上，如果此数字为 0，则为获得最佳性能，IIS 将启动与 NUMA 节点一样多的工作进程。 |

标识详解：

* 本地系统： 具有高权限的受信任帐户，还具有对网络资源的访问权限。
* 网络服务： 用于运行标准的最小特权服务的受限或有限服务帐户。 此帐户具有比本地系统帐户更少的权限。 此帐户可以访问网络资源。
* 本地服务： 受限制或有限的服务帐户，与网络服务类似，旨在运行标准的最小特权服务。 此帐户无权访问网络资源。
* ApplicationPoolIdentity： 当创建新的应用程序池时，IIS 将创建一个虚拟帐户，该帐户具有新应用程序池的名称，并在此帐户下运行应用程序池工作进程。 这也是一个具有最小特权的帐户。
* 自定义帐户： 除了这些内置帐户之外，还可以通过指定用户名和密码来使用自定义帐户。

## **6.快速故障防护**



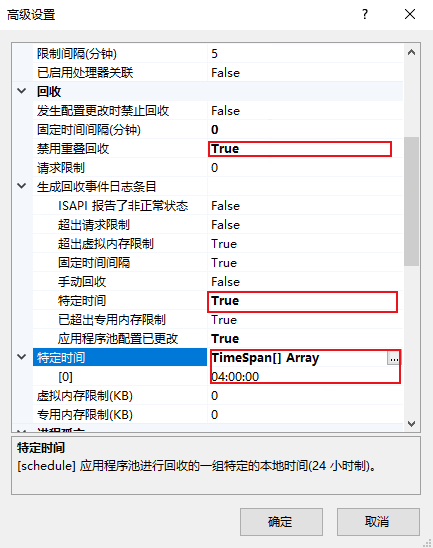
| 属性名称 | 属性详解 |
| --- | --- |
| 服务不可用响应类型 | 如果设为 HttpLevel,那么当应用程序池停止时， HTTP.sys 将返回 HTTP 503 错误。如果设为 TcpLevel，HTTP.sys 将重置连接。如果负载平衡器识别其中一种响应类型，并随后重定向该类型，则此设置非常有用。 |
| 故障间隔（分钟） | 应用程序池发生指定数量的工作进程崩溃（最大故障数）的最短时间间隔（以分钟为单位）。如果低于此间隔，应用程序池将被快速故障防护功能关闭。 |
| 关闭可执行文件 | 当应用程序池被快速故障防护功能关闭时所运行的可执行文件。可以使用它来配置负载平衡器，将此应用程序池的通信重定向至其他服务器。 |
| 关闭可执行文件参数 | 当应用程序池被快速故障防护功能关闭时运行的可执行文件的参数。 |
| 已启用 | 如果设为 True，则当在指定的时间段（故障间隔）内出现指定数量的工作进程崩溃（最大故障数）的情况时，应用程序池将被关闭。默认情况下，如果在5分钟的间隔内发生5次崩溃，应用程序池将被关闭。 |
| 最大故障数 | 应用程序池被快速故障防护功能关闭之前允许的最大工作进程崩溃数。 |

# **应用程序池优化配置**

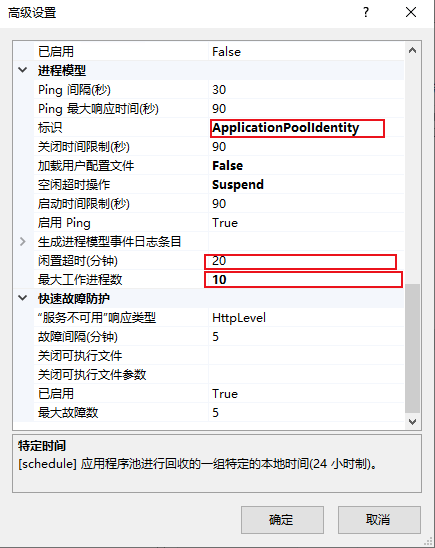
## **1.常规设置**

1. 队列长度： 默认值1000，将原来的队列长度改为 65535。
2. 启动32位应用程序：默认值False，改为True。
3. 托管管道模式：Integrated 或 Classsic。  
   

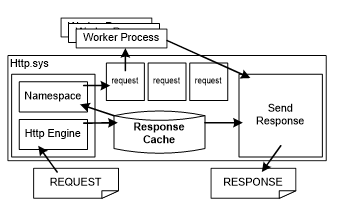
## **2.回收设置**

1. 禁用重叠回收。
2. 设置为特定时间=true，每天晚上04：00分回收。  
   

## **3.进程设置**

1. 标识设置,根据实际情况选择，可参照上面的用户定义。
2. 设置闲置超时，进程启动后，规定时间（根据实际情况设置）内未分配任务则回收此资源。
3. 设置工作进程数。  
   

# **HTTP.sys优化配置**

HTTP.sys 负责连接管理和请求处理。 可以从 HTTP.sys 缓存提供请求，或将请求传递到工作进程进行进一步处理。 可以配置多个工作进程，从而以降低的成本提供隔离。 有关请求处理的工作原理的详细信息，请参阅下图：  
  
HTTP.sys 包括响应缓存。 当请求与响应缓存中的条目相匹配时，HTTP.sys 会直接从内核模式发送缓存响应。 某些 web 应用程序平台（如 ASP.NET）提供了一些机制，可以在内核模式缓存中缓存任何动态内容。 IIS 10.0 中的静态文件处理程序在 http.sys 中自动缓存经常请求的文件。

## **1.内核模式设置**

与性能相关的 HTTP.sys 设置分为两大类：缓存管理和连接和请求管理。 所有注册表设置都存储在以下注册表项下：  
HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Services\Http\Parameters

## **2.缓存管理设置**

HTTP.sys 提供的一个优点是内核模式缓存。 如果响应在内核模式缓存中，则可以完全从内核模式满足 HTTP 请求，这会大幅降低处理请求所需的 CPU 开销。 但是，IIS 10.0 的内核模式缓存基于物理内存，项的开销是其占用的内存。  
缓存中的条目只有在使用时才有用。 但是，无论是否正在使用该条目，该条目始终使用物理内存。 你必须评估缓存中某项的有用性 (通过考虑可用资源 (CPU 和物理内存) 以及工作负荷要求，从缓存中为其提供服务所需的节省时间) 及其成本) (其成本。 HTTP.sys 尝试在缓存中仅保留有用的、主动访问的项，但你可以通过优化特定工作负荷的 HTTP.sys 缓存来提高 web 服务器的性能。  
下面是 HTTP.sys 内核模式缓存的一些有用设置：

* UriEnableCache 默认值：1  
  如果值为非零值，则启用内核模式响应和片段缓存。 对于大多数工作负荷，缓存应保持启用状态。 如果需要很低的响应和碎片缓存，请考虑禁用缓存。
* UriMaxCacheMegabyteCount 默认值：0  
  一个非零值，该值指定可用于内核模式缓存的最大内存。 默认值为0，使系统能够自动调整可用于缓存的内存量。
* UriMaxUriBytes 默认值：262144字节 (256 KB)  
  内核模式缓存中条目的最大大小。 不会缓存大于此的响应或碎片。 如果有足够的内存，请考虑增加限制。 如果内存有限，且大项 crowding 较小的项，则可能会降低限制。
* UriScavengerPeriod 默认值：120秒  
  清理程序会定期扫描 HTTP.sys 缓存，并删除清除程序扫描之间未访问的条目。 将清除周期设置为较高的值将减少清除清理的次数。 但是，缓存内存使用量可能会增加，因为在缓存中可以保留较旧、不经常访问的条目。 将该时间段设置得过低会导致清除清理次数过多，并可能导致刷新和缓存改动过多。

## **3.用户模式设置**

本部分中的设置将影响 IISÂ10.0 工作进程的行为。 其中的大多数设置都可以在下面的 XML 配置文件中找到：  
% SystemRoot% \ system32 \ inetsrv \ config \applicationHost.config  
使用 Appcmd.exe、IIS 10.0 管理控制台、WebAdministration 或 IISAdministration PowerShell Cmdlet 来更改它们。 大多数设置是自动检测的，它们不需要重启 IIS 10.0 工作进程或 web 应用程序服务器。 有关 applicationHost.config 文件的详细信息，请参阅[ApplicationHost.config简介](https://docs.microsoft.com/en-us/iis/get-started/planning-your-iis-architecture/introduction-to-applicationhostconfig" \t "https://www.cnblogs.com/gurenyumao/p/_blank)。

## **4.NUMA 硬件的理想 CPU 设置**

从 Windows 2016 开始，IIS 10.0 支持其线程池线程的自动理想 CPU 分配，以提高 NUMA 硬件的性能和可伸缩性。 此功能在默认情况下处于启用状态，可通过以下注册表项进行配置：  
HKEY\_LOCAL\_MACHINE\System\CurrentControlSet\Services\InetInfo\Parameters\ThreadPoolUseIdealCpu  
启用此功能后，IIS 线程管理器将尽最大努力基于当前负载在所有 NUMA 节点的所有 Cpu 之间平均分配 IIS 线程池线程。 通常情况下，建议将 NUMA 硬件的默认设置保持不变。

## **5.用户模式缓存行为设置**

本部分介绍影响 IISÂ10.0 中的缓存行为的设置。 用户模式缓存是作为一个模块来实现的，该模块侦听集成管道引发的全局缓存事件。 若要完全禁用用户模式缓存，请从 applicationHost.config 的 System.webserver/globalModules 配置节中的已安装模块列表中删除 FileCacheModule ( # A0) 模块。  
system.webserver/缓存

| Attribute | 说明 | 默认 |
| --- | --- | --- |
| 已启用 | 当设置为 False时，禁用用户模式的 IIS 缓存。 如果缓存命中率非常小，则可以完全禁用缓存，以避免与缓存代码路径相关联的开销。 禁用用户模式缓存不会禁用内核模式缓存。 | 正确 |
| enableKernelCache | 当设置为 False时禁用内核模式缓存。 | 正确 |
| maxCacheSize | 将 IIS 用户模式缓存大小限制为指定的大小（以 Mb 为单位）。 IIS 根据可用内存调整默认值。 根据经常访问的文件集的大小以及 RAM 或 IIS 进程地址空间的大小，仔细选择值。 | 0 |
| maxResponseSize | 将文件缓存到指定大小。 实际值取决于数据集中最大文件的数量和大小，以及可用 RAM。 缓存大型、频繁请求的文件可以降低 CPU 使用量、磁盘访问和相关的延迟。 | 262144 |

## **6.压缩行为设置**

默认情况下，从7.0 开始的 IIS 压缩静态内容。 此外，在安装 DynamicCompressionModule 时，会默认启用动态内容压缩。 压缩可减少带宽使用量，但会增大 CPU 使用率。 如果可能，压缩内容缓存在内核模式缓存中。 从8.5 开始，IIS 允许为静态和动态内容单独控制压缩。 静态内容通常是指不会更改的内容，如 GIF 或 .HTM 文件。 动态内容通常由脚本或服务器上的代码生成，即 ASP.NET 页面。 您可以自定义任何特定扩展的分类为静态或动态。  
若要完全禁用压缩，请从 applicationHost.config 的 System.webserver/globalModules 节中的模块列表中删除 StaticCompressionModule 和 DynamicCompressionModule。

## **7.并发设置**

### **ASP.NET 3.5**

默认情况下，ASP.NET 限制请求并发以降低服务器上稳定状态的内存消耗。 高并发性应用程序可能需要调整一些设置以提高整体性能。 可以在 aspnet.config 文件中更改此设置：

<system.web>

<applicationPool maxConcurrentRequestsPerCPU="5000"/></system.web>

以下设置对于完全使用系统资源非常有用：

* maxConcurrentRequestPerCpu 默认值：5000  
  此设置限制系统上同时执行的 ASP.NET 请求的最大数量。 默认值为保守以减少 ASP.NET 应用程序的内存占用。 考虑在运行执行长时间同步 i/o 操作的应用程序的系统上增加此限制。 否则，用户可能会遇到高延迟，因为在使用默认设置时，由于队列限制超出了队列限制，导致队列或请求失败。

### **ASP.NET 4.6**

除了 maxConcurrentRequestPerCpu 设置外，ASP.NET 4.7 还提供了一些设置，以提高严重依赖于异步操作的应用程序的性能。 可以在 aspnet.config 文件中更改设置。

<system.web>

<applicationPool percentCpuLimit="90" percentCpuLimitMinActiveRequestPerCpu="100"/></system.web>

* percentCpuLimit 默认值：90将大量负载 (超出硬件) 功能时，此类情况下会出现一些可伸缩性问题。 此问题是由对异步方案进行分配的性质导致的。 在这些情况下，将在异步操作启动时进行分配，并且在完成时将使用它。 在这段时间，itâs 非常可能将对象移动到第1代或第2代垃圾回收器。 发生这种情况时，增加负载会显示每秒请求数增加 (rps) ，直到达到某个点。 传递该点后，GC 中花费的时间会开始成为一个问题，rps 将开始进行 dip，这会造成负面影响。 若要解决此问题，当 cpu 使用率超出 percentCpuLimit 设置时，请求将发送到 ASP.NET 本机队列。
* percentCpuLimitMinActiveRequestPerCpu 默认值： 100 CPU 限制 (percentCpuLimit 设置) 不是基于请求数，而是取决于请求的费用。 因此，可能只需要几个占用 CPU 的请求，这会导致在本机队列中进行备份，使其不会从传入请求中清空。 若要解决此 problme，可以使用 percentCpuLimitMinActiveRequestPerCpu 来确保在限制开始之前提供最少数量的请求。

# **影响 IIS 性能的其他问题**

以下问题可能会影响 IIS 性能：

* 安装不能识别缓存的筛选器  
  如果安装的筛选器不能识别 HTTP 缓存，将导致 IIS 完全禁用缓存，从而导致性能不佳。 在 IISÂ6.0 之前编写的 ISAPI 筛选器会导致此行为。
* 通用网关接口 (CGI) 请求  
  出于性能方面的考虑，不建议在 IIS 中使用 CGI 应用程序来处理请求。 经常创建和删除 CGI 进程涉及到很大的开销。 更好的替代方法包括使用 FastCGI、ISAPI 应用程序脚本、ASP 或 ASP.NET 脚本。 每个选项都提供隔离。  
  ****注意****：  
  1.所有设置更改完毕，需要重启IIS。  
  2.更多详细设置，请参考[微软官方文档](https://docs.microsoft.com/en-us/windows-server/administration/performance-tuning/role/web-server/tuning-iis-10" \t "https://www.cnblogs.com/gurenyumao/p/_blank)。