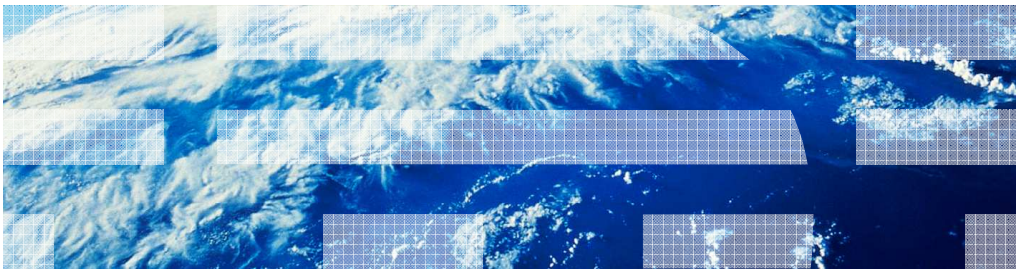
CICS Transaction Server V4.2

Connectivity enhancements for HTTP



This module describes the connectivity enhancements for HTTP that are introduced in CICS® Transaction Server Version 4.2.

Table of contents

CICS has been enhanced to provide new functionality in two areas to manage HTTP connections. The first area is HTTP outbound connection pooling, which is a technology that allows processor usage to be reduced when multiple transactions all attempt to connect to the same server throughout a day. The second area is inbound connection throttling, which has been designed to help prevent an individual CICS region from becoming overloaded in a z/OS® environment that uses a port sharing mechanism.

HTTP outbound connection pooling

- Reuse of connections for outbound HTTP requests in or across tasks
- Reuse connections that have the same properties, as defined by a URIMAP resource
- SOCKETCLOSE timeout option on client URIMAP
  - Nonzero value manes HTTP requests using that URIMAP can use connections (sockets) from a pool
    - Applies to any HTTP requests using the same client URIMAP
    - No code changes needed to benefit, except when using the CICS web interface
    - Benefits HTTP event processing adapter
  - SOCKETCLOSE is timeout for length of time socket remains available for reuse in the pool
  - Socket is removed from pool if errors are returned or if any problems are detected

HTTP outbound connection pooling is a technology used widely in networking and middleware applications. It allows a connection to be reused multiple times in an individual task or across multiple tasks.

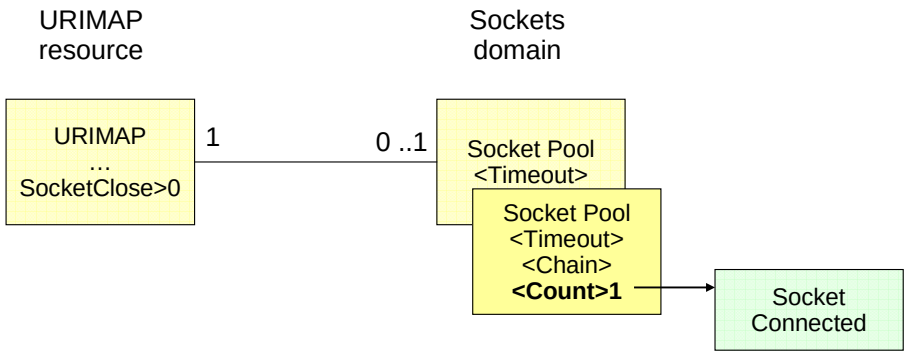
CICS TS 4.2 now enables both new and existing applications that meet certain requirements to exploit this new capability. The requirement on an application is that it must use a URIMAP resource to identify the endpoint to which it sends its request. The URIMAP must also have the SOCKETCLOSE attribute set to a nonzero value. The SOCKETCLOSE value on the URIMAP resource defines how long a connection is kept in the pool of connections for reuse. This means that existing applications that already use a URIMAP to define their endpoint require no code changes to exploit HTTP outbound connection pooling. Instead you can just update the URIMAP definition for the application.

To prevent the pool containing connections that are no longer viable, there are several situations that cause a connection to be either removed from the pool or placed in the pool after use. These are the connections being closed by the server, the length of time that a connection being in the pool exceeds the SOCKETCLOSE value, the last response received on the connection was not ok, or the MAXSOCKETS value of the region has been reached and so a connection is closed so that there is a socket available for use in another connection. The connections in a pool are also removed when the URIMAP resource associated with that pool is disabled.

SOCKETCLOSE attribute

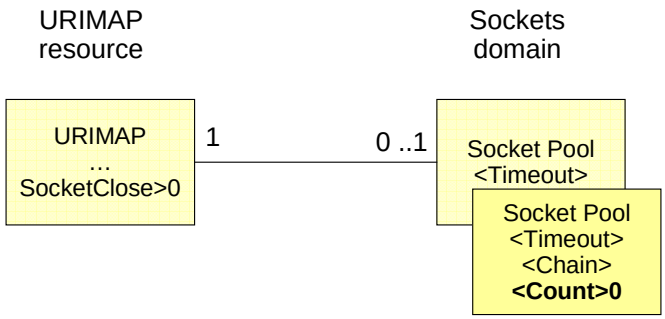
This screen capture from the CICS Explorer™ shows an example URIMAP definition. The highlighted field is the new SOCKETCLOSE attribute, which for this example is not set and therefore uses the default value. The default value means that the behavior of the URIMAP resource is the same as previous releases of CICS TS, so no outbound connection pooling is performed.

HTTP connection pooling scenario

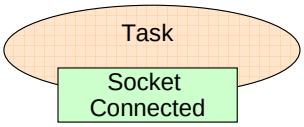


The next few slides describe how a connected socket from the pool gets used and returned to the pool. In this scenario a URIMAP resource has been defined and installed with a SOCKETCLOSE value greater than zero. This in turn means that a socket pool exists for that URIMAP. There might be other socket pools in the CICS region, but there is only ever a maximum of one socket pool associated with each URIMAP. In the pool of connected sockets there is one available for reuse by another task.

HTTP connection pooling scenario



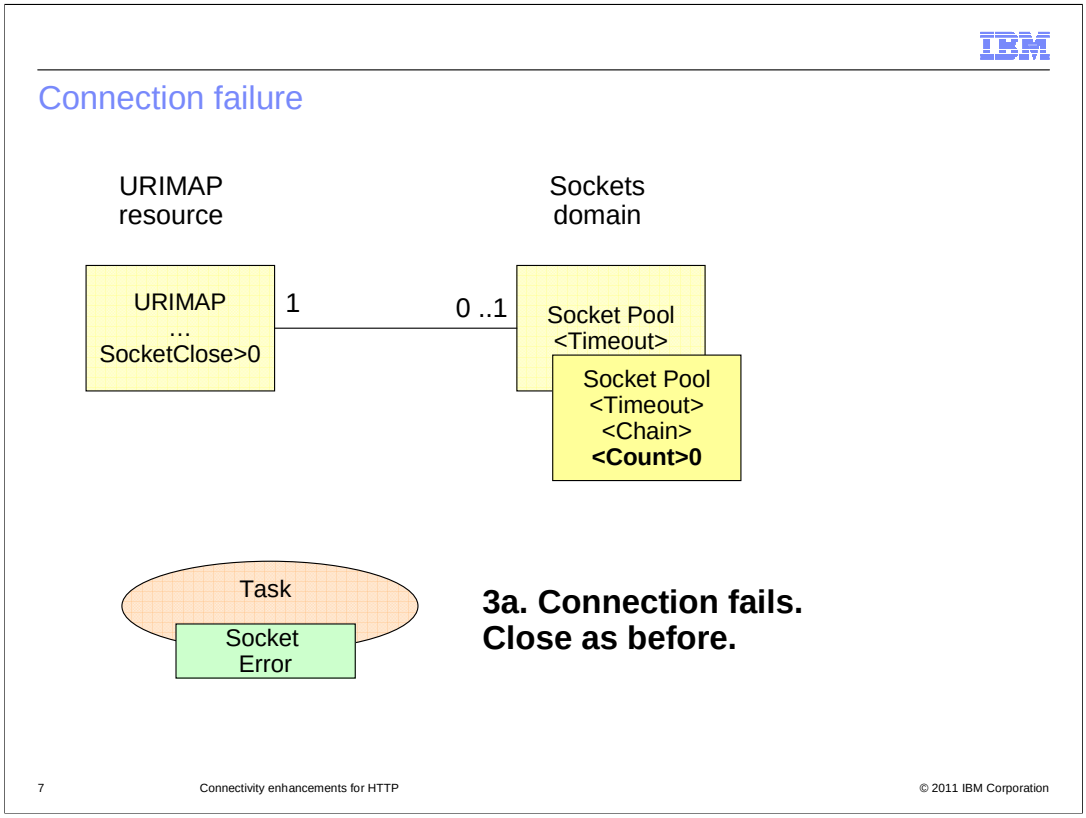
1. Create a connection



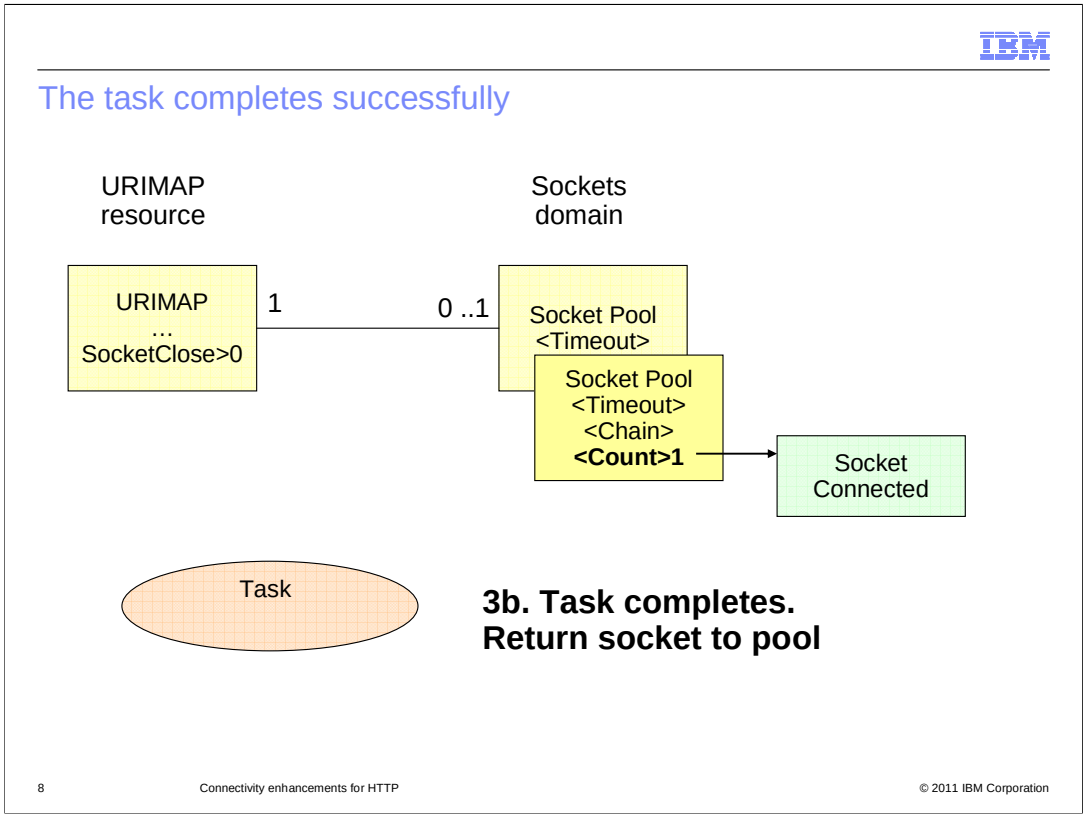
2. Socket owned by task. No longer in pool

6 Connectivity enhancements for HTTP

A new task has created a connection to a server using the URIMAP resource to specify the endpoint. In previous releases of CICS, a new socket is created at this point and a connection established. However, because HTTP outbound connection pooling has been configured for this URIMAP, some of that overhead is saved because a previously created and connected socket is removed from the pool and associated with the task instead. After the connected socket is associated with the task, it is treated in the same way as a connection in previous versions of CICS and is no longer associated with the pool.



When using HTTP connections, it is possible that an error might occur. For example, the server might close the connection prematurely or a network outage might occur. In these scenarios, putting the socket back into the pool after the task has finished using it is counter productive and so it gets closed and discarded.



In most cases, it is unlikely that a problem will occur that causes a connected socket to be closed and discarded. In this case, when an application makes an EXEC CICS call to close the connection, instead of closing it CICS returns the connected socket into the pool it came from and disassociates the task from it. At this point the connected socket is available for another task to reuse it.

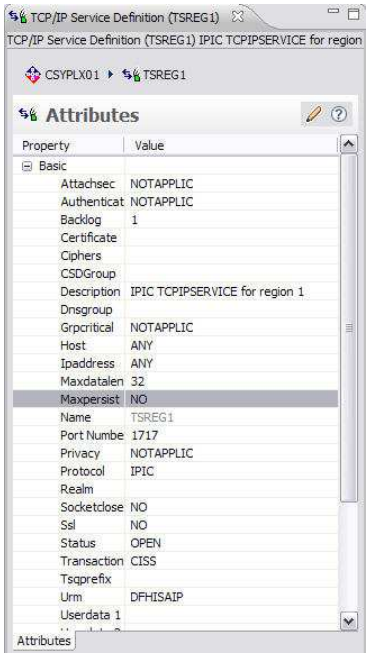
HTTP inbound connection throttling

- Option to limit the number of persistent connections from web clients that are allowed on a port (at any one time)
- MAXPERSIST attribute on TCPIPSERVICE
  - Number of persistent connections
  - Socket closed if number exceeded
  - Default is NO
- Provides a way to throttle incoming requests without leaving them queueing

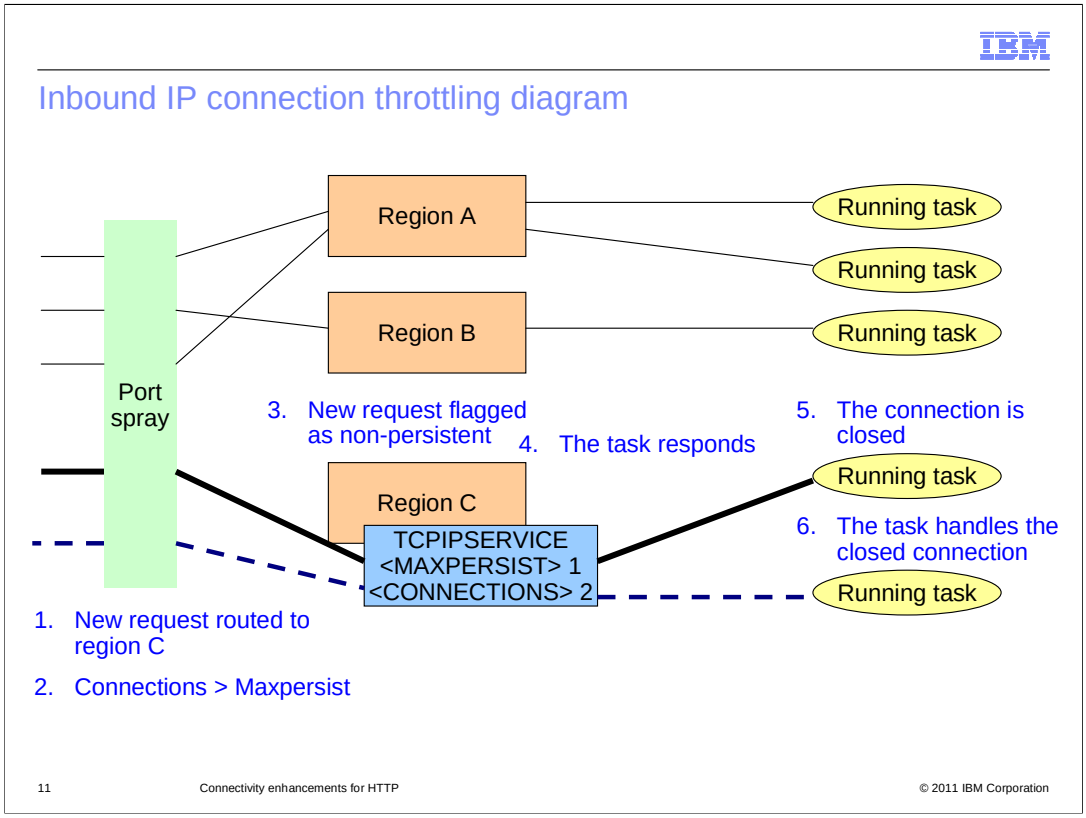
HTTP inbound connection throttling provides the ability to limit the number of concurrent, persistent connections that a CICS region can receive on specific ports. You can enable this feature by setting the MAXPERSIST attribute on a TCPIPSERVICE resource definition before installing it. The MAXPERSIST attribute sets a threshold, above which any new connection made on that port is closed after a response is sent. This forces a client to reconnect if they want to make further requests to a service. The default setting is NO so that CICS behaves as it has in previous releases.

Do not use this setting in a single region environment, because it increases the network overhead and processing done by the CICS region. In a multi-region environment, where a z/OS port sharing mechanism is set up to allow multiple regions to transparently provide the same service, this setting can help prevent an individual region from becoming overloaded.

MAXPERSIST attribute



This screen capture shows an example TCPIPSERVICE resource in the CICS Explorer. The highlighted field is the new MAXPERSIST attribute, which for this example has the default value of NO specified. This means that no inbound connection throttling occurs in the CICS region.



This diagram describes how the new inbound IP connection throttling works. The port sharing layer routes requests to one of three CICS regions, A, B, or C. The solid black lines show requests that have come in, have been routed to one of the three CICS regions and are being processed as normal. In region C, the TCPIPSERVICE resource contains a MAXPERSIST value of 1. This value is not recommended but is low to make it easier to explain the concept in the diagram. Region C already has one connection denoted by the solid black line.

The blue dashed line shows a new task and helps to describe how a CICS region that already has the same number of connections as the MAXPERSIST value handles it. Firstly, a new request hits the port sharing layer and gets routed a region, in this case region C. Secondly, region C notices that the new connection takes the total number of connections above the MAXPERSIST value. Thirdly, it flags the new connection as one that is not persisted but still allows the request to be processed as normal. In steps 4 and 5, after the task has finished processing the request and has sent a response, region C closes the connection.

If the client needs to make further requests it must attempt to reconnect. In this situation, the task hits the port sharing layer again. It is likely be routed to one of the other regions that is not at its MAXPERSIST threshold and so can maintain a persistent connection with that region. If the client is routed back to the region that is at its threshold, the same processing of the request occurs. The more times a client attempts to reconnect in a correctly configured system, the smaller the chances of not being routed to a region with capacity.

Feedback

Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send email feedback:

mailto:iea@us.ibm.com?subject=Feedback_about_CICSTS42_HTTP_connections.ppt

This module is also available in PDF format at: ../CICSTS42_HTTP_connections.pdf

You can help improve the quality of IBM Education Assistant content by providing feedback.

Trademarks, disclaimer, and copyright information