Tivoli Federated Identity Manager

Performance Tuning Guide

Version 1.0

This document is intended to be used by FIM customers for general FIM performance tuning and best practices. The guide assumes the audience has certain level of knowledge about WebSphere Application Server and Tivoli Federated Identity Manager.

Document: Luobin Wang/Singapore/IBM (wanglb@sg.ibm.com)

Table of content

Chapter 1: Tune WAS JVM heap for FIM

As an application running on WAS (WebSphere Application Server), FIM (Federated Identity Manager) will need a proper WAS JVM (Java Virtual Machines) to perform well. In this section, the discussion will go over WAS Heap size determination and garbage collection settings to suit FIM performance.

1.1 WAS JVM heap setting place

To modify WAS JVM heap settings:

a. *In the administrative console, click* ***Servers > Server Types > WebSphere application servers >*** *server_name.*

b. *In the Server Infrastructure section, click* ***Java and process management > Process definition > Java virtual machine***.

c. *Specify a new value in either the* ***Initial heap size*** *or the* ***Maximum heap size*** *field.*

1.2 Recommend setting for JVM heap

The recommended JVM heap size for FIM to run is 1 GB. However, it may need to change based on incoming request rate and session settings. This will be discussed further in later sections.

1.3 Understanding the implications of heap size

Setting the heap as larger as possible may not be a good idea for several reasons:

a. *The start up of the JVM will take longer time.*

b. *When garbage collection happens, it will take longer time.*

Setting the heap small will also have some disadvantages. The most important one is that it may cause the garbage collection process to happen every frequently and cause the application performance throughput to drop.

1.4 Find the suitable heap size setting

The JVM size should always be lower then the physical memory that the machine has. If it exceeds the limit, the swapping will cause the performances of JVM degrades heavily.

The following process can be used to determine the suitable heap size:

a. *Set the JVM start up and maximum heap size to the same starting value.*

b. Enable Verbose garbage collection setting from the administrative console for garbage collection monitoring

- *In the administrative console, click* ***Servers > Server Types > WebSphere application servers >*** *server_name.*
- *In the Server Infrastructure section, click* ***Java and process management > Process definition > Java virtual machine****.*
- *Ensure that the check box next to Verbose garbage collection is selected, and then restart the server.*

c. *Load the application with the expected user load and user scenarios.*

d. *Collect native_stderr.log file and analyze the GC (Garbage Collection) log entries.*

The optimal result for the average time between garbage collections is at least 5-6 times the average duration of a single garbage collection. If it is not achieved the application is spending more than 15 percent of its time in garbage collection. Then the rational step is to increase the heap size.

In the log entries, it will specify how long it takes for each GC process to complete. If the time is too long (normally should be within 2 seconds) for the GC, it usually means the heap size is too large.

Sometimes, because of the concurrent users in FIM, the heap is set to a high value and the GC will take every long or happen very frequently. In this case, tuning FIM and WAS session settings is needed to make it perform well. This will be discussed in later sections.

1.5 Recommendations

The recommended JVM heap size for FIM to run is 1 GB and it may need to change based on incoming request rate and session settings. This will be discussed further in later sections.

For more information on WAS JVM tuning, please see: http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/topic/com.ibm.websphere.nd.multiplatform.doc/info/ae/ae/container_tune_jvm.html

For technical guide on tune Java GC, please see:

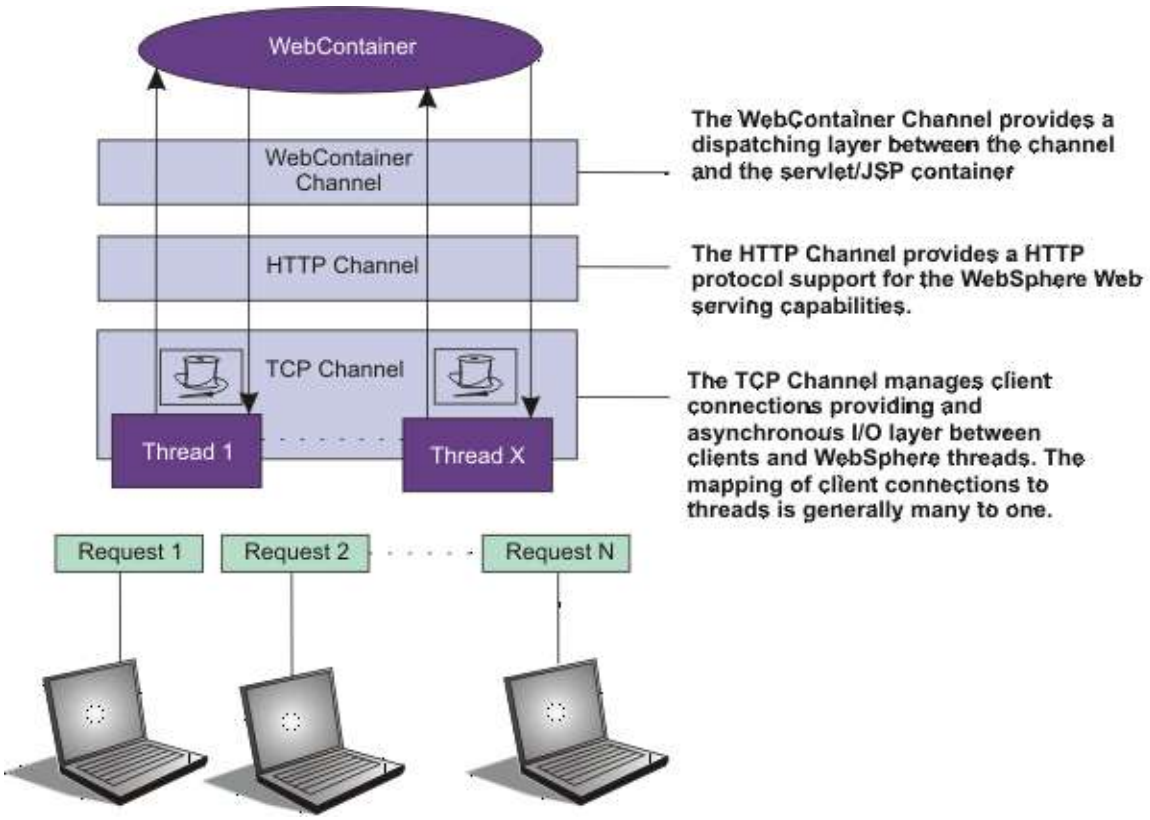
http://www.ibm.com/developerworks/library/i-gctroub/

Chapter 2: Tune WAS Transport channel services for FIM

The transport channel services manage client connections and I/O processing for HTTP and JMS (Java Management Services) requests. By applying suitable settings to each channel, the ability for FIM to handle concurrent requests can be improved and the usability also benefit from those tuning.

During installation, WAS will assign default values to each transport channel services. Changing the default values for settings on one or more of the transport channels associated with a transport chain can improve the performance of that chain.

Here is a figure from WAS showing the Transport Channel Service layout.



From FIM performance perspective, some tuning is recommended.

2.1 Adjust Default TCP channel setting

The major concern here is the **Maximum open connection** property. The value is set to 20000 by default, which is the maximum number of connections allowed. This should be changed to a lower value depending on the level of concurrency and application response time. High concurrency or long response time application will need a higher value of open connections. The recommended setting from Web Sphere tuning guide is 500.

In addition, long response time will also require tuning on the connection **Inactive timeout** setting. It needs to be changed to an enough long value to waiting for the application response and write it back to the client. The default in WAS is 60 seconds. Unless FIM is under unexpected extreme load, this is adequate for most cases.

To modify those two values:

a. *In the administration console, click **Servers > Application** **servers >** server_name **> Ports**.*

b. *Then click **View associated transports** for the appropriate port.*

c. *Select the transport chain whose properties you are changing. Click on the TCP transport channel defined for that chain.*

d. *Click **Apply** and then **Save** to save these changes.*

2.2 Adjust Default HTTP channel setting

The default HTTP transport channel setting in WAS is adequate for FIM to perform well. One option that is potential for performance boost is the HTTP keep-alive setting. The persistent (keep-alive) connection setting controls that whether connections are left open between requests. Leaving the connections open can save setup and teardown costs of sockets if your workload has clients that send multiple requests. The default value is true. When FIM is set up for STS (security token services) or some case in SPS (security and privacy services), the client only send single requests over a considerable long period of time. In this case, disable the option and close the connections right away is a better choice, rather than to have the HTTP transport channel setup the timeout to close the connection at some later time. However, it should be designed and tested carefully before doing so to avoid the false positive case.

To modify the setting:

a. *In the administration console, click **Servers > Application servers >**server_name **> Ports**. Then click **View associated transports** for the appropriate port.*

b. *Select the transport chain whose properties you are changing.*

c. *Click on the HTTP transport channel defined for that chain.*

d. *Click **Apply** and then **Save** to save these changes.*

2.3 Adjust Default Web container transport channel settings

The major purpose is to tune the Write buffer size parameter to a value such that most requests to be written out in a single write. The default value for the write buffer is 32768 bytes. This is sufficient for FIM in most cases. However, if FIM is configured such that the size is bigger then that, such as numerous attributes in SAML (security assertion markup language) messages, changing the value could introduce a better performance.

To modify the setting:

a. *In the administration console, click **Servers > Application servers >**server_name **> Ports**. Then click **View associated transports** for the appropriate port.*

b. *Select the transport chain whose properties you are changing.*

c. *Click on the Web container transport channel for that chain.*

d. *Click **Apply** and then **Save** to save these changes.*

2.4 **Adjust thread pool settings**

Each TCP transport channel is assigned to a particular thread pool. Thread pools can be shared between one or more TCP transport channels as well as with other components. The default settings for a TCP transport channel is to have all HTTP based traffic assigned to the ***WebContainer*** thread pool and all other traffic assigned to the ***Default*** thread pool. In certain environment, assigning a thread pool to http port can help isolate the network traffic and improve dealing with high concurrency requests.

Typical applications usually do not need more than 10 threads per processor. Default setting in FIM is adequate in the default setting. One exception is if there is some of server condition, such as a very slow backend request, that causes a server thread to wait for the backend request to complete. In such a case, CPU usage is usually low and increasing the workload does not increase CPU throughput. Thread dumps show nearly all threads in a call out to the backend resource. If this condition exists, and the backend is tuned correctly, try increasing the minimum number of threads in the pool until you see improvements in throughput and thread dumps show threads in other areas of the runtime besides the backend call. Another scenario that needs to tune the thread setting is when FIM is deployed in a WAS cluster environment. Additional thread setting will need to be tuned for dynamic cache synchronization and others. This will be discussed in detail in later sections about cluster environment tuning for FIM.

Minimum thread pool setting does not mean that those threads are initialized righter after the application server starts. Threads are added to the thread pool as the workload assigned to the application server requires them, until the number of threads in the pool equals the number specified in the Minimum size field. After this point in time, additional threads are added and removed as the workload changes. However the number of threads in the pool never decreases below the number specified in the Minimum size field, even if some of the threads are idle.

When facing the need to change the thread pool size associated with certain TCP channel, modify as following:

a. *In the administration console, click* **Servers > Application servers >** server_name **> Ports**. *Then click* **View associated transports** *for the appropriate port.*

b. *Select the transport chain whose properties you are changing.*

c. *Click on* **Thread pools >** threadpool_name *and adjust the values specified for the Minimum Size and Maximum Size parameters for that thread pool.*

d. *Click* ***Apply*** *and then* ***Save*** *to save these changes.*

For more information on WAS TCP setting, please see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Furun_chain_typetcp.html

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Ftprf_tunechain.html

For more information on WAS Thread settings, please see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Ftprf_tunetcpip.html

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.multiplatform.doc%2Finfo%2Fae%2Fae%2Fuejb_thrdpool.html

Chapter 3: Tune WAS Session Management for FIM

In some configuration, FIM will need to store certain session information in WAS. A typical example is FIM is configured as SAML2.0 SP (service provider). In those situations, it is important to adjust the session management settings in WAS to avoid performance bottlenecks.

The http session is store in WAS application server JVM. The resource, mainly heap memory, is shared by the applications in the server. The session information will take larger memory as more information is captured in the session data and more sessions come in to the server. If the sessions are not timed out or destroyed at log out, the memory available for use will decrease and limit the later operations on application objects. It will lead to frequently GC, which causes performance degradation, and eventually out of memory errors. In lab testing environment, it has been indicated that a SAML2.0 session data in FIM will occupy a memory size of 70k-100k, which is quite large.

3.1 Adjust default application server session settings

Two parameters in the session setting are particularly important from performance perspective, **Maximum in-memory count** and **Session timeout**. The default value for maximum in-memory count is 1000 and Allow overflow is checked. The default value for Session timeout is 30 minutes.

Maximum in-memory count has different meanings in different context. The meaning differs depending on whether you are using in-memory or distributed sessions. For in-memory sessions, this value specifies the number of sessions in the base session table for each web module. For distributed sessions, this value specifies the size of the memory cache for each web module's sessions. In the discussion of heap consumption, it refers to in-memory sessions.

Use the **Allow overflow** property to specify whether to limit sessions to this number for the entire session management facility or to allow additional sessions to be stored in secondary tables. It is used by default. There are several concerns to use this option. First, the session data is optimized for searching in primary table. Second, it will post the risk of having out of memory errors in the application server. If the allow overflow is not checked, When the session cache has reached its maximum size and a new session is requested, the session management facility removes the least recently used session from the cache to make room for the new one.

It is recommended that the maximum concurrent user in the server is carefully estimated and set the session count slightly higher then that value. With that, the Allow overflow can be unchecked to avoid further risks. However, the maximum concurrent user sessions in the environment is closely related to the Session timeout value.

The **Session timeout** value can be over write by the value in web modules, and it is use as default if the value is not specified in the modules. The value is not accurate to seconds. The SessionManager invalidation process thread runs every *x* seconds to invalidate any invalid sessions, where x is determined based on the session timeout interval specified in the session manager properties. For the default value of 30 minutes, *x* is around 300 seconds. In this case, it could take up to 5 minutes (300 seconds) beyond the timeout threshold of 30 minutes for a particular session to become invalidated. When you are estimating the maximum number of concurrent user sessions, this factor should be taken in to consideration.

To modify these settings to suit the environment, follow the following steps:

a. *In the administration console, click **Servers > Application servers >**server_name. Then click **Session management** for settings.*

b. *Specify a value for the **Maximum in-memory session count**.*

c. *Choose to check or uncheck **Allow overflow** option.*

d. *Specify a value for your Session timeout.*

e. *Click **Apply** and then **Save** to save these change*

3.2 **Using Application level for Session Management**

As stated in above section, session management in application server can be over written by Application level settings or Web Module settings. This is more appropriate when FIM is sharing the application server environment with some other applications which makes changing default application server settings not possible. The parameters for the application level session management are the same as the Application server level.

To use the enterprise applications level settings to override the setting, following these steps:

a. *In the administration console, click **Applications > Application Types >WebSphere enterprise applications**. Then click **ITFIMRuntime**.*

b. *Under **Web Module Properties**, click **Session management**.*

c. *Check **Override session management** box.*

d. *Change the values in the page according to the environment needs.*

e. *Click **Apply** and then **Save** to save these change*

3.3 Using Web Module level session management

In some occasions, customized session management for different web modules may need to be configured. One example would be differentiated session behaviors in STS and SPS services in FIM. From performance perspective, configure just right settings for different components in FIM production environment may help. However, this will involved slightly complex management effort.

To implement the Web Module level session management, following these steps:

a. *In the administration console, click **Applications > Application Types >WebSphere enterprise applications**. Then click **ITFIMRuntime**.*

b. *Under **Modules**, click **Manage Modules**.*

c. *Click on the module need to be changed.*

d. *Under **Additional Properties** click **Session Management**.*

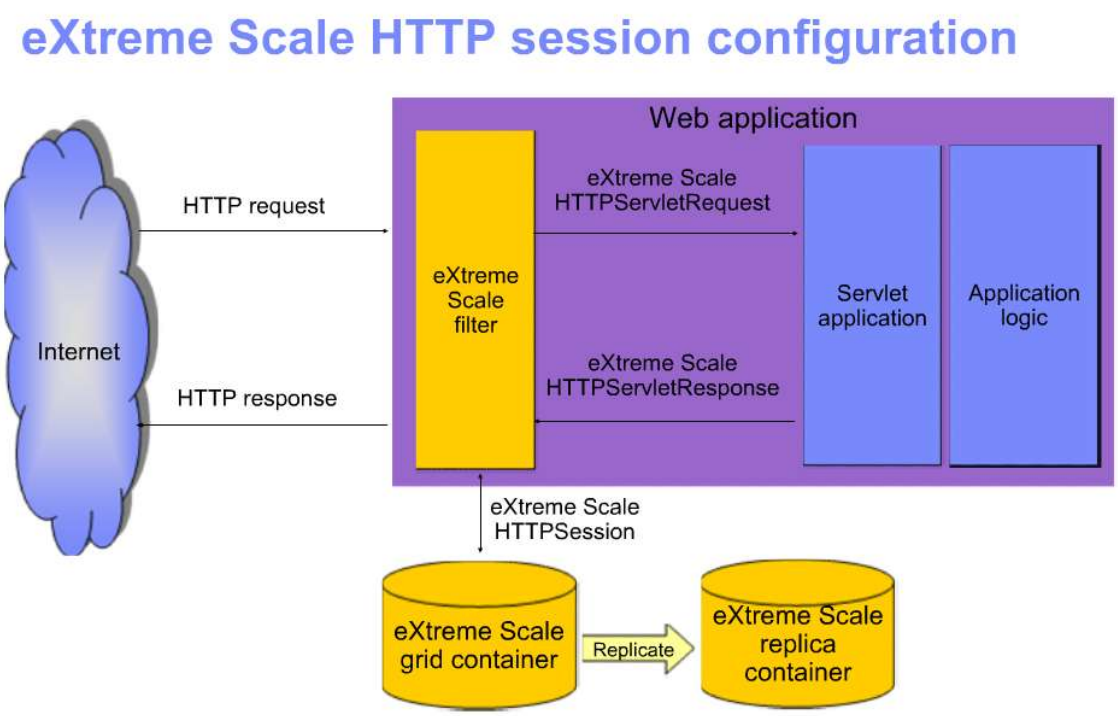
e. *Click **Override session management** box.*

f. *Change the values in the page according to the environment needs.*

g. *Click **Apply** and then **Save** to save these change*

3.4 Using advanced session management tools

When the session in-memory implementation becomes the bottleneck of performance after all the tuning possible, WAS provides alternative choices. The Distributed environment setting in Session management allows use distributed session to databases. An even more advanced integration for session management is available using WebSphere eXtreme Scale with DataPower.



For more information in WAS session settings, please see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v7r0/index.jsp?topic=%2Fcom.ibm.websphere.nd.iseries.doc%2Finfo%2Fiseriesnd%2Fae%2Fuprs_rsession_manager.html&resultof=%22%73%65%73%73%69%6f%6e%22%20%22%6d%61%6e%61%67%65%6d%65%6e%74%22%20%22%6d%61%6e%61%67%22%20

For more information of using eXtreme Scale with DataPower, please see:

http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.wxs/wxs/7.0/Administration/WXS70_HTTP_Session_Management/player.html

http://publib.boulder.ibm.com/infocenter/ieduasst/v1r1m0/index.jsp?topic=/com.ibm.iea.wdatapower/wdatapower/1.0/xc10/Administration/XC10_SessiondatagridConfiguration/player.html

For more information in WAS session trouble shooting, please see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r0/index.jsp?topic=%2Fcom.ibm.websphere.base.doc%2Finfo%2Faes%2Fae%2Frtrb_httpsessprobs.html

Chapter 4: Tune WAS Cluster Cache Replication for FIM

For high availability, load distribution and many other reasons, FIM could be deployed into WebSphere Network Deployment environment. In a cluster environment, cache replication is important when IHS (IBM HTTP Server), or other load balancing points, routs the request randomly among cluster members. This section will go over how to enable the caching and change the default settings in order for FIM to perform well.

4.1 Enable Dynamic Caching in WAS

In early versions of WAS (6.1 & 7), when the WAS cluster is created, a replication domain is created and assigned to the cluster by default. If it is not created because of version or additional replication domains are needed, follow the steps to create the replication domain first:

a. *In the administration console, click **Environment> Replication domains**. Then click **New…**.*

b. *Specify a **Name** for the replication domain.*

c. *Set the **Request timeout**.*

d. *Chose **Entire Domain** for **Number of replicas**.*

e. *Click **Apply** and then **Save** to save these change*

To enable the replication, do the same steps for all the cluster members:

a. *In the administration console, click **Servers > Server Types >WebSphere application servers**. Then click server_name.*

b. *Expand **Containers Services**. Then click on **Dynamic cache service**.*

c. *Check **the Enable cache replication**.*

d. *Use the replication domain just now created or the default as **Full group replication domain**.*

e. *Choose **Both push and pull** as **Replication** type.*

f. *Specify a value for **Push frequency**.*

g. *Click **Apply** and then **Save** to save these change*

4.2 **Dynamic Caching tuning from Application server level**

In previous step, the procedure to enable the dynamic caching already suggested a few parameters can be tuned for performance purposes. This chapter will discuss the **Cache size** and **Push frequency** settings.

The cache size setting in application server level limits the overall cache instances. If it is too low, the program may spend unexpected time to organize and coordinate the instances. However, if it is too high, the memory consumption is quite considerable. In a large scale environment, disk offload is required to make is sustainable. The small cache size ensures the synchronization is fast and neat. The large size ensures the capability for large scale and the speed, response time from user experience, may scarify a bit.

The push frequency is the same as the cache size. The more frequent you push, the faster the information is made available across the cluster, and the faster the response time can be. However, every time it pushes, some CPU is wasted if there is no much update or the update is not required. If the push frequency is set to 0 seconds, the application server converts the property value to the default value, which is 1.

Suggestion for the **Cache size** is that, after GC happened, there should be at least 40% of the JVM heap still available. If it cannot reach the number, try to increase the heap size, or make the cache object invalidation faster.

Suggestion for the **Push frequency** is that, it should be 1 second when the CPU load has not exceeded designed figure. If the setting is increased to higher value, FIM should be set to cater the possibility of needed objects not replicated yet. This is covered in following chapter in this section.

To change these two settings:

a. *In the administration console, click* ***Servers > Server Types >WebSphere application servers****. Then click server_name.*

b. *Expand* ***Containers Services****. Then click on* ***Dynamic cache service****.*

c. *Specify a value for* ***Cache size****..*

d. *Specify a value for* ***Push frequency****.*

e. *Click* ***Apply*** *and then* ***Save*** *to save these change*

4.3 Change specific Cache instance size

When FIM is deployed, it will create a set of default cache instance with default size (1000). The size can be changed depending on the services type and scale. To do so, following the steps:

a. *In the administration console, click* ***Resources > Cache instances >Object cache instances****. Then click on the cache instance name need to be changed.*

b. *Specify a value for* ***Cache size****.*

c. *Click* ***Apply*** *and then* ***Save*** *to save these change*

4.4 Thread pool tuning for Cache replication

When the replication is enabled, the thread pool that the replication service is using should be set to suite the need. For recent 6.1 and 7.0 fix pack levels DRS (data replication services) moved to its own thread pool and the size is already set. For early fix pack and version, the DRS is using the default thread pool. In this case, the minimum and maximum size for the thread poll should be at least 40. To do so, follow the steps:

a. *In the administration console, click* **Servers > Application servers >** server_name

b. *Under* ***Additional properties*** *Click on* ***Thread pools*****.**

c. *Click on **Default**, and adjust the values specified for the Minimum Size and Maximum Size parameters.*

d. *Click **Apply** and then **Save** to save these changes.*

4.5 Change Retry setting when using caching replication

As discussed in chapter 4.2, when the push frequency is set, there will be a delay for other cluster members in the cluster when a cache object is created. From performance perspective, the frequency may need to change to suite the different resource constrains. In this case, additional FIM parameters should be tuned to suit the need. There are two which is particularly important, **DistributedMap.GetRetryLimit** and **DistributedMap.GetRetryDelay**.

The default value for the retry limits is 2, and retry delay is 2 seconds. The retry control how many trial the wrapper will query the distributed map before returning that the data is not in the map. The delay controls the interval between each trial. These two settings will influence the potential CPU usage or waste and the response time to the request. Generally, the retry limits should not go beyond 5 and the delay is no to go beyond then the push frequency in the replication setting. The recommended value for the delay is 1 second or less.

To change the default setting for those two parameters, variable will be added in manually in FIM runtime properties. Following the steps:

a. *In the administration console, click **Tivoli Federated Identity Manager > Domain Management > Runtime Node Management**.*

b. *In the Runtime Nodes portlet, click **Runtime Custom Properties.***

c. *The Runtime Custom Properties panel is displayed. Click **Create**.*

d. *Enter a string in the **Name** field. Do not insert the space character in this field.*

e. *Enter a string in the **Value** field. Spaces are allowed in this field.*

f. *Choose one of the following actions:*

g. *Click **Apply** to apply the changes that you have made without exiting from the panel.*

h. *Click **OK** and load the configuration to Runtime.*

For more information on WAS dynamic cache, please see:

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Fudyn_rcachesettings.html

http://publib.boulder.ibm.com/infocenter/wasinfo/v6r1/index.jsp?topic=%2Fcom.ibm.websphere.express.doc%2Finfo%2Fexp%2Fae%2Ftdyn_dynamiccache.html

For more information in WAS performance tuning in Cache and Data Replication Service, please see:

http://www-01.ibm.com/support/docview.wss?uid=swg27006431

For more information on FIM customer properties, please see:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=%2Fcom.ibm.tivoli.fim.doc%2Ftfim611_cfg126.htm

Chapter 5: Tune IHS and WebSEAL with FIM

In FIM environment, IHS is commonly used to load balancing the communication with WAS cluster environment. IHS, or another HTTP server, export a central point for FIM cluster members and distributed the workload to those members. WebSEAL, which is a closely integrated authentication and authorization product with FIM, is commonly used as a Point of Contact Server for FIM. Using those two components can easily bring in a number of advantages, and, sometime, it is essential to use them. However, it may also introduce some performance issues to the environment, such as longer response time. In this section, a few check-up steps will be discussed to make sure the integration will not bring in any performance bottleneck.

5.1 **Identify and Tune IHS Performance Issues**

When IHS is used in the environment, there may be a few problems introduced, such as unresponsive behavior, slow response time and failed requests, etc. Those problems, in most cases, happen because of the concurrency settings and IHS server working with WebSphere IHS plug-in. The first thing is to identify the simultaneous connections the IHS will need to support. The typical way to determine the maximum number of simultaneous connections is to use mod_status reports.

To enable the report, follow the steps:

a. *Locate **http.conf** file for the HTTP server, and open with text editor.*

b. *Locate the **Loadmodule staus_module modules/mod_status.so** configuration stanza in the configuration file.*

c. *In **Allow from**, add the domain that will be used to view the status.*

```
# This example is for IBM HTTP Server 2.0 and above
# Similar directives are in older default configuration files.

Loadmodule status_module modules/mod_status.so
<Location /server-status>
SetHandler server-status
Order deny,allow
Deny from all
Allow from .example.com    <--- replace with "." + your domain
name
</Location>
```

d. ***Save** the changes and **Restart** IHS server.*

e. *From the domain that is added, access the IHS server server-status page (url assuming IHS server on ihs.exapmle.com http://ihs.example.com/server-staus/)* .

f. *Look for a line like the following:*

```
80 requests currently being processed, 70 idle workers
```

The concurrent requests that is being processed is the simultaneous requests the IHS is handling. If the number of idle workers is 0 or near to 0, the concurrency setting needs to be adjusted. The typical setting will need to change is the **MaxClients** and **ThreadsPerChild** configurations in **http.conf** file. For more information please see the links at the end of this section.

If a slow response time is still showing after fixing the concurrency setting, the next step is to indentify the delay happened at IHS or at the WebSphere Application server. IHS uses the WebSphere Web Server plug-in to communicate the application server. The mod_status can be used here again to get the diagnose information. In order to do so, an additional configuration needs to be changed in **http.conf** file:

```
# ExtendedStatus controls whether Apache will generate "full"
status
# information (ExtendedStatus On) or just basic information
(ExtendedStatus
# Off) when the "server-status" handler is called. The default is
Off.
#
LoadModule status_module modules/mod_status.so
<IfModule mod_status.c>
ExtendedStatus On
</IfModule>
```

When accessing the IHS server with the following url *http://ihs.example.com/server-staus/?showmodulestate (Assuming IHS on ihs.example.com),* it will show a snapshot of current requests status and associated modules. The information of the requests will look like this:

Srv	PID	Acc	M	Module	CPU	SS	Req	Conn	Child	Slot	Client	VHost	Request
0-0	4461	0/136/136	_		0.11	0	0	0.0	2.33	2.33	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1
0-0	4461	0/135/135	W	mod_was_ap20_http.c	0.03	3	0	0.0	2.31	2.31	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1
0-0	4461	0/135/135	W	mod_was_ap20_http.c	0.15	3	0	0.0	2.31	2.31	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1
0-0	4461	0/136/136	_		0.11	0	0	0.0	2.34	2.34	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1
0-0	4461	0/134/134	_		0.15	0	0	0.0	2.31	2.31	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1
0-0	4461	0/135/135	_		0.12	0	0	0.0	2.33	2.33	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1
0-0	4461	0/133/133	W	mod_was_ap20_http.c	0.10	3	0	0.0	2.29	2.29	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1
0-0	4461	0/135/135	W	mod_was_ap20_http.c	0.09	3	0	0.0	2.33	2.33	127.0.0.1	localhost.localdomain	GET /snoop/ HTTP/1.1

Columns with particular interest here are the **SS** and **Module**. **SS** is the time that has elapsed since starting of the request and the **Module** is the current processing unit of that request. If the module is ***mod_was_ap20_http.c*** or ***mod_app_server_http.c***, then the request is currently being processed by the WebSphere plug-in. Check the WebSphere plug-in log file for processing problems. If unable to determine if the application responds normally when connecting directly to WebSphere, perhaps due to application authentication requirements which are fulfilled by modules enabled in IBM HTTP Server, then the issue is most likely in WebSphere. If it only show when connection through HTTP server, please refer to more tunings for IHS server at the end of this section.

5.2 Tune WebSEAL as POC server

In lab testing environment, WebSEAL has proven to be reliable as sustainable working as POC (point of contact) server in FIM environment. Both the resource consumption (CPU time) and response time has seen improved by moving from WAS POC to WebSEAL POC server. It is highly recommended to use WebSEAL from performance perspective.

The default settings for WebSEAL are sufficient for it to perform well in most scenarios. The only concern is the worker threads starvation when working with FIM. If the user load is high, which introduces high concurrency, and the FIM server response is slow, it may lead to the starvation.

There are few ways to detect the work threads starvation. The easy way to detect worker threads starvation is to check the WebSEAL message log typically located at `/var/pdweb/log/msg__webseald-default.log`. When the soft or hard thread worker limit is reached, a message is generated in this message log. The soft and hard worker thread limits are controlled by the `worker-thread-hard-limit` and `worker-thread-soft-limit` stanza entries in the webseal configuration file, `/opt/pdweb/etc/webseald.conf` file. The unit of measure for these limits is percentage of total threads. A value of 100 indicates a message is generated when all worker threads are in use. When the number of active worker threads reaches the *hard* limit and the hard limit is not the maximum value of 100, in addition to generating a message, a 503, `Service Unavailable` error is returned to the requesting Web browser.

Other indication of worker threads starvation includes:

- Web browsers get timeout errors.
- Response times are high, yet WebSEAL server CPU utilization is low.

- Thread stack dumps indicate WebSEAL worker threads are blocked on receive and not on a mutex, meaning that they are waiting for work.

- TCP/IP packet traces indicate the number of active requests that WebSEAL is processing is equal to the number of worker threads.

Once the problem is indentified as worker thread, change the thread limit stanza using this formula:

```
(expected rate of requests per second) * (expected response time in
seconds per request from FIM)
```

However, as each additional worker thread will consume 450kb virtual storage and two TCP/IP sockets in webseal, increasing the number may lead to other performance issue. When the webseal process memory usage goes beyond the physical memory available on the machine, it may lead to slow response time because of paging, or cause thrashing and lead to server failure.

Another point to take care is the SSL and credential cache in webseal. When WebSEAL is used as POC server, it is usually used as SSL enforcement point and authentication point too. When it is the case, there may be a need to tune the SSL session cache and user credential cache because of the memory limitation. The SSL session cache uses about 250 bytes of process memory per entry and the credential cache uses about 7.5KB of process memory per entry. The following stanza entries in the `webseald.conf` are relevant to these caches:

```
[ssl]
ssl-v2-timeout = 100
ssl-v3-timeout = 7200
ssl-max-entries = 4096

[session]
max-entries = 4096
timeout = 3600
inactive-timeout = 600
```

The `ssl-max-entries` and `max-entries` stanza entries control the size of the SSL and credential caches.

For more information on IHS server performance issue identification and tuning, please see:

http://publib.boulder.ibm.com/httpserv/ihsdiag/ihs_performance.html

http://publib.boulder.ibm.com/httpserv/ihsdiag/WebSphere.html

For TAM WebSEAL tuning guide please see:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.itame.doc/am611_perftune.htm

Chapter 6: Tune Alias Service and DB connections

After IHS diagnose, the module status may show long SS (elapse time since request started) time with module ***mod_was_ap20_http.c*** or ***mod_app_server_http.c***. It is a likely indication of FIM slow response to IHS to finish the request. One possible reason, in SAML2.0 configuration or others, is because of the Alias service which is using LDAP (Lightweight Directory Access Protocol) or other Database connections.

Currently, there is no easy way to identify the Alisa or DB connection bottlenecks. The approach to identify, at this stage, is to enable the trace log for FIM in WAS, and analysis the log entries. It will show that the execution is waiting for LDAP response. Another way is to take javacore snapshot of WAS JVM, and look for waiting threads. If it show large number of thread in waiting status for LDAP or DB response, the service need to be tuned.

There are two places to look into when concerning about the services, connection pool size and DB operation time.

6.1 Setting Alias Connection Pool Size

When LDAP is used as the alias service and the connection pool is not sufficient to handle the concurrency, it will result in slow response time or failed request because of time out from the Point of Contact servers. The trace log from FIM will not show errors or exception as FIM is still waiting for backend services to finish, while the POC server gives time out error to user and fail the request. This is highly unexpected as the resource in FIM is wasted and user experience is damaged.

To avoid, make change to the connection size of FIM alias services to the sufficient large to support the concurrency. Follow the steps to make the change:

a. *In the administration console, click **Tivoli Federated Identity Manager > Domain Management > Alias Service Settings**.*

b. *When LDAP is used, it will display the configuration parameters for the LDAP server.*

c. *Choose from **LDAP hosts** to load the setting for the host.*

d. *Change the **Maximum Number of Connections** value.*

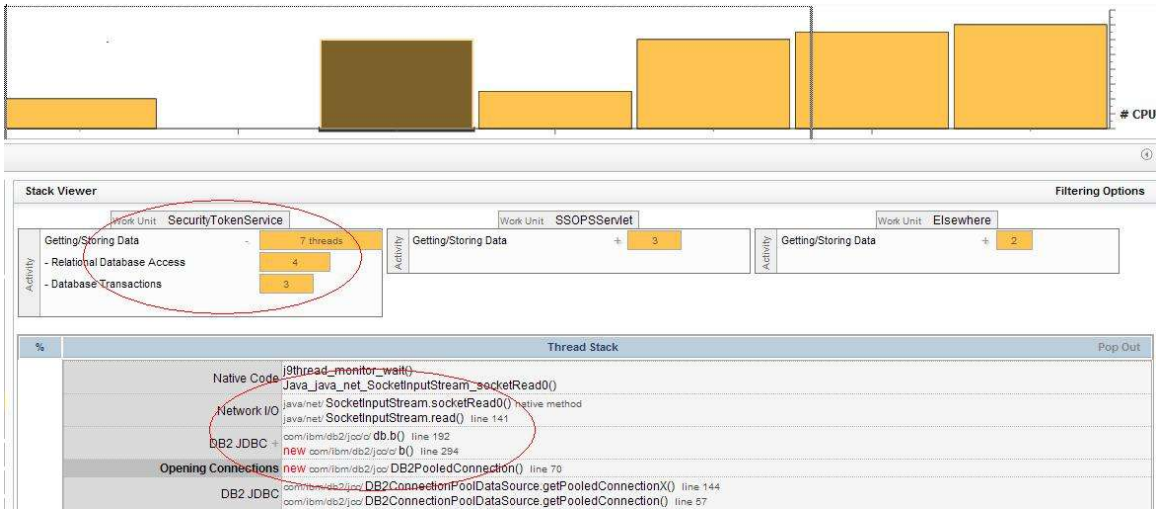
e. *Click **Apply** to apply the changes that you have made without exiting from the panel.*

f. *Click **OK** and load the configuration to Runtime.*

Recommended value for this number is at least half of the concurrent requests that the IHS module status has indicated (see Chapter 5.1), or half the live work threads in WebSEAL. However, this will be large dependent on the LDAP server response time or in some other case the DB response.

6.2 **Identify and Tune DB Response Time that FIM Uses**

By analyzing WAS java core snapshot files, it will give some insight where the slow response time could be. Here is an example of that showing that DB is responsible for the long response time (using IBM internal tool WAIT, other javacore analysis tool will also work):



It show that the JDBC connection to DB2 is open, but wait for response from DB, which indicates slow response time from the database.

Other tools of the same functionality are already available from IBM, such as IBM Thread and Monitor Dump Analyzer for Java. The link to the tools is provided at the end of this chapter.

When this happens, the DB need to be tuned for quicker response to FIM. If TDS (Tivoli Directory Server) or DB2 is used, here are some quick tips to try.

The first thing is to in crease buffer pool page size in DB2. To do so, follow these steps:

a. *Open **DB2 Control Center** using DB2 admin account.*

b. *Expand **All Databases**, right click on the database FIM is using and select **Configuration Parameters**.*

c. *Locate **BUFFPAGE** parameter in **Performance** section, and specify a value for it.*

d. *Click **OK** and **restart** the database.*

As starting point, the BUFFPAE value is 10% of the total memory the DB2 machine. Further analysis is to look into the buffer pool hit ratio and try to push the ratio to 95% or even 100% if possible. For more information of how to find the ratio and more tuning tips for DB2, please look at the links at the end of this section.

Another useful analysis is to look at DB2 dynamic SQL snapshot for the execution time the DB operations. More information is provide at the end if this section.

For more information related to Alias setting, please see:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/topic/com.ibm.tivoli.fim.doc/tfim611_cfg129.htm

For more DB2 Tunings, please see DB2 tuning guide from DB2 info center or see the link below:

http://www.ibm.com/developerworks/data/library/techarticle/anshum/0107anshum.html#sortheapsize

For TDS related DB2 tuning regarding slow response time, please see:

http://www.ibm.com/developerworks/tivoli/library/t-tds-perf2/

IBM Thread and Monitor Dump Analyzer for Java:

https://www.ibm.com/developerworks/mydeveloperworks/groups/service/html/communityview?communityUuid=2245aa39-fa5c-4475-b891-14c205f7333c

Chapter 7: FIM Tuning for SAML 2.0

When FIM is configured to perform SAML2.0 for single sign-on, FIM can act as either Service Provider or Identity Provider, or both. To tuning FIM to performance well for SAML2.0, here is the recommended approach:

a. *Identify the transaction rate the environment will need to handle.*

b. *Calculate the maximum concurrent user based on the transaction rate and session lifetime.*

c. *Adjust the JVM Heap size to suite the maximum number of users (see **Chapter 1**).*

d. *Adjust TCP Channel and Thread settings (see **Chapter 2**).*

e. *If LDAP is used as Alias database, increase the number of connections based on the transaction rate and tune LDAP server (see **Chapter 6**).*

f. *If slow response time behavior is observed, tuning IHS or WebSEAL POC server to suite the concurrency (see **Chapter 5**).*

7.1 Identify Session Data Size

Typically, SP (Service Provider) will need to maintain use session data for a period of time. This leads to high memory consumption in JVM heap. The session data size can be identified by analyze the JVM heap dump. To identify the session size:

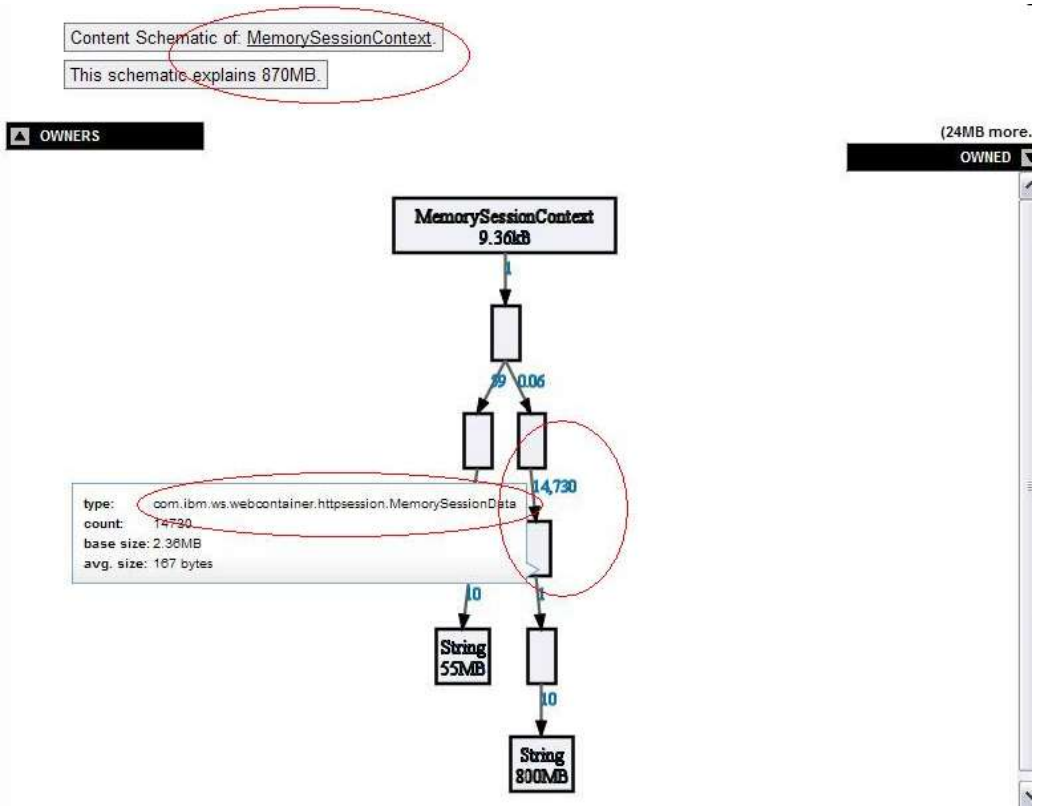
a. *Performance a few SSO via SAML2.0 to create some live session object in SAML SP.*

b. *Take a HEAP dump of the WAS hosting FIM SP.*

c. *Use Eclipse Memory Analyzer (or other heap analysis tools) to count .httpsession.MemorySessionData objects and the heap size it has been taken.*

d. *Calculate the size of a single session will take and use it to configure JVM HEAP size or adjust the life time of the session objects to suite the environment.*

Here is a sample result of the session data analysis in lab testing environment (using a different tool rather then Eclipse Memory Analyzer):



It shows that the session context is taking 870MB of heap, and there are 14,730 counts of the session object data. Therefore, each live session will consume 60kb+- heap memory in this configuration.

If the session has to be there for a long time and the transactions are exhausting the heap to out of memory error, the heap size need to be increased. Otherwise, considering reduce the lifetime for the session.

7.2 **Adjust Distributed Cache setting**

SAML2.0 keeps one distributed cache in WAS, SAML Single Sign-On Caches. When FIM is deployed on WAS cluster environment, it will degrade the response time greatly if the cache replication is not configured properly. This is even more obvious in SAML2.0 configuration, as SAML2.0 flow involves a few steps of message exchange between IP and SP to complete a single sign-on.

Unlike other cache instances, *itfim-samlsso* instance will need to be replicated faster as the time spent on this will reflected directly to user response time. A number of settings can be adjusted to improve user experience and performance. These settings include the **cache instance size**, **replication frequency**, **distributed map retry delay** and **distributed map retry limit** from FIM runtime.

The default cache size for *ifim-samlsso* is 1000. It is adequate for most cases. When artifact binding is used in SAML flow, the IP or SP stores the user information in this map and waits for the relying parties to retrieve the information. When it reaches it size limit, FIM frees up the space using longest inactive method. If the cache size is too small and currency is high, it may lead to unsuccessful authentication. If the cache size is too large, it will occupy too much memory and CPU resource for replication. To identify the needed cache size, use the following formula:

(Peak time Login Rate)(Expected Response time from partner in seconds) *1.2*

If the number is less then 1000, keep the default. If it is higher, increase it to the value from the calculation.

To modify the cache size for specific cache instances, follow the steps:

a. *In the administration console, click **Resources > Cache instances >Object cache instances**.*

b. *Click on **itfim-samlsso** to see the properties..*

c. *Specify a value for **Cache size**.*

d. *Click **Apply** and then **Save** to save these change*

For **replication frequency** setting, please read **Chapter 4.2** for detail guide.

For **distributed map retry delay** and **distributed map retry limit** from FIM runtime, please read **Chapter 4.5** for detail guide.

Chapter 8: FIM Tuning for OAuth

In latest FIM version (6.2.2), OAuth version 1.0a and 2.0 are both supported. One OAuth flow involves a few token exchanges between users, clients and OAuth servers. The tokens need to be cached in FIM, when FIM is performing the role of OAuth server. The default OAuth token caches mechanism implemented in FIM is distributed map implementation. Other customized cache implementation is also supported by FIM, such in-memory token cache and JDBC implementation to suit DB2 database. The token implementation configuration parameters will be discussed for both OAuth 1.0a and OAuth 2.0. After that, the benefits and concerns for each token implementation will be discussed to help identify the most suitable one based on environment.

8.1 **OAuth 1.0a Token Cache Tuning Parameters**

For FIM OAuth1.0a support, three configuration parameters will influence the live tokens in server.

- **Maximum allowable clock skew between OAuth server and client (seconds):**

  As the name suggests, this parameter controls the maximum allowable time difference between the OAuth server and the OAuth client.

  Typically the skew time is a small number. When it is set to big value, it would keep the token for too long which is not desired and non-value adding. However, when the server has many clients connect, and they are not NTP synchronized, there will be a need to allow some skewness. The best practice here is to identify the clients' time differences with server and set the value.

- **Temporary credentials and verification code lifetime (seconds):**

  The value controls the validity of the temporary credentials and verification code in seconds. Those temporary credential and verification codes are generated during the OAuth flow and will not need to maintain once the clients respond to server and finish the authentication flow. The value should be relatively small and it is influence by client's performance.

- **Maximum token credentials lifetime (seconds):**

The value controls the token credentials lifetime lapse in seconds. When this lifetime expires, the client cannot access the protected resource. The resource owner must reauthorize the client to access the protected resource. It will greatly depend on the clients' behavior to the resources. If the client only needs to retrieve the protected resource in short time spam, the value can be set to small. The risk is that, if the user request the resource frequently through the client and the value is small, multiple complete OAuth flow will be required and server load will be high. While when keep it large, the CPU resource will be saved as the server can reuse the authenticated credential. The concern for setting a high value is it may exhaust the memory of WAS server. They user and clients behavior should be studied and value can be updated based on the study.

To modify the values for the three configuration parameters, follow the steps:

a. *In the administration console, click **Tivoli Federated Identity Manager > Configure Federated Single Sign-on > Federations**.*

b. *Select Oauth1.0 Federation and click on **Properties…***

c. *Under **OAuth 1.0 Configuration Settings**, locate **Client Provider Configuration**.*

d. *Change the value for the three parameters discussed above.*

e. *Click **OK** and load the configuration to Runtime.*

8.2 OAuth 2.0 Token Cache Tuning Parameters

For FIM OAuth 2.0 support, three configuration parameters will influence the live tokens in server.

- **Maximum authorization grant lifetime (seconds):**

This value controls the maximum duration of a grant where the resource owner authorized the OAuth client to access the protected resource. This lifetime must be greater than the authorization code and access token lifetimes. When this lifetime expires, the resource owner must reauthorize the OAuth client to obtain an authorization grant to access the protected resource.

- **Authorization code lifetime (seconds):**

Specifies the maximum duration of a grant where the resource owner authorized the OAuth client to access the protected resource. This lifetime must be greater than the authorization code and access token lifetimes. When this lifetime expires, the resource owner must reauthorize the OAuth client to obtain an authorization grant to access the protected resource. These will need to be consistent with the how fast the client can response to the FIM server.

- **Access token lifetime (seconds):**

This value controls how long the client can use the access token to retrieve the protected resources once it is issued. When this lifetime expires, the OAuth client cannot use the current access token to access the protected resource. It will greatly depend on the clients' behavior to the resources. If the client only needs to retrieve the protected resource in short time spam, the value can be set to small. The risk is that, if the user request the resource frequently through the client and the value is small, multiple complete OAuth flow will be required and server load will be high. While when keep it large, the CPU resource will be saved as the server can reuse the authenticated credential. The concern for setting a high value is it may exhaust the memory of WAS server. They user and clients behavior should be studied and value can be updated based on the study.

To modify the values for the three configuration parameters, follow the steps:

a. *In the administration console, click **Tivoli Federated Identity Manager > Configure Federated Single Sign-on > Federations**.*

b. *Select Oauth2.0 Federation and click on **Properties…***

c. *Under **OAuth 2.0 Configuration Settings**, locate **Client Provider Configuration**.*

d. *Under **Authorization Grant Types Configuration**, specify the value for the first two parameters discussed above.*

e. *Under **Access Token Configuration**, Change the value for the third parameter discussed above.*

f. *Click **OK** and load the configuration to Runtime.*

8.3 **Token Cache Implementation Comparison**

As mentioned in the beginning of this chapter, there are a few token cache implementations to use in FIM for OAuth. The default supported implementation is distributed map implementation. In lab testing environment, in-memory token caches and JDBC implementation with DB2 are also tested. These three mechanisms all have their own benefits and concerns. The following table shows the comparison among those three.

	CPU load on FIM server	Memory consumption on FIM Server	Response time for OAuth flow
In-Memory	High	High	Fast
Distributed Map	High	Medium(can enable disk over-flow)	Medium
JDBC with DB2	Low	Low	Relatively slower than the other two

The in-memory and distributed map implementations is easy to configure and tune. However, when the user and client base is large and concurrency rate for the access is high, it will result to performance bottlenecks to CPU and memory easily. The JDBC with DB2 implementation on the other hand is scalable and can be extended in the future for increasing user and client base.

If In-Memory mechanism is implemented, please read **Chapter 1** to tune JVM heap size and GC process carefully for better performance.

If Distributed Map mechanism is used as default setting and FIM deployed in cluster environment, please read **Chapter 4** to tune cache replication.

If JDBC with DB2 mechanism is implemented, please read **Chapter 6.2** for more tuning tips.

Chapter 9: FIM Tuning for User Self Care

When FIM is configured for USC (User Self Care) functionalities, the performance can be improved by adjusting settings for several distributed caches.

User Self Care maintains three different distributed caches:

- Account Create Cache
- Forgotten Password Cache
- Secret Question Failure Cache

The caches are shared among WebSphere Application Server cluster members to permit a user operation to be properly handled. This sharing is required in case different phases of the operation take place on different nodes.

9.1 USC Cache Instance Overview

User Self Care uses the WebSphere Distributed Object Cache technology for implementation of the caches. See the WebSphere Application Server documentation for details on this caching technology.

There are two parameter types that affect each User Self Care distributed cache:

Entry lifetimes

These parameters are set in the User Self Care response file. Cache entries are retained until either the lifetime is hit or the user finishes the operation requiring the cache entry. The names and settings of these cache-specific parameters are described in the individual cache tuning descriptions later in this set of topics.

Cache sizes

These parameters are set in the administrative console by accessing ***Resources*** > ***Cache Instances*** > ***Object Cache Instances***. The Cache size parameter controls

how many concurrent entries are retained in the cache. The names and settings of these cache-specific parameters are described in the individual cache tuning descriptions

The caches must be sized adequately so users can perform operations that require a distributed cache in the configured time period. If a cache is too small, users might not be able to validate their accounts or recover their passwords during the specified time period. The time period in the configuration for lifetime of the cache entries can be adjusted according to the environment.

For example, to give users two minutes to finish an account recovery validation, configure the entry lifetime for the account recovery validation cache to be two minutes. If expecting two users per second to perform an account recovery operation, set the account recovery validation cache to at least 240.

Determine the appropriate size using the following calculation:

120 seconds x 2 users/second = 240

The default size of the account recovery validation cache is 1000 entries. This default would be adequate for this example. Other operations, such as account creation, might require an increase in the cache size.

Depending on expected system usage, the size of one or more caches might need to be increased. This adjustment can affect hardware requirements. Cache entries take up memory and must be replicated between systems in the cluster.

9.2 Cache Instance and Parameters

Account create cache

This cache stores the data from the user inputs during the account creation and e-mail validation process. When the user finishes the validation, the User Self Care recovers the data from the cache to create an account in the registry.

Parameter	Description
AccountCreateLifetime	Specifies the amount of time, in seconds, that User Self Care recognizes the account creation request as valid, and retain the request in the internal cache.
itfim-usc_accountcreate	Cache size is controlled by the `itfim-usc_accountcreate` cache size.

Unlike other operations, each account creation operation creates two cache entries. One entry is consists only of the user ID and a key. The second entry consists of all the data that the user enters in the account creation form.

You configure cache entry lifetimes to be 120 seconds. You expect a peak number of users enrolling during a new application provisioning operation to be 10 each/second. You might want to size your cache as follows:

10 users/second x 2 entries/user x 120 seconds/entry = 2400 x 20% buffer ~= 3000.

Forgotten password cache

This cache stores the user ID during the Forgotten Password validation operation.

Parameter	Description
AccountRecoveryValidationLifetime	Specifies the amount of time, in seconds, that User Self Care considers the account validation request to be valid.
itfim-usc_forgottenpassword	Cache size is controlled by the `itfim-usc_forgottenpassword` cache size. This entry is small, consisting essentially of the user ID and a key.

Secret question failure cache

This cache stores the number of failed secret question answer attempts that have been performed.

Parameter	Description
AccountRecoveryFailureLifetime	Specifies how long, in seconds, the program retains record of an unsuccessful account validation attempt.
itfim-usc_secretquestionfailures	Cache size is controlled by the `itfim-usc_secretquestionfailures` cache size. This entry is consists only of a number and a key.

9.3 Notes for Tuning Cache Instance

Configuration of WebSphere Application Server operations can improve tuning of the caches.

- Replication

  WebSphere does not automatically replicate all of the cached data between nodes. Instead, it just replicates the keys between nodes and only retrieves the data when requested by a particular node. If a key is requested on a particular node system that is not found in the cache, User Self Care attempts the cache lookup operation. The attempt provides time for WebSphere Application Server to finish any possible replication.

- Cache flushes

Restarting WebSphere Application Server clears caches and returns them to a clean state.

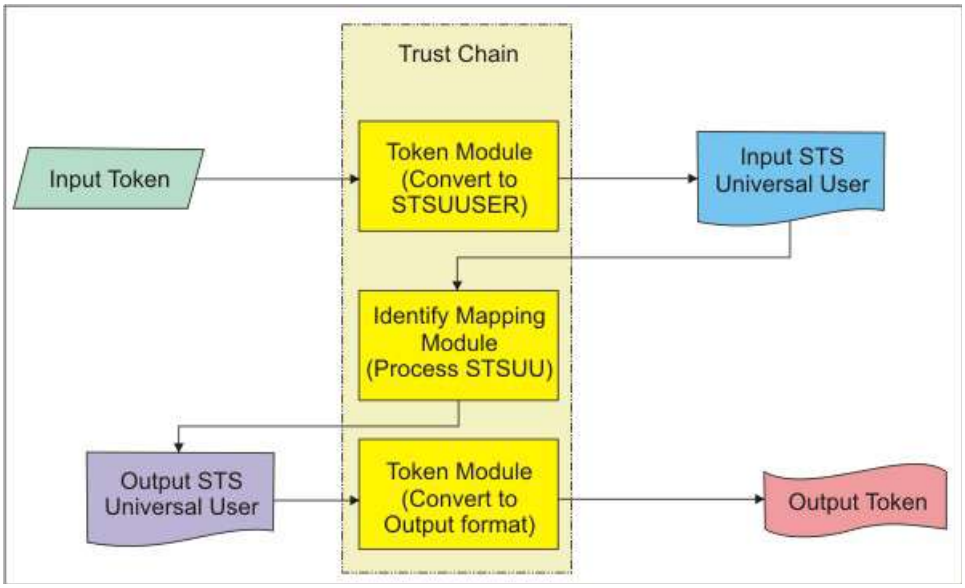
- Removal of User Self Care caches

  Cache entries are retained until either the entry lifetime is hit or the user finishes the operation requiring the cache entry.

Chapter 10: FIM Tuning for STS Services

Unlike other services type, when FIM is configured for Security Token Services (STS), the major performance bottleneck is often CPU other then memory.

In lab testing environment, it has shown that the identity mapping rules module used in STS can influence the CPU consumption. In STS services, typical setting is that an incoming token is validated and then translated to a STSUU token, and then the STSUU is translated to the token needed to issue. During the process, mapping rules is used for the token translation, or XML translation. FIM support two types of mapping rules for xml transformation currently, XSL and Java Script. Java Script mapping rules is relatively less CPU consumptive.



10.1 **Modify STS Identity Mapping Rule**

To modify STS services identity mapping rule, follow the steps:

a. *In the administration console, click **Tivoli Federated Identity Manager > Configure Trust Service > Trust Service Chains**.*

b. *Select the STS Chain need to modify and click on **Properties…***

c. *Under **Trust Service Chain Modules**, select the **Module Instance** that is in charge of Identity Mapping, then click **Properties…**.*

d. *Click on **Modify Rule…***

e. *Under **XSL or JavaScript file Containing Identity Mapping Rule Local Path**, **Browse…** to the file.*

f. *Click on **Import File**, the content should be should load up to **Identify Mapping Rule** panel.*

g. *Click **OK** and load the configuration to Runtime.*

10.2 Develop Identify Mapping Rules

When FIM is installed, there are a few samples available for mapping rules both in xsl and js format. XSL mapping rule samples can be found under folder ***FIM_Install_root/examples/mapping_rules*** and JS mapping rules can be found under folder ***FIM_Install_root/examples/js_mappings***.

To develop customer identity mapping rules, please read FIM configuration guide, or visit:

http://publib.boulder.ibm.com/infocenter/tivihelp/v2r1/index.jsp?topic=%2Fcom.ibm.tivoli.fim_bg.doc_6.2.1%2Fconcept%2Fconfig%2Foutside%2Fcustomidentitymap.html