PERFORMANCE AND THREADPOOLS

with HP Web Jetadmin

CONTENTS

OVERVIEW

HP Web Jetadmin is a powerful printer management software solution designed to install, configure, troubleshoot and manage a printer fleet. Sizing a Web Jetadmin installation and optimizing it for performance can be challenging. There are many variables involved to accurately provide a recommendation for number of devices. The hardware where Web Jetadmin is installed, enabled features, and number of devices will directly influence the performance of a Web Jetadmin installation. This document will discuss sizing a Web Jetadmin installation, features that affect performance, troubleshooting performance issues, and how to potentially improve the performance of many tasks in HP Web Jetadmin by manipulating the number of threads to be used for each task.

SIZING

A very common question when initially implementing HP Web Jetadmin is "How many devices can I include in my installation?" The question seems simple, but the answer is complex. Web Jetadmin contains a great deal of functionality and feature sets that vary in both resource requirements and performance expectations. It is not a one-size-fits-all type of solution. The hardware where Web Jetadmin is installed will directly influence the answer. The number of devices and types of features will certainly influence the answer. This paper will not provide an exact answer to the above question. However, it will examine features that affect sizing and performance, and it will provide examples of number of devices tested in various scenarios by the Web Jetadmin test lab for comparison purposes. The true number of supported devices on any given Web Jetadmin installation will depend upon many factors beginning with hardware.

HARDWARE REQUIREMENTS

Performance in HP Web Jetadmin will only be as powerful as the hardware where HP Web Jetadmin is installed. Commercial implementations of HP Web Jetadmin will function much better on server class hosts. Disk system performance is also critical to the operation of HP Web Jetadmin. Much of the disk activity is tied to HP Web Jetadmin interaction with a SQL database instance. High performance storage speeds up these operations and helps the application to run much better.

Server Hardware

HP recommends the following server hardware configuration:

- 4 or more processor cores
- 2.8 GHz or higher processor speed
- 4 GB or more of RAM
- 4 GB of available storage

HP highly recommends a 64-bit system.

Virtual Machines

For VMware environments, appropriate processors and RAM must be dedicated for each virtual machine. For a VMware server, the virtual machine network must be set to bridged to facilitate HP Web Jetadmin communications. It is very important to configure VMware so that its guest or virtual systems have enough resources to support HP Web Jetadmin and Microsoft SQL server. To ensure that the appropriate resources are provisioned, see the support documentation for the VMware version you are using.

64 bit vs. 32 bit

Recent software improvements in HP Web Jetadmin have increased resource capacity requirements. HP strongly recommends the 64-bit editions of Windows for production HP Web Jetadmin installations:

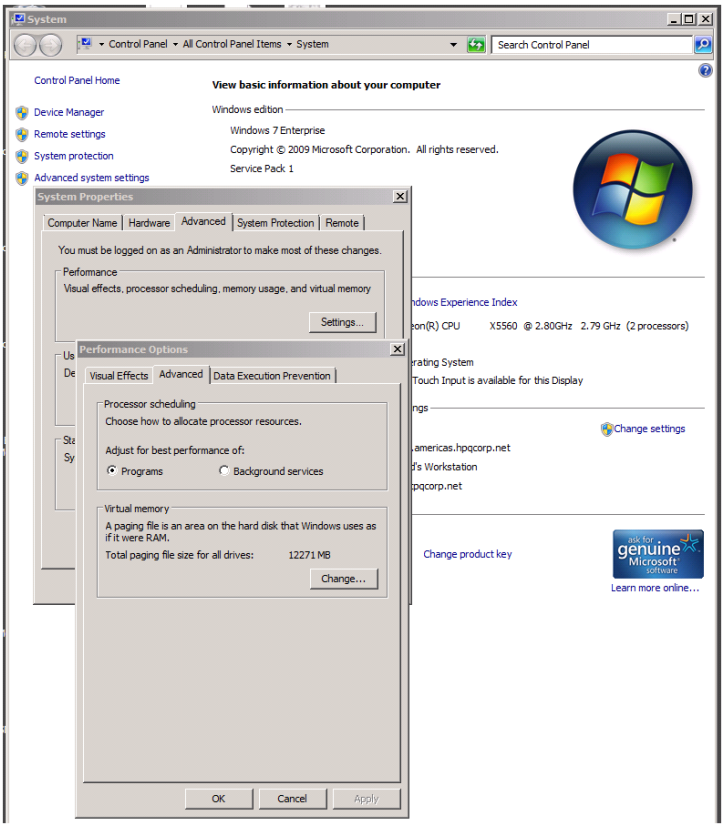
- Microsoft Windows Server 2008 (R2 or 64-bit edition)
- Microsoft Windows Server 2003 SP2 or later (64-bit edition)
- Microsoft Windows 7 (64-bit edition)

32 bit systems cannot manage memory effectively enough to handle the memory usage required in Web Jetadmin.

Memory

RAM Paging is a process that Windows uses when it runs out of RAM (memory). If Web Jetadmin is running on a host that has run out of RAM and the host has begun RAM paging, the application will respond very slowly. The UI will be very sluggish. Most of the applications on the host will also be sluggish. Steps can be taken to troubleshoot this problem. Free memory can be observed through Windows Task Manager, Performance tab, Physical Memory, Available. More memory may be needed to be added if available memory is low. Performance counters can be used to see if Web Jetadmin or the database needs additional memory. See Appendix B for more information. Running the client on a separate machine than the Web Jetadmin server can also help.

The Windows default maximum paging file size may be inadequate, even on 64-bit systems, for large Web Jetadmin installations supporting thousands of devices. When this happens, Windows may be unable to fulfill requests for additional memory causing the HPWJA Service to fail. The Windows 7 paging file size is configured in the Virtual Memory dialog of the Advanced Performance Options tab of the Advanced System Properties settings.



Disk

Another performance consideration is the disk subsystem. If the database is running from a very slow disk, performance will be affected. Also, the disk should be de-fragmented periodically to make sure the files are in a "good" state. Use drive-Properties, Tools, Defragmentation to determine if the drive is in need of a Defragment operation.

CONCURRENT USER LOGIN

Web Jetadmin is a client/server application meaning the client can run on a different machine than the server. The majority of the work is contained and managed by the server. Multiple clients can be connected to the same server. A client connects to the server and uses the .NET remoting channel to communicate. There are two types of communication that occur between the client and the server, client driven request and events from the server.

A client makes requests to the server in response to user actions or server events. This starts when the user requests to launch the client. The client establishes a connection with the server, retrieves updates, gets any information from the server needed to launch the UI. As the user uses the application, the client will request data from the server or request work to be done by the server as necessary. This traffic is generally very predictable as it is a direct result of work the user wants.

HP Web Jetadmin has been architected for multiple user access. The application was designed to support 15 or more concurrent connections while maintaining a high degree of performance. Of course, application usage will make the performance at each client vary somewhat depending on client load and the hardware hosting the application. The hardware and software requirements should be considered when the application is to be loaded with multiple and simultaneous user connections. An example would be where a user is tasked with discovery of all devices in the enterprise while at the same time another user is establishing groups and generating reports. In this case, on a slower host, HP Web Jetadmin may exhibit slow performance when large reports are being generated and when discoveries are being run at the same time. System resources, network bandwidth and other factors will have an impact on how well the software performs under these conditions.

APPLICATION PERFORMANCE

HP Web Jetadmin processes large amounts of data while allowing users to work productively. When clients are viewing device lists or accessing device configuration or status details, HP Web Jetadmin spends CPU bandwidth on multiple device data retrieval. In addition, the application collates and communicates device information back to multiple client sessions. An observant administrator using tools such as Microsoft's Performance Monitor can view the result of this background processing. Here are some characteristics about performance and HP Web Jetadmin software:

- During background task executions, the application might consume a high percentage of the host's processor for only short durations of time.
- Consumption of system processor may be pronounced on host systems that meet only minimum hardware requirements.
- Consumption of system processor may be pronounced on single host systems that are running both the HP Web Jetadmin application and the client application on the same host.

- Performance gains can be realized with HP Web Jetadmin running on multiple processor systems.

HP Web Jetadmin relies on network traffic to keep the information displayed to the user reasonably up to date. The product carefully makes numerous tradeoffs to balance network traffic with up to date information and performance.

The following section will examine several types of features and functionality in Web Jetadmin and their affect on performance. The following definitions are used to describe resource usage:

- Light - General minimal resource usage, with occasional spikes in resource usage. Impact on other applications would be minimal, but sometimes noticeable.
- Fluctuating Moderate - Moderate resource usage with frequent spikes in resource usage. Impact on other applications would be noticeable, but not generally excessive
- Fluctuating High - Resource Usage, especially CPU usage, is generally at 100% with regular dips down to lower levels. Impact on the OS and other applications is severe, although they remain functional.
- Severe High - Resource usage is High and remains there for extended periods. Web Jetadmin is generally not very responsive. Impact on the OS and other applications is severe, as resources are generally unavailable.

Generally all activities that engage in significant database activity, especially those that both read and write, will jump to the Fluctuating High resource usage level for the duration of the action. The system returns to a Light level of resource usage after the action has been completed.

Discovery

Running a discovery will generate traffic between the server and the device to identify the device and add it to HP Web Jetadmin. Some discovery methods are more expensive than others when it comes to network traffic. For instance, the active directory discovery method is one of the cheaper discovery methods to perform with regards to network traffic generated. It gleans IP addresses using print queue objects in the customer's active directory then talks to only network nodes that have printers on them. The same can be said of Specified Address discoveries as they are directed to specific IP addresses already known to be printers. Conversely, an IP range method is more expensive on the network because each network address in the range needs to be communicated with to determine if it has a printer that we know about.

Resource usage for the most part is somewhat minimal during discoveries regardless of the method.

Light to Fluctuating Moderate

Adhoc requests

Many parts of the client require device data. Depending on how up to date the client wants the device information, the server may return data it already has about the device, or it may need to talk to the device to get the information. This class of requests is generally driven by the user on an 'on demand' basis. With multiple clients, 'on demand' requests have a potential of overwhelming the system. To compensate for this, the Web Jetadmin server will try to reduce the device traffic through the use of thresholds. If multiple clients request the same data on the same device within some time threshold only one communication will go out to the device. Also, the relevant results of all device requests are shared with all clients regardless of the client that originally requested the data. This allows all clients to potentially benefit with up to date information at low cost.

Fluctuating Moderate

Device polling

The device list is core to the HP Web Jetadmin product. The device list content is automatically kept reasonably up to date through a slow polling mechanism on the server to each device. The entire device list is registered on a background poller for the columns in the current layout on a per client basis when that client is on a device list page. This mechanism is intended to gradually fill in the entire set of data to be ready when the user requests it by scrolling down through the list. Each device in view registers with the device list poller. This mechanism is intended to update the visible space at a much faster rate than the background poller would get the information. The registration occurs once the scrolling in the device list has stopped for a short time. The devices are unregistered once scrolling starts again.

Note: with a single client, the updates to the visible devices should start happening quickly at a rate consistent with the device list poller settings. With multiple clients, the updates should continue to be regular but with less frequency

Each object has a time threshold associated with it which allows HP Web Jetadmin to prevent a device communication if it already has reasonably up to date information. If one client just asked for the device description a minute ago, the threshold may cause that device communication to be skipped. Device data is cached in the client to minimize additional communication with the server. Generally, if a part of the UI requests device data, the client cache can prevent communication back to the server.

Light [Spikes in CPU usage during polling, dropping to negligible between polling]

Automatic groups

Automatic groups make use of the background poller. Any automatic group or filter group in the system registers a set of requests from the device when polled. When Web Jetadmin detects changes in the information being requested, it sends out a device changed event that causes groups and filters to re-evaluate the device in question against the group or filter criteria.

Basic (no-policy) bound groups cause a slight increase in resource usage, but not a significant amount. Increasing the number of groups populating at once and the number of devices going into the groups does little more than extend the time taken to populate the groups.

Autogrouping with multiple policies, including an alert subscription, can dramatically increase resource use. Large quantities of disk access can be caused by simultaneous reading of the device data from the database (to filter the devices), writing the new group membership to the database, writing the content of the custom request to the database, and reading the alert eligibility information from the device requests.

Fluctuating High

Report Generation

Data collection is performed once per day and fluctuates on amount of data collected. Report Generation analyzes the data that is collected nightly and formulates reports for the user. Both of these areas can cause Fluctuating High resource usage returning to Low as soon as the task is completed.

Fluctuating High

Device List Export

Exporting devices can be defined to only pull data from the database or to pull all data from all devices on-the-fly, so resource usage can vary. Large numbers of columns and devices can cause high levels of resource usage duration the duration of the polling of the devices.

Fluctuating High to Severe High

Device Configuration

Configuration occurs at a steady rate and never taxes the system above Fluctuating Moderate resource usage. Some configuration items cause more overall network traffic than others, but resource usage always remains low.

Light to Fluctuating Moderate

Alerts

Subscribing to a simple set of alerts (Paper Out, Toner Low, Toner Out, Offline, Printer Error, Intervention Needed, Paper Jam, Cover Open, and Manual Feed Needed) can cause varied system resource usage during this time, dropping to Fluctuating Moderate for short periods or running up to Severe High for short periods. The majority of the time it runs at Fluctuating High resource usage. After completion, it drops back down to Low resource usage.

Fluctuating High to Light when complete

Processing of alerts usually involves polling of devices dictated either by traps or an adaptive/static polling rate. Futuresmart devices use Web Services Eventing to process alerts. In all cases, resource usage maintains acceptable unless pollers are becoming overloaded.

Light to Fluctuating Moderate

Firmware

Populating firmware information on the firmware tab can spike due to the size of the files being imported. Downloading firmware is typically light on system resources as it involves opening up a Port 9100 TCP connection and sending down an rfu file for many devices. Futuresmart devices use web services for upgrades.

Light to Fluctuating Moderate (when running 8 concurrent upgrades)

SIZING EXAMPLES

While it is difficult to provide exact recommendations as all installations will vary, the Web Jetadmin team has run many tests to provide examples of what can be performed before performance degrades under certain scenarios. The team benchmarked the following scenarios to attempt to determine the "tipping point" when Web Jetadmin performance degrades.

Test Plan Platforms - all systems running Windows 2003 R2

- 8 VMs running base hardware configuration – Dual core processor with 4GB RAM: client, server, and DB running on the same system.
- 1 VM running base configuration with off-box DB.
- 1 VM running a quad processor with 12 GB RAM.

Test Plan Scenarios

- Two of the machines performed Discovery (one with a device list of 1100 and the other with a device list of 4000).
- Two of the machines configured Alerts that began at a "safe" level and slowly increased until a failure point is encountered (one with a device list of 1100 and the other with a device list of 4000). The Alerts exercised were General Alerts with the options of Advisory, Media Path, and Supplies.
- Two of the machines performed Data Collections that began at a "safe" level and slowly increased until a failure point is encountered (one with a device list of 1100 and the other with a device list of 4000).

- Two of the machines performed Data Collections with Hourly Peak Usage exclusively (one with a device list of 1100 and the other with a device list of 4000). This is due to early experiments showing that this particular collection presents a heavy load due to the frequency of queries to the devices.

Test Plan Strategy – Scenario Matrix

- At the completion of these scenarios there will be a matrix applied that will entail each system performing the various operations (Discovery, Alerts, Data Collections) simultaneously, with a slowly increasing load that will mirror the "ramp" of the individual scenarios (one with a device list of 1100 and the other with a device list of 3900 = 16* subnet .)

Completed Discovery

- Discovery - Imported list of 1100 devices Result = Success
- Discovery - Run Full Subnet through Templates with each portion being smaller than a Class B Network. Result = Success

Completed Alerts

- Alerts - (1100 devices) General Alerts - 100 to Max Result = Success
- Alerts - (4000 devices) General Alerts: 100 to Max Result = Success

Completed Data Collection

- Data Collection - (1100 devices) All except Hourly Peak Usage - 100 to Max
  Result = Success
- Data Collection - Hourly Peak Usage (1100 devices) - Data Collection - 100 to Max
  Result = Success
- Data Collection - (4000 devices) All except Hourly Peak Usage - 100 to Max
  Result = Success

Completed Remote DB with SSD

- Data Collection - (4000 devices) SSD Remote DB - All except Hourly Peak Usage : 100 to Max
  Result = Success
- Alerts - (4000 devices) SSD Remote DB General Alerts: 100 to Max
  Result = Success
- Data Collection - (4000 devices) SSD Remote DB – Hourly Peak Usage : 100 to Max
  Result = Success

Conclusions

- Individual tasks, no matter how intensive, do not cause issues for Web Jetadmin when running on 64 bit systems when the Device List is limited to approximately 1100 devices.
- System intensive tasks such as Hourly Peak Usage can successfully be applied to 1100 devices with no adverse reactions in Web Jetadmin, including responsiveness of the client, even when client is running on the same host.
- When Device Lists of 4000 devices are used, individual tasks only slow performance at the client, but do not cause system losses (crashes etc.)
- The only limitation on a system with 1100 devices is when applying a large Alert subscription and creating a large Data Collection simultaneously. This can cause the client to become temporarily unresponsive. In this case the server continues to run without interruption.

As a result of these tests, it could be concluded that a system in "full utilization" can support roughly 4000 devices or more. Remember, this is a system that is being utilized with many HP Web Jetadmin

features activated. Can one HP Web Jetadmin server support more than 4000 devices? Absolutely. However, the answer also depends on how you are loading features and devices as well as the types of features and settings activated. As more features are activated on devices within a single Web Jetadmin host, the number of potential devices under management decreases. Administrators may or may not plan to use all HP Web Jetadmin features in one HP Web Jetadmin implementation. For example, there have been installations of Web Jetadmin exceed 20,000 devices in the Device List, but in such a case the installation is only running data collections and reports.

Distributing Web Jetadmin Across Multiple Servers

The first factor in sizing is to decide whether one implementation of Web Jetadmin is possible or should the implementation of Web Jetadmin be broken into multiple servers. If it is believed the number of devices and desired features will overload a single server, perhaps the installations could be split by groups of devices or by functionality.

Consider the multi-regional approach:

- Europe 2300 devices and Reports are needed
- North America 6000 devices and both Reports and Alerting are needed
- Asia 3000 devices and Reports are needed

In this case there is both functionality and devices to consider. In the Europe and Asia cases one server could be implemented for each population, while in the North America case two server implementations may be considered.

Here are some more hints on how Web Jetadmin features + devices impact the system:

Consolidated reports - This is one item that can't really be accomplished through distribution across multiple servers. If the number of devices is large, deployment should consider one server dedicated to reports data collection. In very scaled environments, data may have to be drawn out of multiple Web Jetadmin servers using Database Views.

By User/Peak Usage reports - If By User or Peak Usage is required across a large number of devices, then consolidated reports is probably not an option. These data collections MUST be distributed, so hopefully this work can be distributed across multiple servers by geography, function, or some other logical grouping.

Alerts - Alerts is a very common bottleneck, however, it is also the easiest to distribute across servers. Reducing the alerts load on any given server will increase the performance and responsiveness. Regional or organizational alerting across multiple servers is easy to implement.

Auto-grouping – Many customers may use auto-grouping to drive configuration policies. Auto-grouping causes a constant draw from system resources, so there is often benefit to have a server dedicated to configuration and auto-grouping policies. Many times there is a difference between groups that provide general management structure and organization and groups that drive automatic functionality. It should also be considered whether those auto-groups need to be replicated across all of their servers.

devices per server – when looking to distribute functionality across servers, it's useful to limit the number of devices in each system. This will increase the responsiveness across devices for the intended functionality. For instance, if you have a server dedicated for alerts in the North America region, make sure that the server only knows about the devices in the North America region that need alerts.

Client load - Client load is another item to consider, although it is harder to distribute unless there is a clear delineation by geography or other logical grouping of devices. Many times it is recommended to set aside a server for help desk functionality that tends to have more ad hoc use models.

Sample Design Proposal

Once all options have been considered and the customer environment is understood, it's time to design a proposal. Let's consider a case study of a Web Jetadmin implementation. Key functionality required:

Deploy
- Discover, configure, update firmware on a fleet of 3000 devices
- Use auto-groups + templates to apply security settings
- Export various device list fields to CSV on a schedule
- 15 user potential

Resolve Issues
- Investigate device issues remotely
- Apply resets remotely
- Send PJL and config templates to devices
- Monitor (proactive) 3000 devices
- 50 user potential

Reporting
- Establish data collection on all but Peak Usage and By User on 3000 devices
- Retain data for up to 5 years
- 15 user potential

Design Recommendation – distribute functionality across 3 servers

Here is an actual proposal from a customer implementation in 2009:

• Server 1
– Proactive Management - Distributed Alerts
– 2 or 4 core SQL Express
– ~3000 devices
– Subscriptions 1-3
• Server 2
– Fleet Deployment & Trend Reporting - Consolidated Reports + Admin support
– 4 core + off box SQL
– 3000 devices
– ~15 admins

THREADPOOL SETTINGS

HP Web Jetadmin 10.3 provides an optional technique for improving the performance of many tasks by manipulating the number of threads to be used for each task. Threadpool settings can be configured in an XML file named PerformanceTuning.config.xml under the following directory for Windows 7 and Server 2008 (R2):

> C:\Users\Network Service\Local Settings\Application Data\Hewlett-Packard\HPWebJetadmin\WJAService\Config

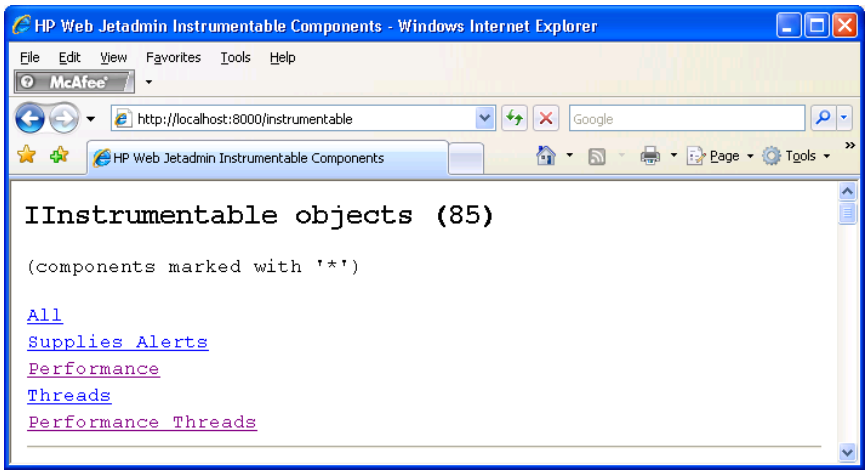
NOTE: On 32 bit systems, it should never be attempted to edit this file. 32 bit operating systems are not equipped to handle any increase in threads.

The suggested use is to monitor the performance threads page under the IInstrumentable page as each default value is updated. Look for large numbers of 'queued' where running is equal to max. Increasing these values may increase performance and may cause other thread pools to back up.

MOAB's IInstrumentable Page

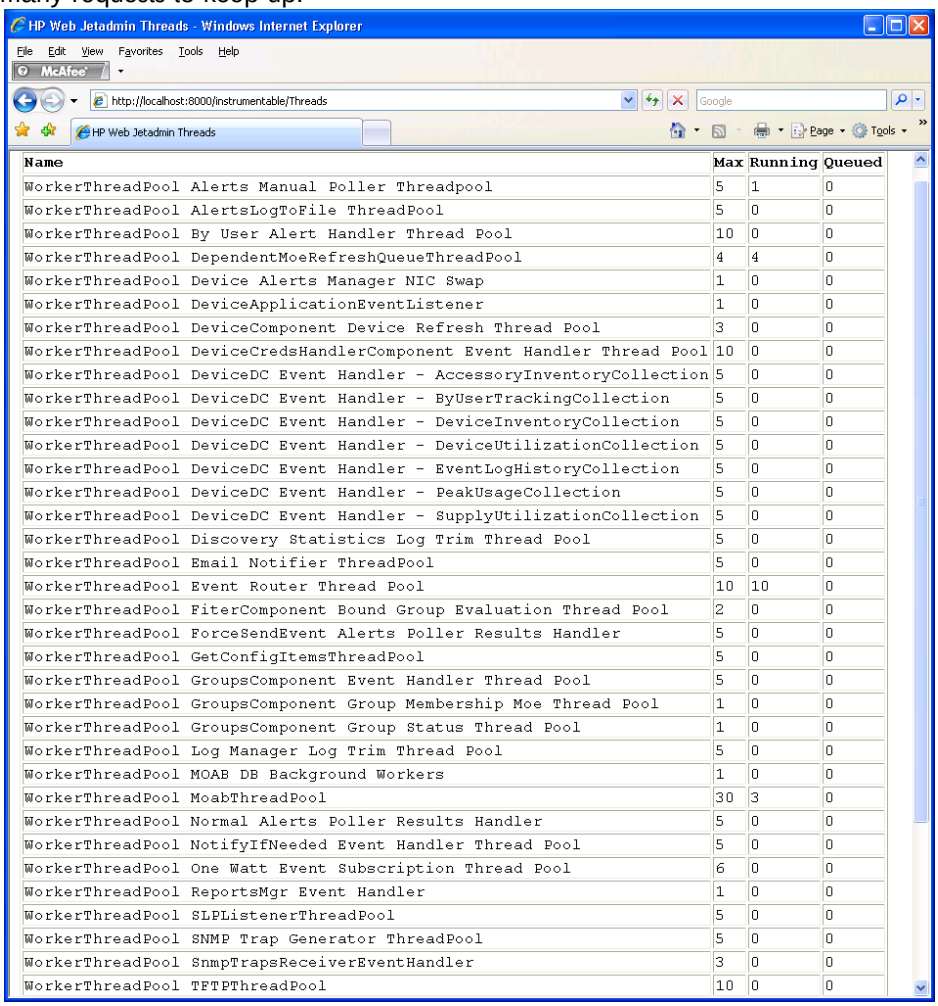
To understand threadpools, one must first understand a critical piece of architecture called MOAB (Managed Object Abstraction) in Web Jetadmin. During discovery, after nodes are discovered, MOAB will further process the nodes to determine if they are devices. MOAB is more or less responsible for getting and setting information on devices. It manages device communication, trying to reduce it where possible by keeping a local cache of data and thresholds for how fresh the data should be. Grouping requests to devices are also managed by MOAB. MOEs are the individual requests issued by MOAB.

First place to look when MOAB is running slow is on the IInstrumentable page found by adding /instrumentable to the Web Jetadmin URL in a browser.



The purpose of using this page is to look for backed up threads. Under the Threads link, If the line "WorkerThreadPool MoabThreadPool" has queued items that aren't decreasing, MOAB is getting too

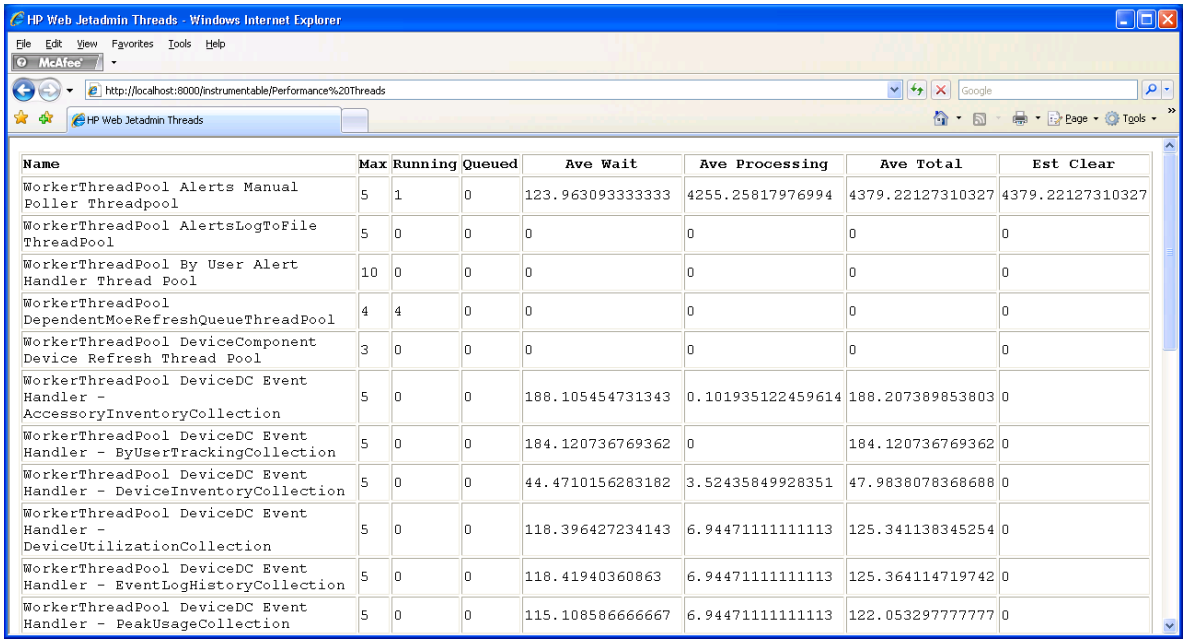
many requests to keep up.



Name	Max	Running	Queued
WorkerThreadPool Alerts Manual Poller Threadpool	5	1	0
WorkerThreadPool AlertsLogToFile ThreadPool	5	0	0
WorkerThreadPool By User Alert Handler Thread Pool	10	0	0
WorkerThreadPool DependentMoeRefreshQueueThreadPool	4	4	0
WorkerThreadPool Device Alerts Manager NIC Swap	1	0	0
WorkerThreadPool DeviceApplicationEventListener	1	0	0
WorkerThreadPool DeviceComponent Device Refresh Thread Pool	3	0	0
WorkerThreadPool DeviceCredsHandlerComponent Event Handler Thread Pool	10	0	0
WorkerThreadPool DeviceDC Event Handler - AccessoryInventoryCollection	5	0	0
WorkerThreadPool DeviceDC Event Handler - ByUserTrackingCollection	5	0	0
WorkerThreadPool DeviceDC Event Handler - DeviceInventoryCollection	5	0	0
WorkerThreadPool DeviceDC Event Handler - DeviceUtilizationCollection	5	0	0
WorkerThreadPool DeviceDC Event Handler - EventLogHistoryCollection	5	0	0
WorkerThreadPool DeviceDC Event Handler - PeakUsageCollection	5	0	0
WorkerThreadPool DeviceDC Event Handler - SupplyUtilizationCollection	5	0	0
WorkerThreadPool Discovery Statistics Log Trim Thread Pool	5	0	0
WorkerThreadPool Email Notifier ThreadPool	5	0	0
WorkerThreadPool Event Router Thread Pool	10	10	0
WorkerThreadPool FiterComponent Bound Group Evaluation Thread Pool	2	0	0
WorkerThreadPool ForceSendEvent Alerts Poller Results Handler	5	0	0
WorkerThreadPool GetConfigItemsThreadPool	5	0	0
WorkerThreadPool GroupsComponent Event Handler Thread Pool	5	0	0
WorkerThreadPool GroupsComponent Group Membership Moe Thread Pool	1	0	0
WorkerThreadPool GroupsComponent Group Status Thread Pool	1	0	0
WorkerThreadPool Log Manager Log Trim Thread Pool	5	0	0
WorkerThreadPool MOAB DB Background Workers	1	0	0
WorkerThreadPool MoabThreadPool	30	3	0
WorkerThreadPool Normal Alerts Poller Results Handler	5	0	0
WorkerThreadPool NotifyIfNeeded Event Handler Thread Pool	5	0	0
WorkerThreadPool One Watt Event Subscription Thread Pool	6	0	0
WorkerThreadPool ReportsMgr Event Handler	1	0	0
WorkerThreadPool SLPListenerThreadPool	5	0	0
WorkerThreadPool SNMP Trap Generator ThreadPool	5	0	0
WorkerThreadPool SnmpTrapsReceiverEventHandler	3	0	0
WorkerThreadPool TFTPThreadPool	10	0	0

The IInstrumentable line indicates if there are queued items less than or equal to the thread count. The key is to determine if there are more items queued than the max thread count consistently over time. The values are pretty meaningless for debugging if looked at for a given point in time. It is when one of those queued values stays high over time when there may be consideration for adjusting threadpool values. There are many threadpools/queues where Web Jetadmin adds quite a bit of work to it, so the queued items may be 10x or more than the number of threads. The threads then do the work and return to zero before the next amount of work is added (typically.) It is when there are always too many queued items or the queued items are constantly growing that should be of concern.

Another area to check is the Performance Threads link. Check if any of the categories have a large average wait time, perhaps bumping that thread count would help.



Name	Max	Running	Queued	Ave Wait	Ave Processing	Ave Total	Est Clear
WorkerThreadPool Alerts Manual Poller Threadpool	5	1	0	123.963093333333	4255.25817976994	4379.22127310327	4379.22127310327
WorkerThreadPool AlertsLogToFile ThreadPool	5	0	0	0	0	0	0
WorkerThreadPool By User Alert Handler Thread Pool	10	0	0	0	0	0	0
WorkerThreadPool DependentMoeRefreshQueueThreadPool	4	4	0	0	0	0	0
WorkerThreadPool DeviceComponent Device Refresh Thread Pool	3	0	0	0	0	0	0
WorkerThreadPool DeviceDC Event Handler - AccessoryInventoryCollection	5	0	0	188.105454731343	0.101935122459614	188.207389853803	0
WorkerThreadPool DeviceDC Event Handler - ByUserTrackingCollection	5	0	0	184.120736769362	0	184.120736769362	0
WorkerThreadPool DeviceDC Event Handler - DeviceInventoryCollection	5	0	0	44.4710156283182	3.52435849928351	47.9838078368688	0
WorkerThreadPool DeviceDC Event Handler - DeviceUtilizationCollection	5	0	0	118.396427234143	6.94471111111113	125.341138345254	0
WorkerThreadPool DeviceDC Event Handler - EventLogHistoryCollection	5	0	0	118.41940360863	6.94471111111113	125.364114719742	0
WorkerThreadPool DeviceDC Event Handler - PeakUsageCollection	5	0	0	115.108586666667	6.94471111111113	122.053297777777	0

ADJUSTING THREADPOOL VALUES

If viewing the performance threads page and nothing appears to be backed up, increasing thread pools may not help much. If something is backed up, increasing the threadpools could decrease CPU load if items are backing up behind those threadpools as a larger number of threads allows more work to finish.

The values that Web Jetadmin offers to be changed in the PerformanceTuning.config.xml configuration file are ones that are tested and produced a performance improvement by increasing the values.

Name	Default Value	Description
ByUserTrackingCollection.Threads	10	Increasing this may help if performing 'by user' data collections on many devices. Will be limited by moab threads.
DavAlertL2FComponent.Threads	5	Increasing this may help if using 'alerts log to file' for large numbers of alert subscriptions
DavEmailNotifierComponent.Threads	5	Increasing this may help if using the email alert notifier for large numbers of alert subscriptions
DavSnmpTrapGenerator.Threads	5	Increasing this may help if using the SNMP Trap Generator notifier for large numbers of alert subscriptions
DeviceComponent.BackgroundRefreshThreads	3	Number of threads used for handling device inactivated and re-activated events. Probably won't help performance.
DeviceConfigComponent.Threads	5	Increasing this may help if performing device configurations on many devices. Will be limited by moab threads.
DeviceDC Event Handler -	5	Increasing this may help if collecting

AccessoryInventoryCollection.ThreadCount		accessory inventory data on many devices. Only used to handle incoming events.
DeviceDC Event Handler - ByUserTrackingCollection.ThreadCount	5	Increasing this may help if collecting by user data on many devices. Only used to handle incoming events.
DeviceDC Event Handler - DeviceInventoryCollection.ThreadCount	5	Increasing this may help if collecting device inventory data on many devices. Only used to handle incoming events.
DeviceDC Event Handler - DeviceUtilizationCollection.ThreadCount	5	Increasing this may help if collecting device utilization data on many devices. Only used to handle incoming events.
DeviceDC Event Handler - EventLogHistoryCollection.ThreadCount	5	Increasing this may if collecting event log data on many devices. Only used to handle incoming events. .
DeviceDC Event Handler - PeakUsageCollection.ThreadCount	5	Increasing this may help if collecting peak usage data on many devices. Only used to handle incoming events.
DeviceDC Event Handler - SupplyUtilizationCollection.ThreadCount	5	Increasing this may help if collecting supply utilization data on many devices. Only used to handle incoming events.
DevicePollingReceiver.ForceSendEvent Alerts Poller Results HandlerThreads	5	Increasing this may help performance during alert subscriptions. Would also help when receiving traps from devices.
DevicePollingReceiver.Normal Alerts Poller Results HandlerThreads	5	Increasing this may help performance for background alerts polling.
DevicePollingReceiver.NumManualPollerThreads	5	Increasing this may help if many alerts are polling only alerts. May help subscribe time for alerts.
DiscoveryManager.NumHostIpResolverThreads	30	Increasing this may help reduce discovery times.
DiscoveryManager.NumResolverThreads	10	Increasing this could help reduce discovery times.
DiscoveryManager.SLPListenerThreadPool	5	Increasing this may help with SLP listen discovery.
DiscoveryMethodSpecifiedAddress.PingThreadCount	30	Increasing this may help if using specified address or subnet range for discovery.
FilterComponent.Threads	2	This is the thread pool used to evaluate auto-group membership. Increasing this may help if have a large number of auto-groups defined.
GroupsComponent.EventHandlerThreads	5	Event handler for groups component. These refer to internal WJA server events. Some examples are: GroupMembershipChangedEvent (someone removed a device from the group), GroupAttributeChangedEvent, EventMoabDataCacheChange (device data changed), EventMoabDeviceActivated (new device in the system).
MoabDevice.DependentMoeRefreshQueueThreads	4	Whenever a moe changes that other moes

		have a dependency on, the thread pool is used to update the moes that have the dependency in the background after sending the event saying the original moe has changed. This is used to "ripple" changes through the system for moes that depend on each other. As an example, the moe 'SupplyBlackCartridge' has dependencies on 'SupplyBlackTonerCartridge' and 'SupplyBlackInkCartridge'.
MoabDevice.LockBucketSize	23	This determines how many DeviceIDs can be locked at a time when reading/writing data to the database for a particular device.
MoabDevice.MaxConcurrentThreadsInResolution	10	Increasing may help moab portion of discovery resolution (shortening overall discovery times)
MoabDevice.MaxThreadPercentageUsedDuringDiscovery	50	The percentage of the overall MoabThreadPoolSize which can be used for discovery.
MoabDevice.MaxThreadPercentageUsedDuringStartup	25	The percentage of the overall MoabThreadPoolSize which can be used during startup.
MoabDevice.MaxThreadPercentageUsedForPolling	25	The percentage of the overall MoabThreadPoolSize which can be used for polling. The pollers affected would only be the moab pollers. This is the setting that affects all moab operations (best bang for the buck). Any operation that requires device communication (either get or set) will benefit.
MoabDevice.MoabThreadPoolSize	30	May help to increase performance when doing actions across multiple devices.
NotificationManager.NumNotificationThreads	6	Used for processing and sending alerts. Increasing may help if many active alert subscriptions.
OxpmComponent.AlertThreads	5	May help if using the OXPm client for alert notification for many devices.
SnmpTrapsReceiver.TrapsEventHandlerThreadCount	3	May help to increase performance if WJA instance is receiving large number of incoming traps (by user data collection, alert subscriptions).
Tftp.TFTPThreadPoolCount	10	May help if doing firmware upgrade on many JetDirect NICs.
WebServer.Threads	5	May help if many WJA clients are being used simultaneously.

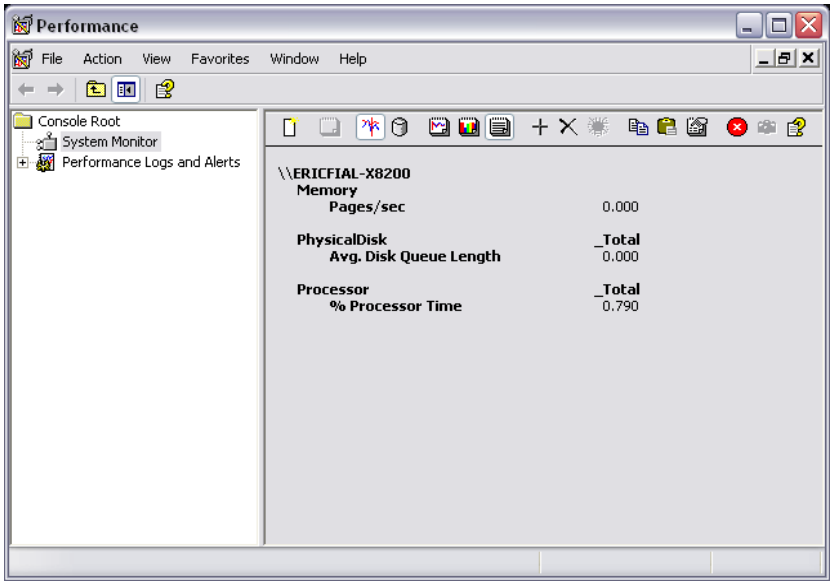
SUMMARY

Sizing a Web Jetadmin installation and optimizing it for performance can be challenging. The hardware where Web Jetadmin is installed, enabled features, and number of devices will directly influence the performance of a Web Jetadmin installation. This document discussed techniques for how to potentially improve the performance of many tasks in HP Web Jetadmin by manipulating the number of threads to be used for each task. Configuring the Windows Paging File Size to provide reliable support for very large servers is also crucial to improving performance and eliminating memory out conditions (crashes).

APPENDIX A

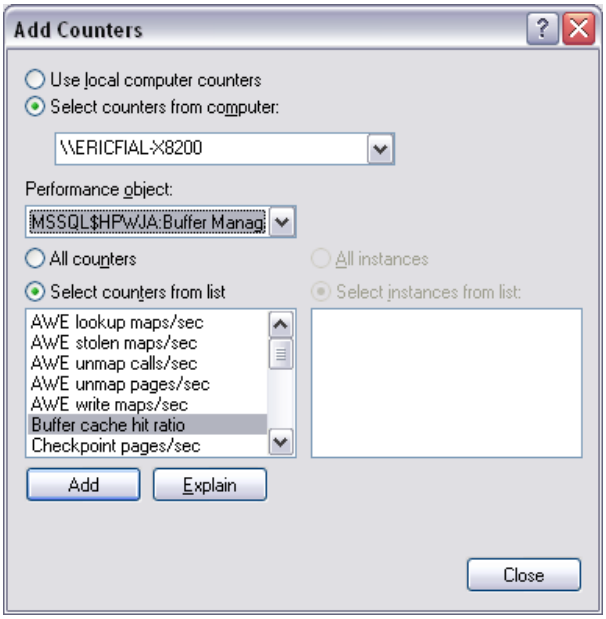
Checking Memory Consumption

Run Perfmon (Start, Run, "perfmon", OK) and switch to "View Report" (Ctrl-R)



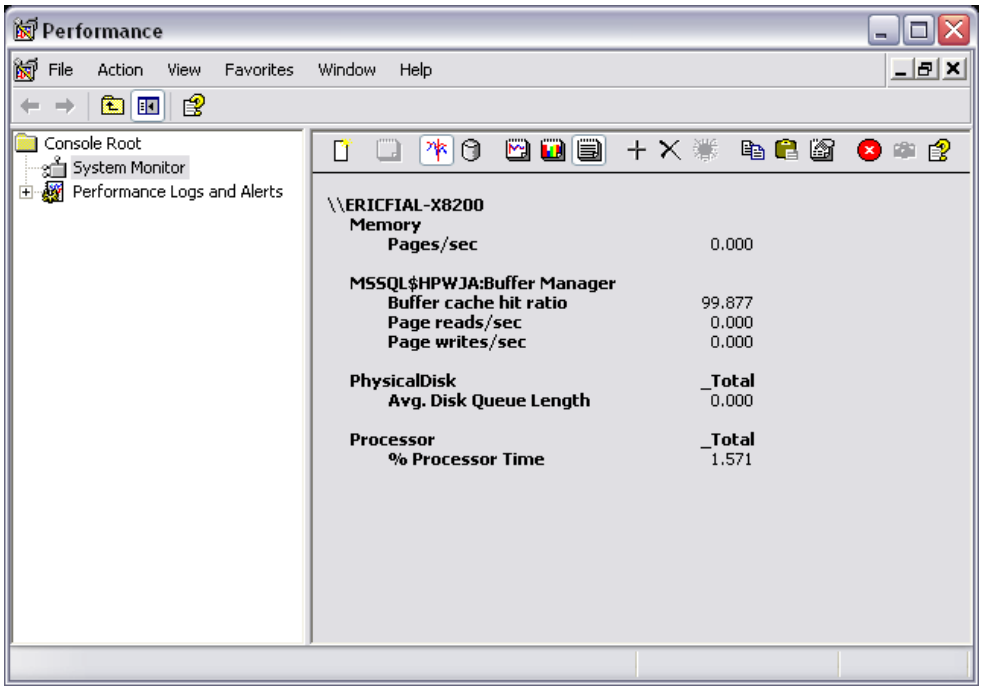
Having an Avg. Disk Queue Length higher than 2 for anything more than a short period of time, is the first sign that more memory may be needed.

Next, right-click and choose "Add Counters" Under Performance object, choose "MSSQL$HPWJA:Buffer Manager"



Add the objects: “Buffer cache hit ratio”, “Page reads/sec” and “Page writes/sec” then close this dialog.

If any of the following are true, more memory may be needed:
Buffer cache hit ratio < 97%
Page reads/sec > 10 for an extended period of time
Page writes/sec > 10 for an extended period of time
These numbers will jump from time to time during normal operations but if they are consistently out of norms, it usually indicates that the database is spending time moving data out of its cache and may benefit from more memory. Typically, when a machine is behaving “badly” the Page reads/sec and/or the Page writes/sec are in the 100’s and the Avg. Disk Queue Length is in double or triple digits.



c03270161_ENW, Rev.1, June 2012