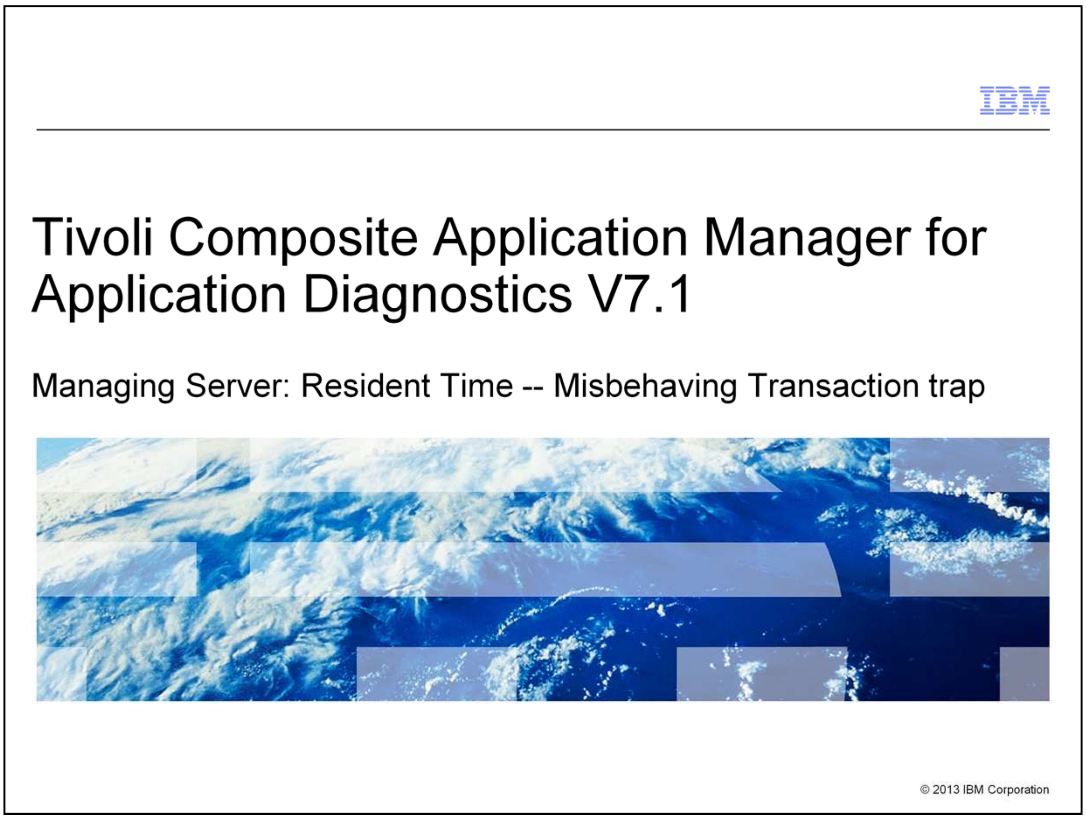
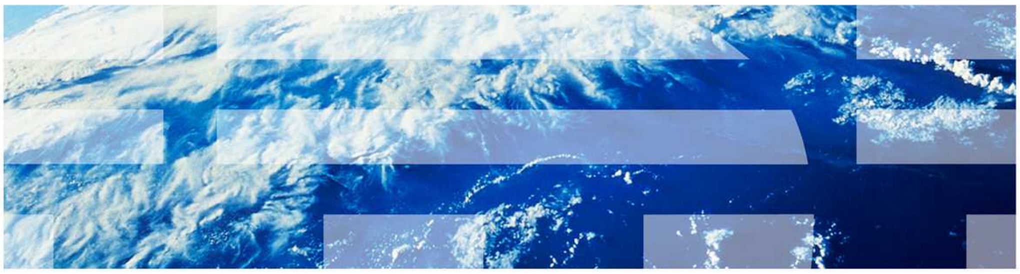


Tivoli Composite Application Manager for Application Diagnostics V7.1

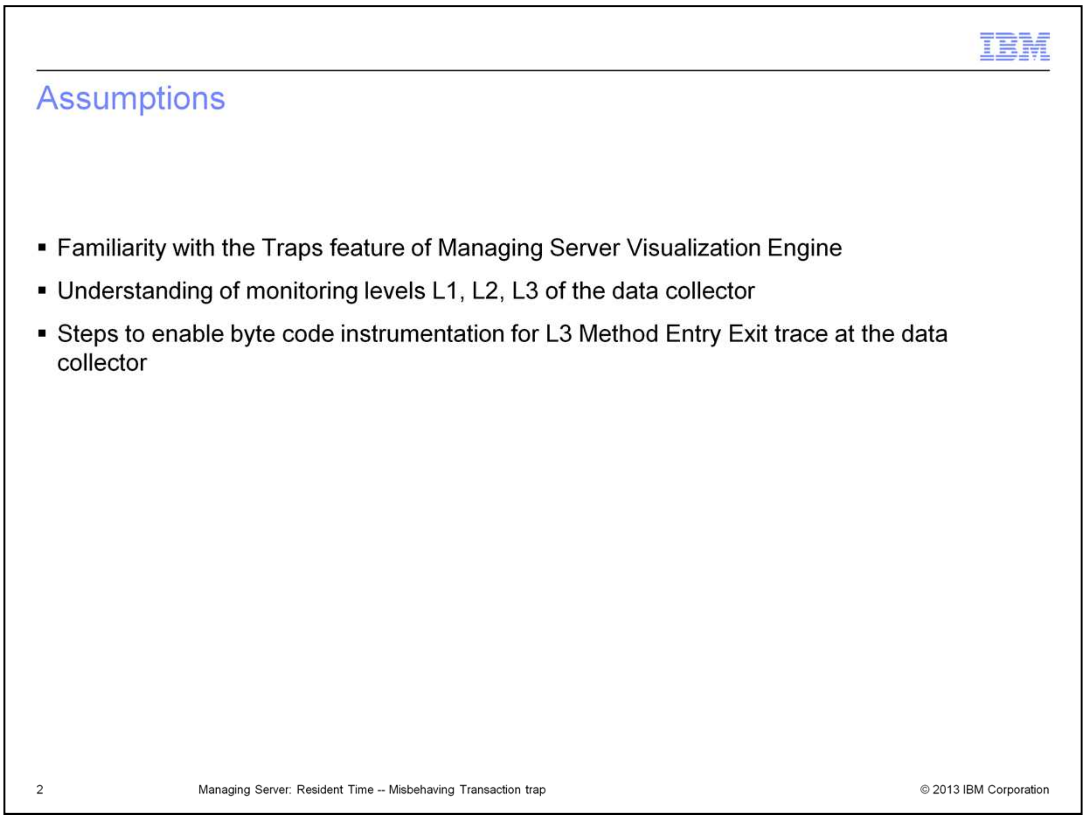
Managing Server: Resident Time -- Misbehaving Transaction trap



Managing Server is a deep-dive diagnostics provider for the Tivoli® Composite Application Manager for Application Diagnostics V7.1.

You can define traps in the Managing Server to get alerts and to collect diagnostic data, when monitored performance data exceeds a specific threshold.

The *Resident Time, Misbehaving Transaction trap* is a specialized trap that you can use to collect a full method level trace of a Java request that is executed on a Java 2EE application server.



Assumptions

- Familiarity with the Traps feature of Managing Server Visualization Engine
- Understanding of monitoring levels L1, L2, L3 of the data collector
- Steps to enable byte code instrumentation for L3 Method Entry Exit trace at the data collector

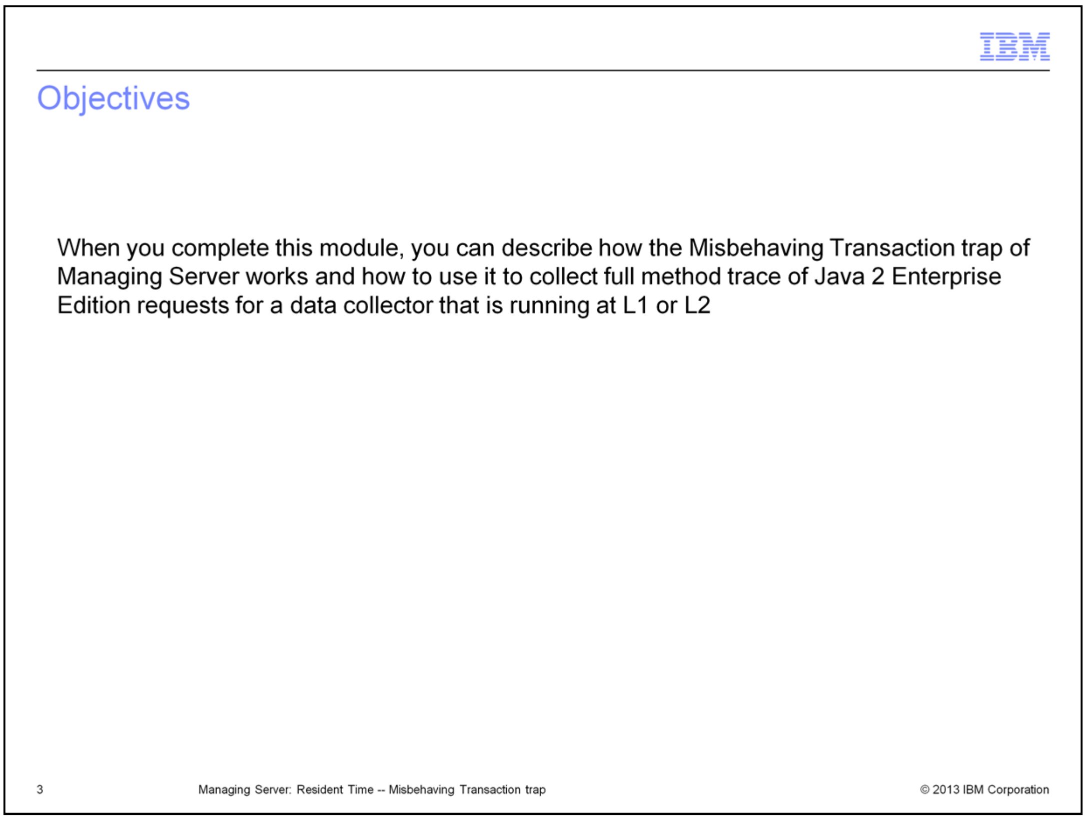
2 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

The module developer assumes that you are familiar with these concepts:

- Traps functionality of Managing Server Visualization Engine

- Monitoring levels L1, L2, and L3 of data collector

- Steps to enable Method Entry Exit Byte Code Instrumentation (BCI) at the data collector

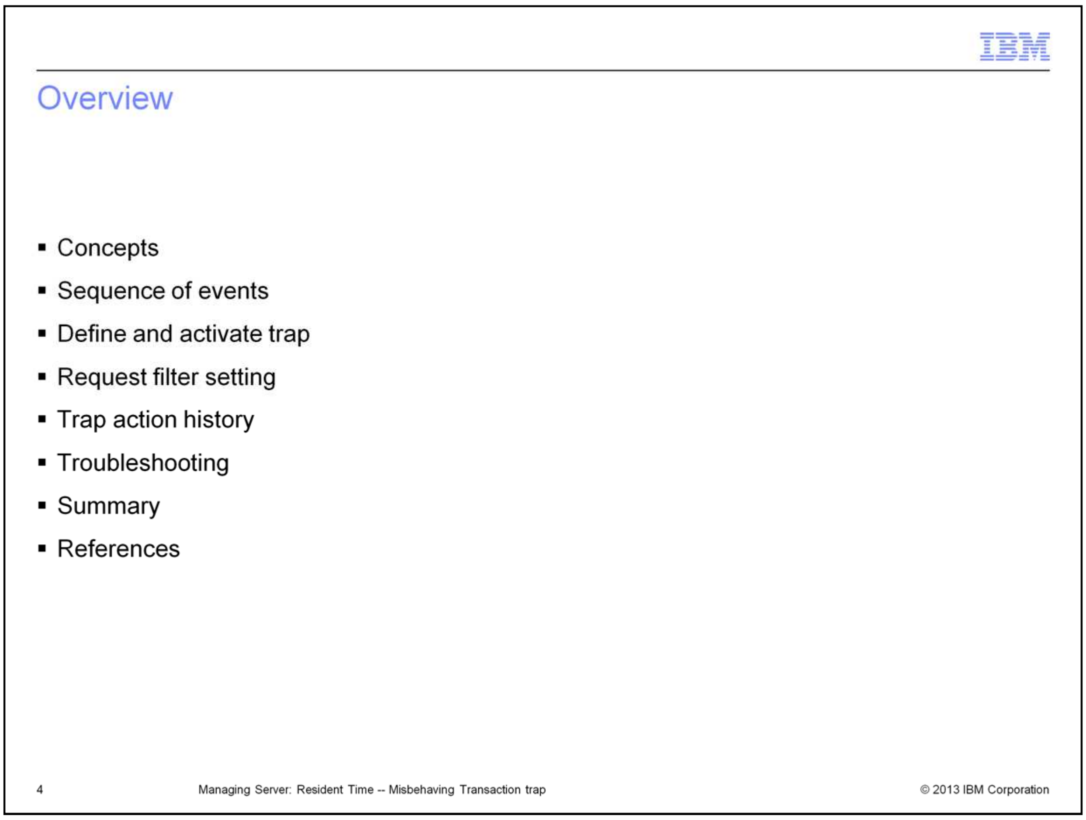


Objectives

When you complete this module, you can describe how the Misbehaving Transaction trap of Managing Server works and how to use it to collect full method trace of Java 2 Enterprise Edition requests for a data collector that is running at L1 or L2

3 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

When you complete this module, you can describe how the Misbehaving Transaction trap of Managing Server works and how to use it to collect a full (Level 3) method trace of requests, even if the data collector for the application server is running at monitoring Level 1 or Level 2.



Overview

- Concepts
- Sequence of events
- Define and activate trap
- Request filter setting
- Trap action history
- Troubleshooting
- Summary
- References

4 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

This module presents these topics:

- Concepts that are related to Misbehaving Transaction trap

- Sequence of events that are initiated by the trap, which result in a full method trace collection

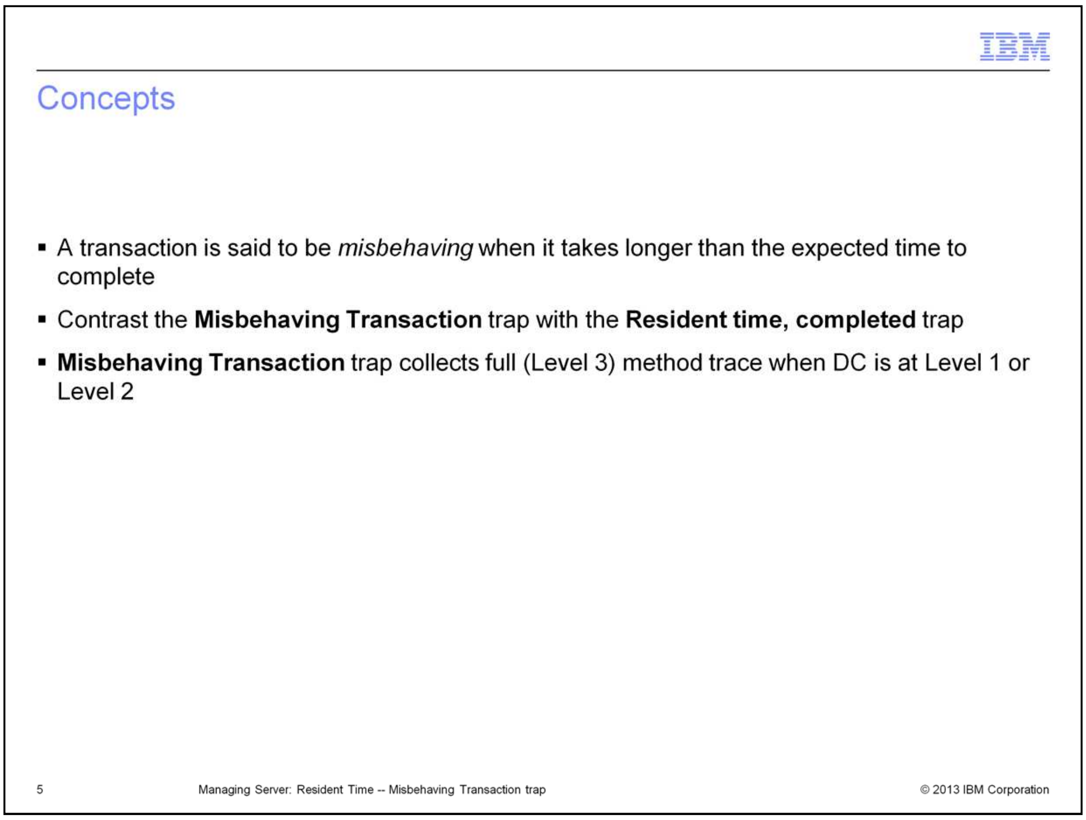
- Steps for defining and activating the trap

- How to use the *Request filter* in a Trap definition

- How to use the **Trap Action History** window to view trap action results, such as method trace

- Some common troubleshooting tips if the trap does not collect the full method trace

- References to related documentation



Concepts

- A transaction is said to be *misbehaving* when it takes longer than the expected time to complete
- Contrast the **Misbehaving Transaction** trap with the **Resident time, completed** trap
- **Misbehaving Transaction** trap collects full (Level 3) method trace when DC is at Level 1 or Level 2

5 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

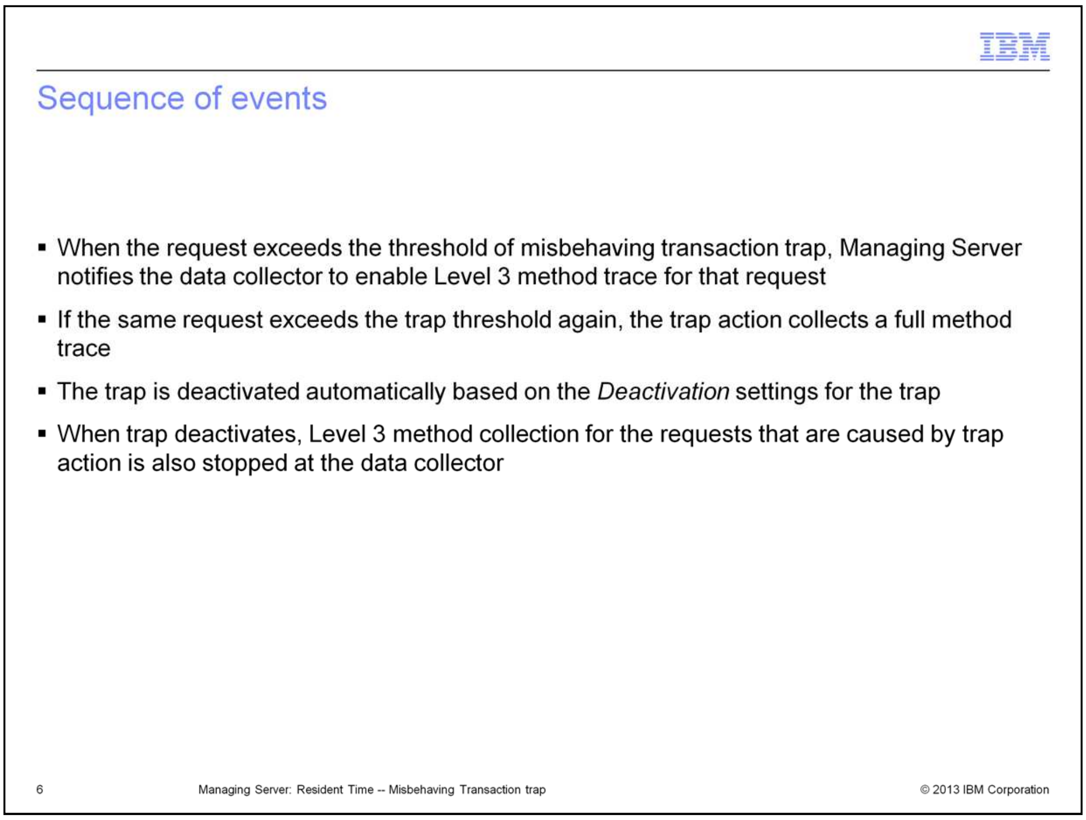
The main concepts for this module are as follows:

The terms *transaction* and *request* are used interchangeably to refer to a Java request executed in an application server JVM.

A transaction is said to be *misbehaving* when it takes longer than expected time to complete; this expected time is the threshold of trap definition.

Another trap, **Resident time - Completed** can also be used to collect method trace for long running requests, but this trap collects a full Level 3 method trace only if the data collector is running at monitoring Level 3. This makes it impossible to capture a full method trace in production with **Resident time - Completed** trap, because in production, enabling the L3 monitoring level has a significant performance impact.

You can use the **Resident time - Misbehaving Transaction** trap to collect a full (Level 3) method trace for requests even if the data collector is running at monitoring levels L1 or L2, which makes the Misbehaving Transaction trap a very effective tool in troubleshooting application performance.



Sequence of events

- When the request exceeds the threshold of misbehaving transaction trap, Managing Server notifies the data collector to enable Level 3 method trace for that request
- If the same request exceeds the trap threshold again, the trap action collects a full method trace
- The trap is deactivated automatically based on the *Deactivation* settings for the trap
- When trap deactivates, Level 3 method collection for the requests that are caused by trap action is also stopped at the data collector

6 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

The sequence of events that leads up to full method trace collection by the Misbehaving Transaction trap are listed:

- When request completion time exceeds the threshold of a Misbehaving Transaction trap, Managing Server notifies the data collector to enable Level 3 method trace *for that request.*

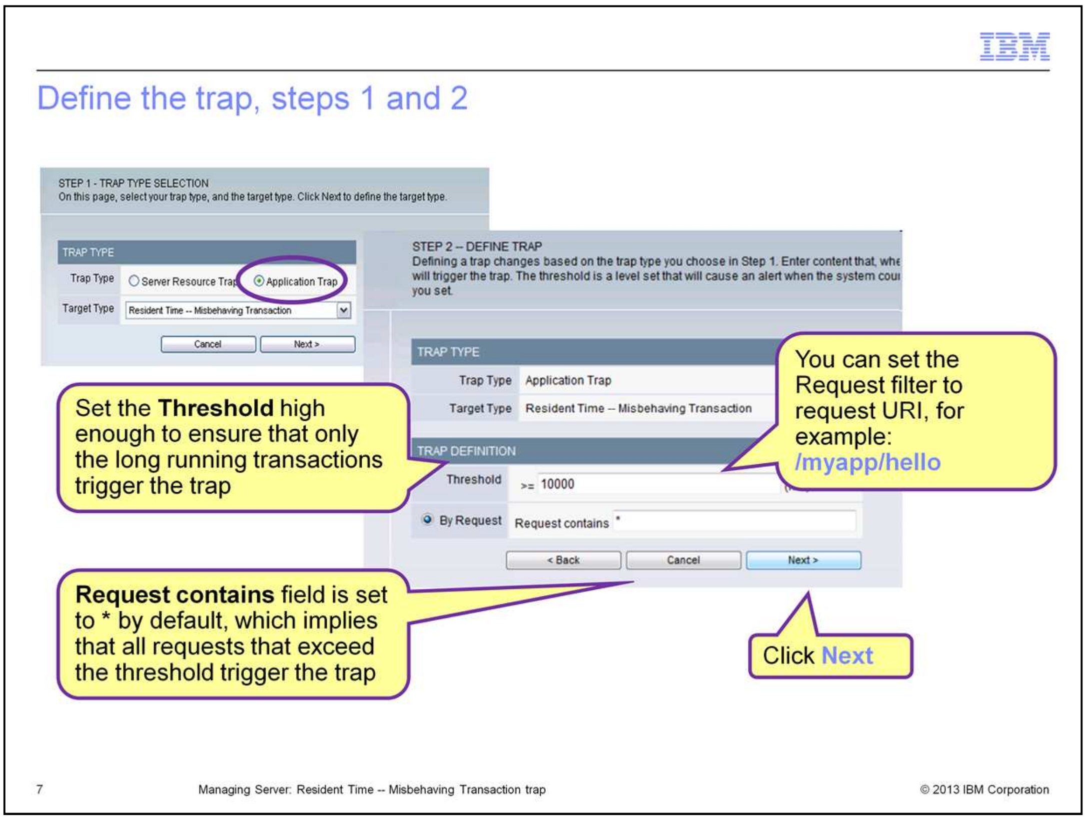
- If the same request exceeds the trap threshold again, a method trace is collected for that request.

- Since the Level 3 method trace was enabled when the trap first triggered, the collected method trace is a full method trace.

- Thus a minimum of two occurrences of the trap are required to collect the method trace.

- During trap activation, *Deactivation* settings can be specified to automatically deactivate the trap.

- When the trap deactivates, Managing Server notifies the data collector to stop L3 method trace collection for the request.



Define the trap, steps 1 and 2

STEP 1 - TRAP TYPE SELECTION
On this page, select your trap type, and the target type. Click Next to define the target type.

TRAP TYPE

Trap Type	Server Resource Trap [✓ Application Trap]
Target Type	Resident Time -- Misbehaving Transaction

STEP 2 -- DEFINE TRAP
Defining a trap changes based on the trap type you choose in Step 1. Enter content that, whe
will trigger the trap. The threshold is a level set that will cause an alert when the system cou
you set.

TRAP TYPE

Trap Type	Application Trap
Target Type	Resident Time -- Misbehaving Transaction

TRAP DEFINITION

Threshold	>= 10000
● By Request	Request contains *

Cancel
Next >
< Back
Cancel
Next >
Set the **Threshold** high enough to ensure that only the long running transactions trigger the trap
You can set the Request filter to request URI, for example: /myapp/hello
Request contains field is set to * by default, which implies that all requests that exceed the threshold trigger the trap
Click Next

7 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

A. To create the trap, select menu option **Problem Determination** >**Trap & Alert Management**. From the Trap and Alert Management window, select the **Create Trap** option.

B. On STEP 1 window, select **Application Trap** type and then **Resident Time – Misbehaving Transaction** target type, and click **Next**.

C. On STEP 2 window, specify the **Threshold** for the trap, which is time in milliseconds. This threshold should be set to a high enough value so that only slow response times can trigger the trap.

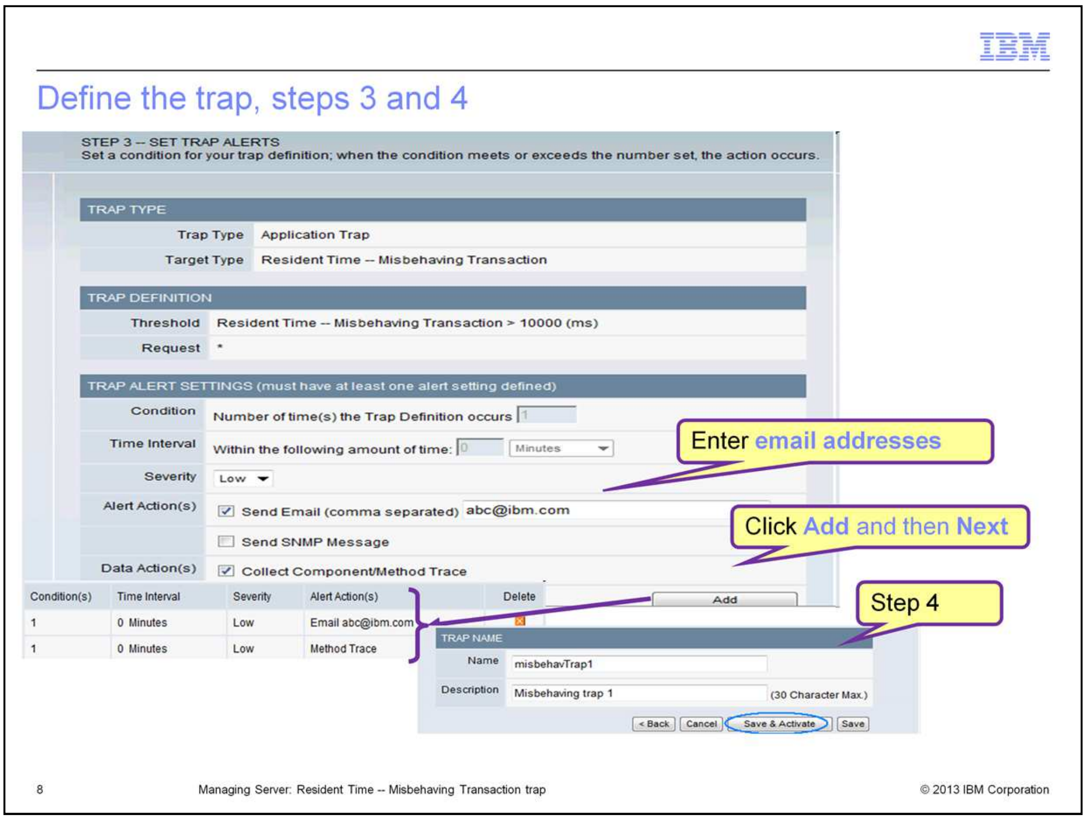
1) The **Request contains** field on this window is set to ***** by default. The setting of * implies that all requests that exceed the threshold will trigger the trap.

2) Note that the single ***** is the only character which is interpreted as a regular expression in this field.

3) For example, if you enter /myapp/* as request filter, it will *not* cause the trap to trigger for request /myapp/hello.

4) To capture a method trace for a specific request, use the request URI (Uniform Request Identifier) or a unique substring of the URI in the request filter.

D. Click **Next** on STEP 2 window.



Define the trap, steps 3 and 4

STEP 3 -- SET TRAP ALERTS
Set a condition for your trap definition; when the condition meets or exceeds the number set, the action occurs.

TRAP TYPE

Trap Type	Application Trap
Target Type	Resident Time -- Misbehaving Transaction

TRAP DEFINITION

Threshold	Resident Time -- Misbehaving Transaction > 10000 (ms)
Request	*

TRAP ALERT SETTINGS (must have at least one alert setting defined)

Condition	Number of time(s) the Trap Definition occurs 1
Time Interval	Within the following amount of time: 0 Minutes
Severity	Low
Alert Action(s)	✓ Send Email (comma separated) abc@ibm.com
	Send SNMP Message
Data Action(s)	✓ Collect Component/Method Trace

Condition(s)	Time Interval	Severity	Alert Action(s)	Delete
1	0 Minutes	Low	Email abc@ibm.com	☒
1	0 Minutes	Low	Method Trace	

TRAP NAME

Name	misbehavTrap1
Description	Misbehaving trap 1 (30 Character Max.)

On STEP 3 window of trap definition, specify actions for the trap.

A. Select the check boxes for **Alert Action** and **Data Action**.

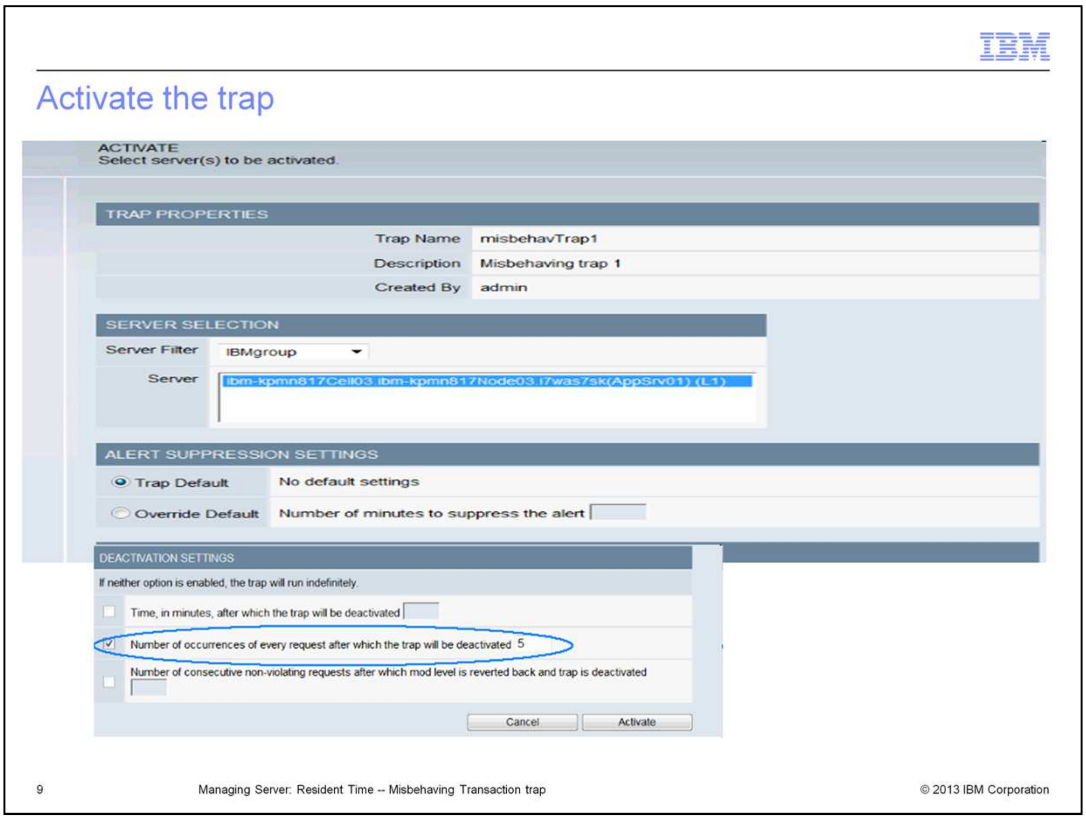
B. Alert action options are **Email** and **SNMP message**. These actions send an alert notification when the trap triggers.

C. The fields **Condition** and **Time Interval** under Trap Alert Settings are disabled for this trap.

D. The only data action available for this trap is to collect method trace.

E. Click the **Add** button to add the two actions to the trap definition and then click **Next**.

On the STEP 4 window, enter **Trap name** and **description**, then click **Save & Activate**.



Activate the trap

ACTIVATE
Select server(s) to be activated.

TRAP PROPERTIES

Trap Name	misbehavTrap1
Description	Misbehaving trap 1
Created By	admin

SERVER SELECTION

Server Filter	IBMgroup
Server	ibm-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk(AppSrv01) (L1)

ALERT SUPPRESSION SETTINGS

● Trap Default	No default settings
○ Override Default	Number of minutes to suppress the alert

DEACTIVATION SETTINGS

If neither option is enabled, the trap will run indefinitely.

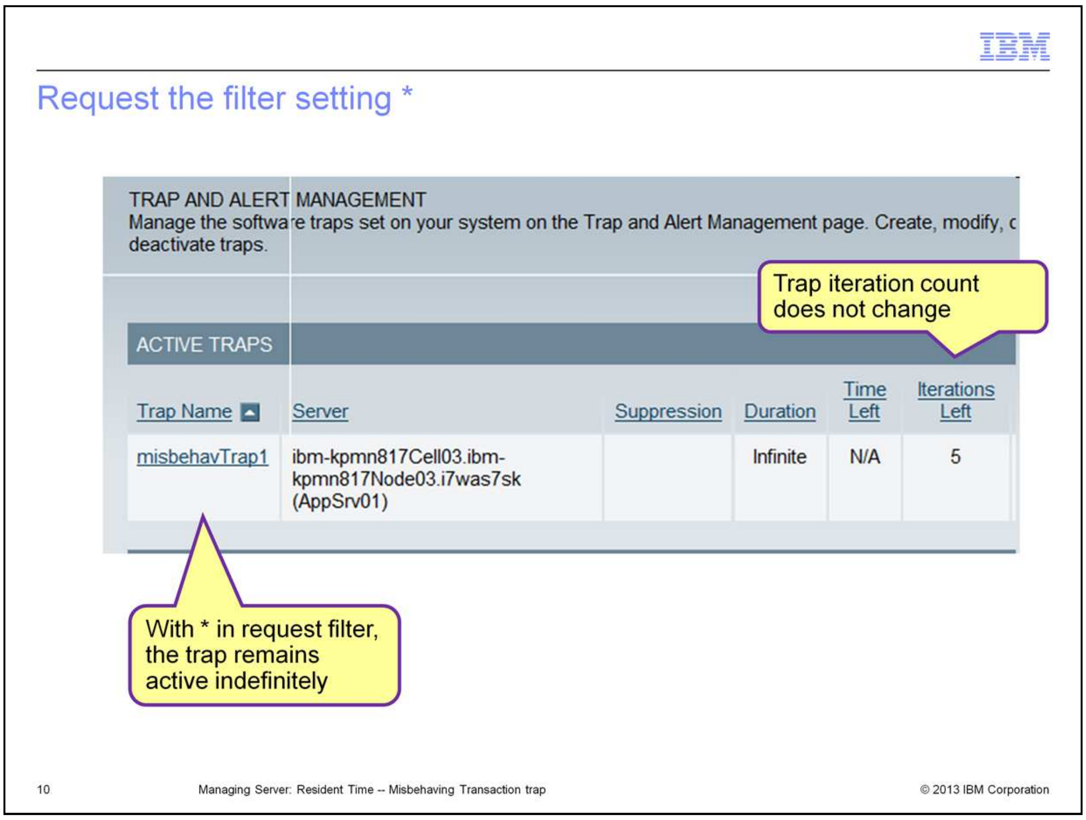
- [ ] Time, in minutes, after which the trap will be deactivated
- [✓] Number of occurrences of every request after which the trap will be deactivated 5
- [ ] Number of consecutive non-violating requests after which mod level is reverted back and trap is deactivated

A. On the **Activate** window, select the data collectors on which to activate the trap.

B. There are three check boxes for **Deactivation Settings**. By default, the setting for **Number of occurrences after which the trap will be deactivated** is set to 5.

C. This setting results in up to four method traces which is sufficient in most cases. You can increase or reduce this count to collect more or fewer method traces.

A minimum setting of **2** is required to collect the method trace because the first occurrence of trap is used to notify the data collector to enable L3 method trace for the request.



Request the filter setting *

TRAP AND ALERT MANAGEMENT
Manage the software traps set on your system on the Trap and Alert Management page. Create, modify, c
deactivate traps.

ACTIVE TRAPS

Trap Name	Server	Suppression	Duration	Time Left	Iterations Left
misbehavTrap1	ibm-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk (AppSrv01)		Infinite	N/A	5

Trap iteration count does not change
With * in request filter, the trap remains active indefinitely

10 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

You should know the details of what to expect when request filter is set to *** .**

For this example, assume that you want the trap deactivation count to be set to **5**.

If **request1** triggers the trap, Level 3 method trace is enabled for it. After the trap triggers five times for **request1**, four method traces are collected and the trap deactivates for **request1**.

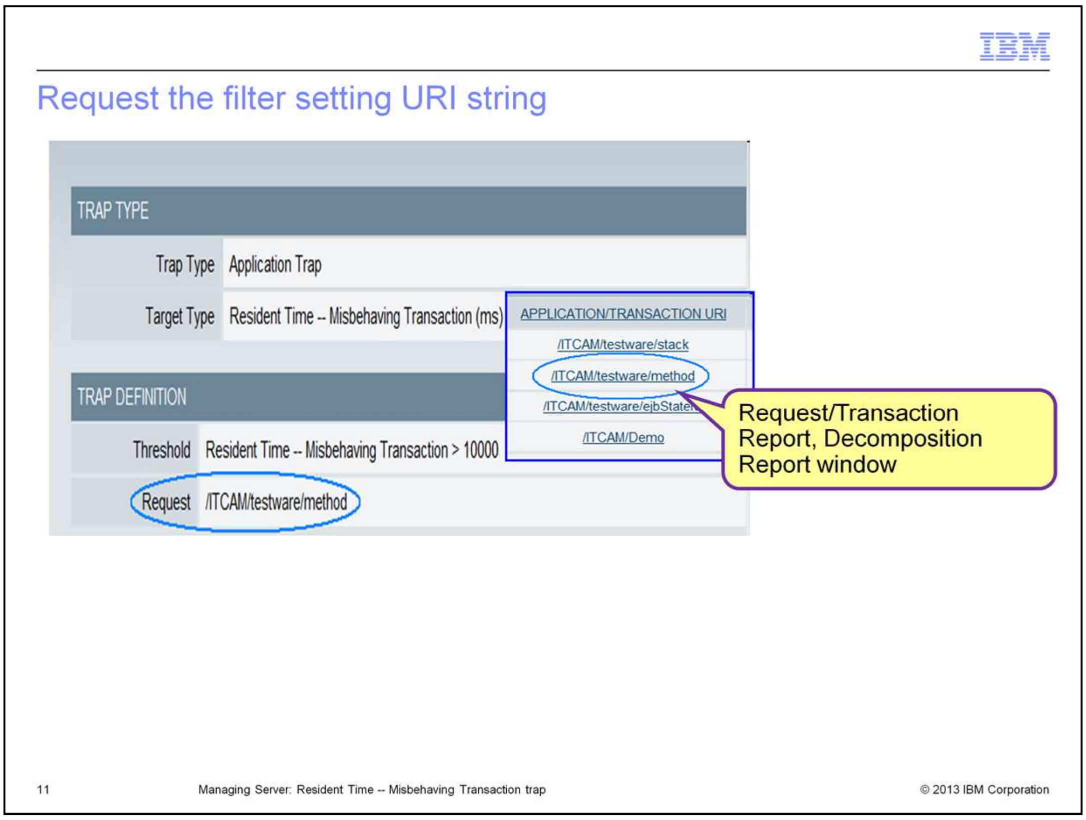
The Level 3 method trace is now disabled for **request1**. However, the trap itself stays active. The **Iterations Left** column on the **Active Traps** table of **Traps Overview** window remains at **5**.

If **request2** triggers the trap, then Level 3 is enabled for **request2** and is then deactivated for **request2** after five trap occurrences for **request2**, and so on.

The trap stays active indefinitely, and must be manually deactivated.

The Managing Server keeps the occurrence count for each individual requests which trigger the trap and deactivates as per the settings for each individual request.

This setting can cause Level 3 method collection for a lot of requests if the trap threshold is set to a low value.



Request the filter setting URI string

TRAP TYPE

Trap Type	Application Trap
Target Type	Resident Time -- Misbehaving Transaction (ms)

TRAP DEFINITION

Threshold	Resident Time -- Misbehaving Transaction > 10000
Request	/ITCAM/testware/method

APPLICATION/TRANSACTION URI

/ITCAM/testware/stack

/ITCAM/testware/method

/ITCAM/testware/ejbState

/ITCAM/Demo

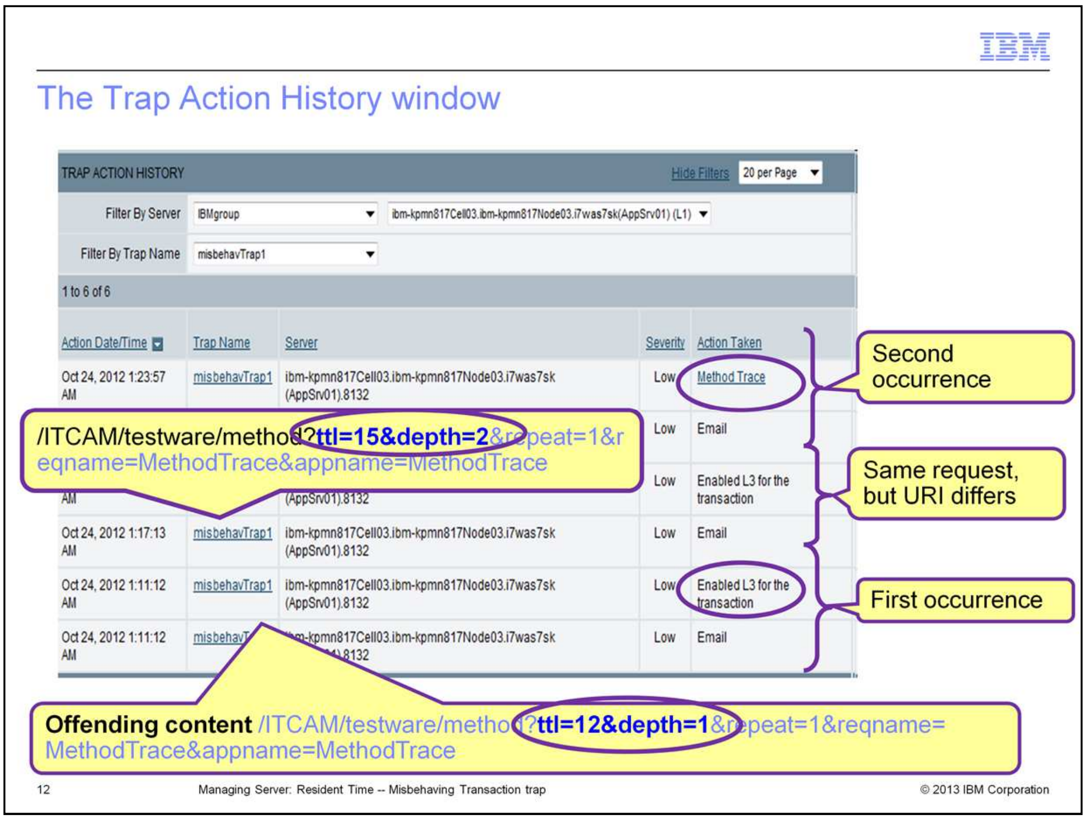
An alternative to the use of ***,** is to specify a request URI in the request filter. This alternative helps to reduce scope of the trap by restricting it to only those requests which satisfy the request filter string.

For example, setting **/ITCAM/testware** in request filter causes the trap to treat **/ITCAM/testware/method** and **/ITCAM/testware/stack** as the same request, even though these two are different requests.

The request filter string should uniquely identify a specific request. For example, setting the filter to string **/ITCAM/testware/stack** excludes the **method** request from this trap.

The URI string to use in the request filter of trap definition might not be obvious because it does not necessarily display in the application browser link.

You can obtain the request URI to use in the filter from application data displayed on Managing Server Visualization Engine windows. For example, **Performance Analysis - Request /Transaction Report**, **Top Reports -Top Slowest Requests**, **In-Flight Request Search, Server Activity Display** all show the request URI data.



The Trap Action History window

TRAP ACTION HISTORY — Hide Filters — 20 per Page

Filter By Server: IBMgroup — ibm-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk(AppSrv01) (L1)

Filter By Trap Name: misbehavTrap1

1 to 6 of 6

Action Date/Time	Trap Name	Server	Severity	Action Taken
Oct 24, 2012 1:23:57 AM	misbehavTrap1	ibm-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk (AppSrv01).8132	Low	Method Trace
[illegible]	[illegible]	[illegible]	Low	Email
[illegible] AM	[illegible]	[illegible] (AppSrv01).8132	Low	Enabled L3 for the transaction
Oct 24, 2012 1:17:13 AM	misbehavTrap1	ibm-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk (AppSrv01).8132	Low	Email
Oct 24, 2012 1:11:12 AM	misbehavTrap1	ibm-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk (AppSrv01).8132	Low	Enabled L3 for the transaction
Oct 24, 2012 1:11:12 AM	misbehav[illegible]	[illegible]-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk [illegible]8132	Low	Email

Second occurrence
/ITCAM/testware/method?ttl=15&depth=2&repeat=1&reqname=MethodTrace&appname=MethodTrace
Same request, but URI differs
First occurrence
Offending content /ITCAM/testware/method?ttl=12&depth=1&repeat=1&reqname=MethodTrace&appname=MethodTrace

The **Trap Action History** window displays the actions that are taken by the trap.

This example shows results from a trap that is defined with request filter set to *.

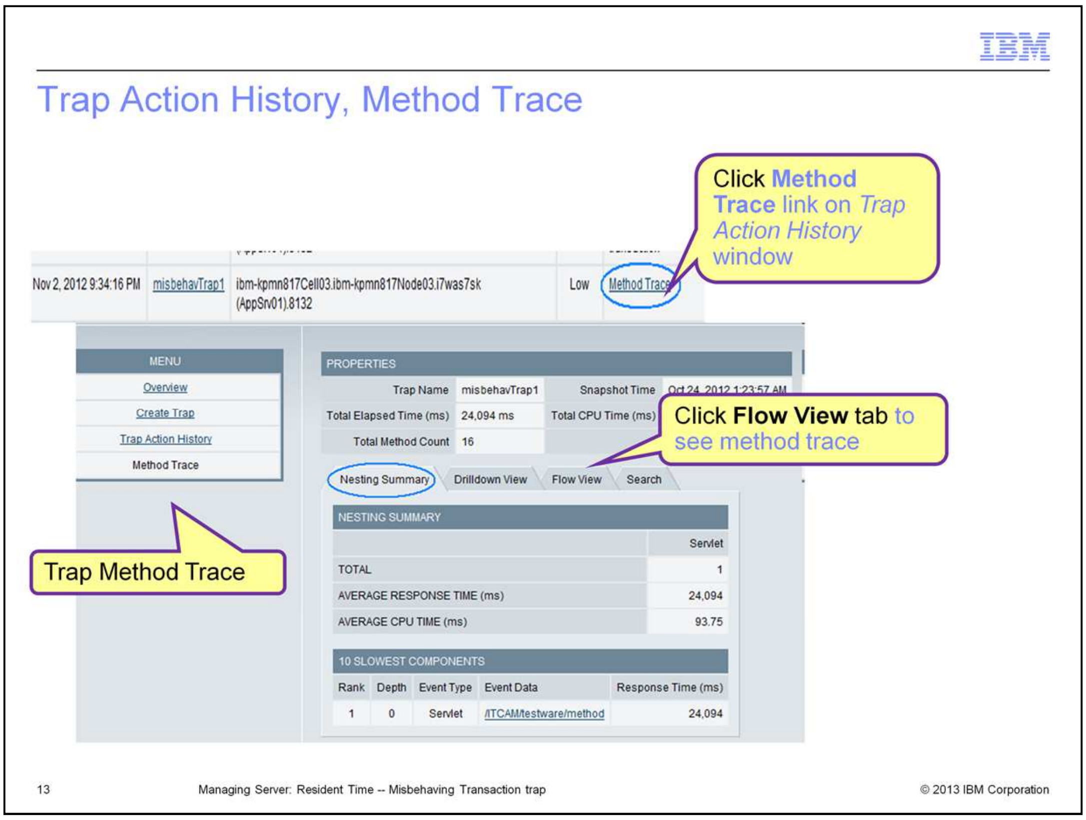
After first occurrence of the trap, the **Action Taken** column shows **Enabled L3 for the transaction**. This value indicates that data collector was notified to enable Level 3 method entry exit collection for the request.

Select the **Trap Name** column link to display **Trap Action History Properties** window, which shows the request URI in the **Offending Content** field.

When the trap triggers again with *identical* URI, Managing Server collects a method trace for the request.

Note: If the same request with different parameters exceeds trap threshold, it is considered an entirely different request by the Managing Server, and a notification to enable **L3 for the transaction** is sent to the data collector again. This behavior is specific to request filter setting of *.

The example shows two occurrences of the trap, one with request parameter **ttl=12&depth=1** and another with **ttl=15&depth=2**. Even though both were the same request **/ITCAM/testware/method,** the Managing Server treated them as different.



Trap Action History, Method Trace

Click **Method Trace** link on *Trap Action History* window

Nov 2, 2012 9:34:16 PM	misbehavTrap1	ibm-kpmn817Cell03.ibm-kpmn817Node03.i7was7sk (AppSrv01).8132	Low	Method Trace

PROPERTIES

Trap Name	misbehavTrap1	Snapshot Time	Oct 24, 2012 1:23:57 AM
Total Elapsed Time (ms)	24,094 ms	Total CPU Time (ms)	
Total Method Count	16		

Click **Flow View** tab to see method trace

NESTING SUMMARY

	Servlet
TOTAL	1
AVERAGE RESPONSE TIME (ms)	24,094
AVERAGE CPU TIME (ms)	93.75

10 SLOWEST COMPONENTS

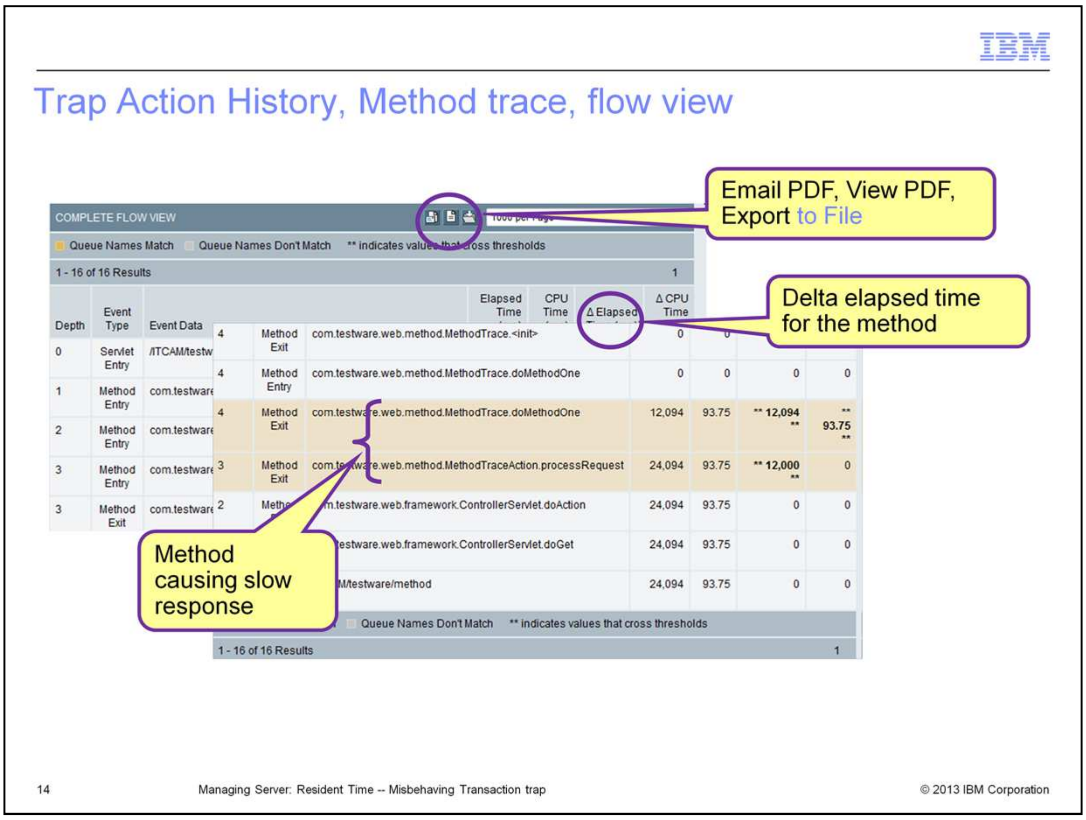
Rank	Depth	Event Type	Event Data	Response Time (ms)
1	0	Servlet	/ITCAM/testware/method	24,094

Trap Method Trace

13 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

To view the method trace, click the **Method Trace** link in the **Action Taken** column of the **Trap Action History** window.

The **Trap Method Trace** window displays. To view the full method trace, click the **Flow View** tab.



Trap Action History, Method trace, flow view

COMPLETE FLOW VIEW

🟧 Queue Names Match ▢ Queue Names Don't Match ** indicates values that cross thresholds

1 - 16 of 16 Results 1

Depth	Event Type	Event Data
0	Servlet Entry	/ITCAM/testw
1	Method Entry	com.testware
2	Method Entry	com.testware
3	Method Entry	com.testware
3	Method Exit	com.testware

			Elapsed Time	CPU Time	Δ Elapsed	Δ CPU Time
4	Method Exit	com.testware.web.method.MethodTrace.<init>			0	0
4	Method Entry	com.testware.web.method.MethodTrace.doMethodOne	0	0	0	0
4	Method Exit	com.testware.web.method.MethodTrace.doMethodOne	12,094	93.75	** 12,094 **	** 93.75 **
3	Method Exit	com.testware.web.method.MethodTraceAction.processRequest	24,094	93.75	** 12,000 **	0
2	Metho	m.testware.web.framework.ControllerServlet.doAction	24,094	93.75	0	0
		estware.web.framework.ControllerServlet.doGet	24,094	93.75	0	0
		M/testware/method	24,094	93.75	0	0

▢ Queue Names Don't Match ** indicates values that cross thresholds

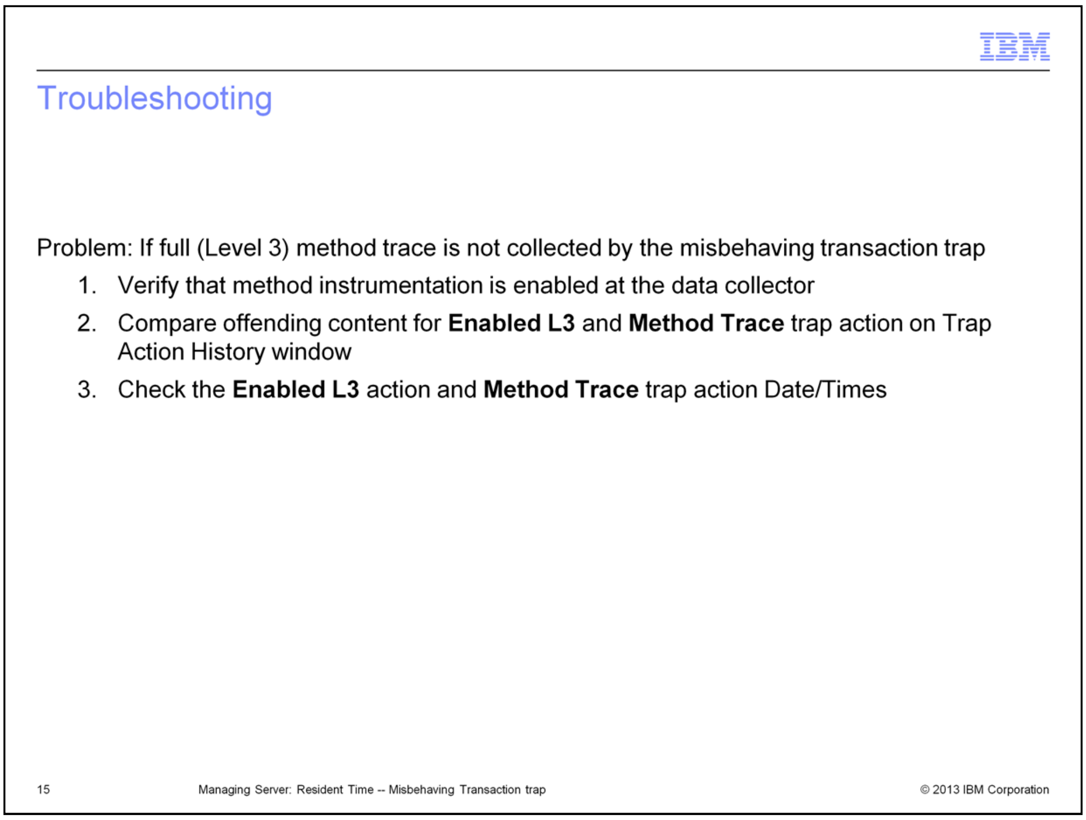
1 - 16 of 16 Results 1

Email PDF, View PDF, Export to File
Delta elapsed time for the method
Method causing slow response

On the **Flow View** tab**,** check the **Delta Elapsed Time** column to locate the methods that took most time to run.

You can click icons for **Email PDF**, **View PDF**, and **Export to File** to export the method trace to an external file that you can send to the developers for analysis.

With *full method trace collection*, the main goal of the **Misbehaving Transaction trap** is achieved.



Troubleshooting

Problem: If full (Level 3) method trace is not collected by the misbehaving transaction trap

1. Verify that method instrumentation is enabled at the data collector
2. Compare offending content for **Enabled L3** and **Method Trace** trap action on Trap Action History window
3. Check the **Enabled L3** action and **Method Trace** trap action Date/Times

15 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

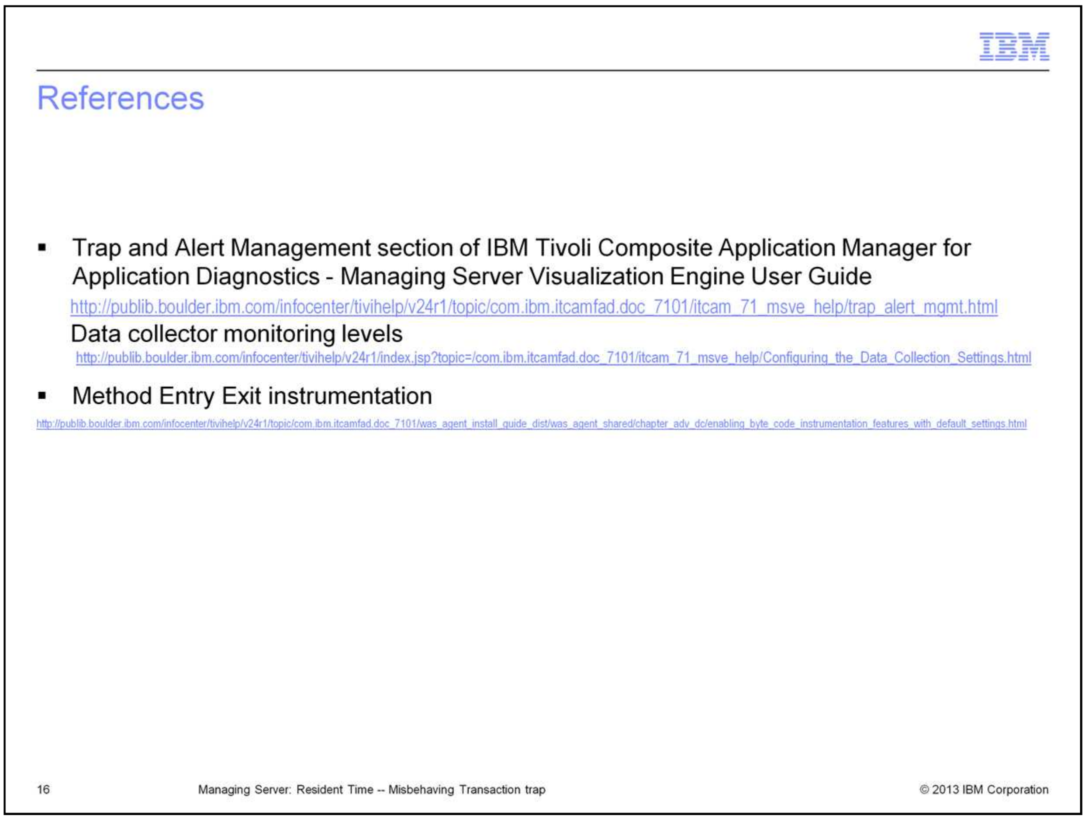
If you find that the trap collected Method Trace with only two methods, Servlet Entry and Servlet Exit, it might be caused by one of these reasons:

1. Verify if the Method Entry Exit instrumentation is enabled at the data collector. This is a required configuration for L3 method trace collection.

2. Verify the offending content of the **Method Trace** action and **Enabled L3** action on **Trap Action History** window. Both should contain the URI of the same request.

3. Inspect the **Action Date/Time** column of **Trap Action History** window to determine if the **Enabled L3** action and **Method Trace** action are very close together in time.

It can take some time for the data collector to enable L3 tracing for the request. If another occurrence of the request occurs right after the first occurrence, then full method trace might not be collected. Subsequent occurrences of the trap will collect full method trace after data collector has enabled **L3 method trace collection**.



References

- Trap and Alert Management section of IBM Tivoli Composite Application Manager for Application Diagnostics - Managing Server Visualization Engine User Guide
  http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/itcam_71_msve_help/trap_alert_mgmt.html
  Data collector monitoring levels
  http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/index.jsp?topic=/com.ibm.itcamfad.doc_7101/itcam_71_msve_help/Configuring_the_Data_Collection_Settings.html
- Method Entry Exit instrumentation

http://publib.boulder.ibm.com/infocenter/tivihelp/v24r1/topic/com.ibm.itcamfad.doc_7101/was_agent_install_guide_dist/was_agent_shared/chapter_adv_dc/enabling_byte_code_instrumentation_features_with_default_settings.html

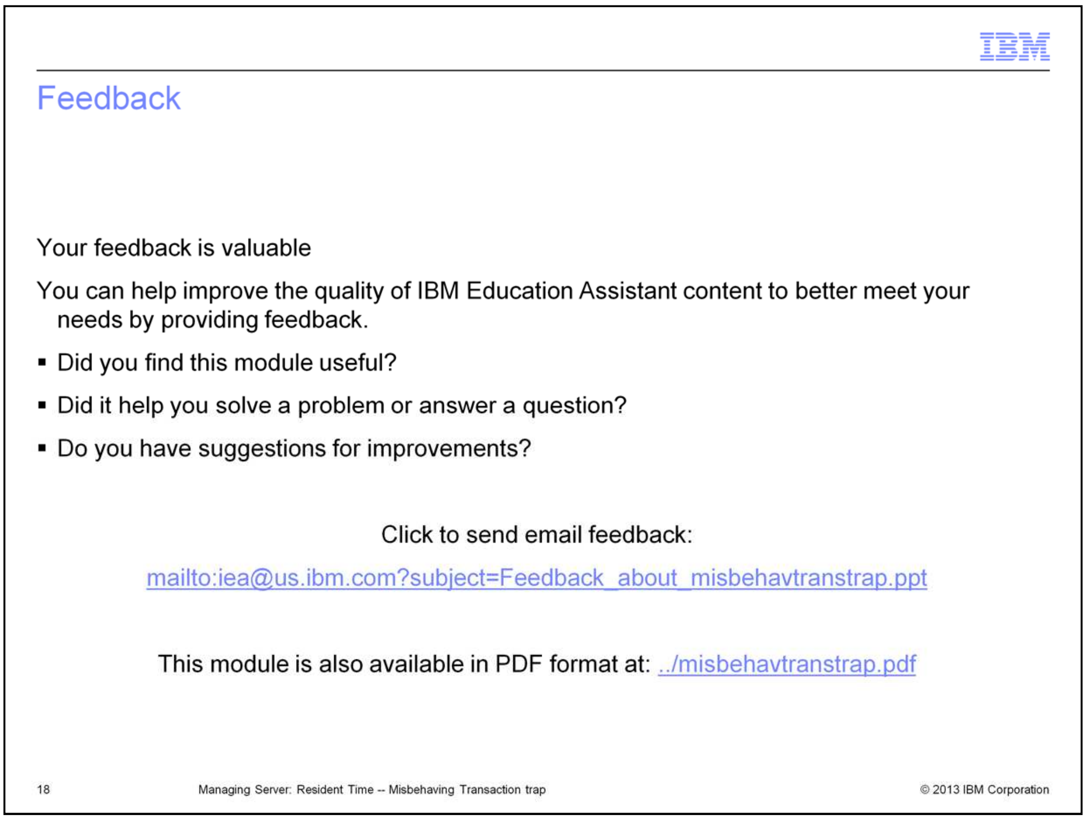
The listed reference information is available if you need it.

Summary

Now that you completed this module, you know how the Misbehaving Transaction trap of Managing Server works and how to use it to collect full (Level 3) method trace for debugging slow requests

17 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

Now that you completed this module, you can use the Resident Time, Misbehaving Transaction trap to use collect full method trace for slow requests.



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?

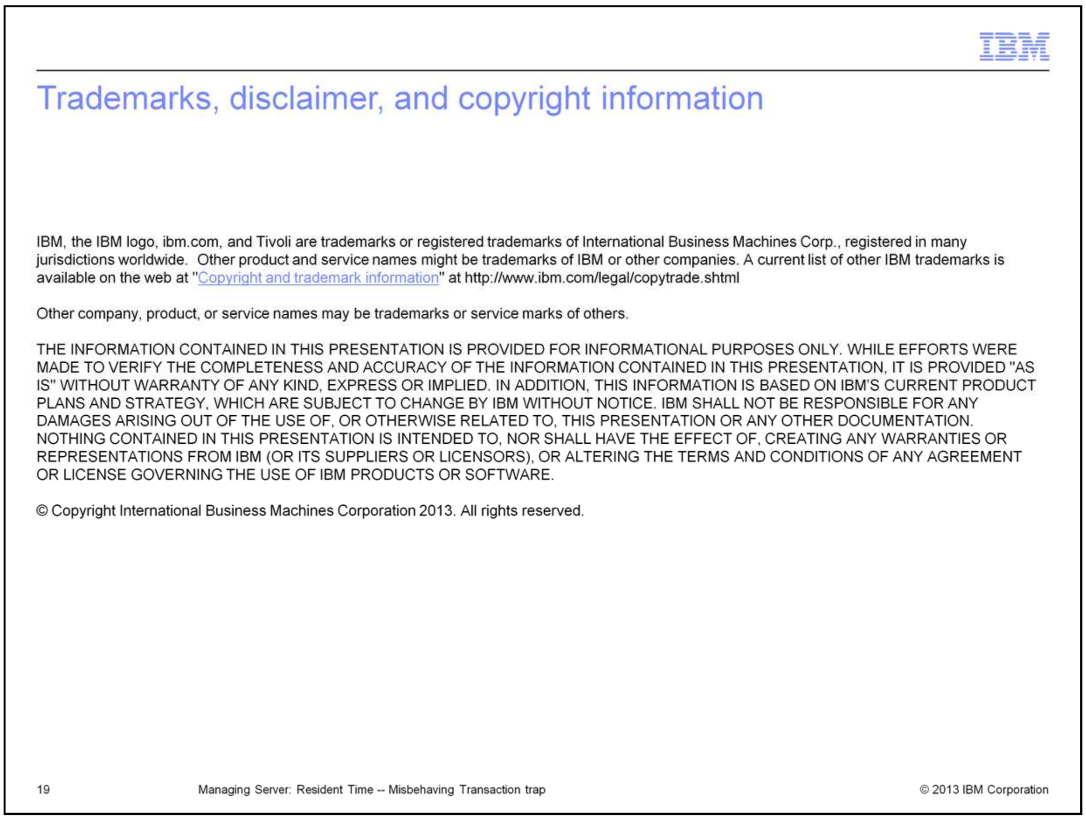
Click to send email feedback:

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

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

18 Managing Server: Resident Time -- Misbehaving Transaction trap © 2013 IBM Corporation

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



Trademarks, disclaimer, and copyright information