

IBM Tivoli Monitoring V6.2.3 Fix Pack 2 and later

Dynamically modify trace settings with the `tacmd` command



IBM Tivoli® Monitoring V6.2.3 Fix Pack 2 and later, Dynamically modify trace settings with the **tacmd** command line interface (CLI) command.

Assumptions

Environment configuration

- IBM Tivoli Monitoring V6.2.3 Fix Pack 03 or later is installed
- The required UNIT trace information

The assumptions for this module are that you have completed installing IBM Tivoli Monitoring V6.2.3 Fix Pack 02 or later. You should have a basic knowledge of the debugging UNIT trace.

The IBM Tivoli Monitoring support team provides the UNIT trace.

Objectives

When you complete this module, you can perform these tasks:

- Set the trace setting with the **tacmd** command
- List the existing trace setting with the **tacmd** command

When you complete this module, you can set and list the trace settings with the **tacmd** CLI command.

Overview

- Benefits of the Command Line Interface command
- Help information for **tacmd settrace**
- Help information for **tacmd listtrace**
- Usage scenario
- Known limitations and things to consider
- Debugging

This module provides you with a brief introduction about setting a trace dynamically with the Command Line Interface (CLI) and the **tacmd** command. It explains the benefits of this command and the syntax for the commands. There is also information about the usage scenario and known limitations. This command is available in IBM Tivoli Monitoring V6.2.3 Fix Pack 02 and subsequent releases.

Benefits

- Eliminates the requirement to log in to the Service Console to set the trace dynamically
 - The Service Console requires a valid user ID and password to the target
 - Firewall rules might block access to the listener port of the remote **SOAP**
- Can automate changes with the **tacmd** command
- Reduces total cost-of ownership of Tivoli Monitoring products
- Simplifies the trace requests sent to clients
- Can use scripts and cron jobs with the **tacmd** command to modify trace settings during off hours
- Can change dynamic tracing from any machine in the infrastructure. Users can modify the traces on agent boxes that are connected to the Tivoli Enterprise Monitoring Server
- Can dynamically change tracing while a product is running so that you can collect better data in the log files
 - Provides better data with the first log file you send with a PMR and provides sufficient context information to diagnose the problem
 - Possibly prevents a request for more documentation

5

Dynamically modify trace settings with the tacmd command

© 2013 IBM Corporation

This slide explains the benefits of the tacmd CLI command.

The command eliminates the need to log in to the Service Console to set a trace dynamically. Sometimes it is not convenient or practical for a customer to use the Service Console on a remote endpoint. The Service Console also requires a valid user ID and password to the target system, whereas the tacmd command eliminates this requirement.

Firewall rules might block access to listener port on the remote SOAP while the CLI does not use any specific port.

You can automate changes with the tacmd command. Automation can provide these benefits:

- Reduce total cost-of ownership of Tivoli Monitoring products.
- Simplify the trace requests sent to clients. The tacmd settrace CLI makes it much easier to set requested traces. There are fewer chances for typographical errors and miscommunication. Customer Support can email a full tacmd settrace command string, that you can copy-and-paste into the command line.
- Scripts and cron jobs can be used with the tacmd command to modify trace settings during off hours with tacmd settrace. You can create a cron job or other time-driven scripts to change RAS1 logging at the time a sporadic problem most often occurs.

Help for the tacmd settrace command (1 of 3)

- The **settrace** command modifies the RAS1 logging level or restores the original tracing after diagnostic data is captured on a local or remote managed system

```
tacmd settrace
{-m|--system} SYSTEM
{-p|--property} PROPERTY
{-o|--option} OPTION
[{-d|--description}] DESCRIPTION
```

- The second form of **settrace** is used to undo any tracing changes that are made and restore the original value

```
tacmd settrace
{-m|--system} SYSTEM
{-p|--property} PROPERTY
{-r|--restore} RESTORE
```

Where:

-m|--system Specifies which managed system to send the **settrace** command. Only one managed system can be specified per command.

-p|--property Identifies the RAS1 trace property to set. Valid values are:
KBB_RAS1 **KDC_DEBUG**
KDE_DEBUG **KDH_DEBUG**
KBS_DEBUG

-o|--option Identifies the trace options to set for the **-p property**

Kxx_DEBUG trace properties support one-character values:

Y for Yes	N for Normal
I for Inhibit (NONE)	V for Verbose
S for State	T for Trace
D for Detail	M for Maximum
A for All	

The **settrace** command modifies the RAS1 logging level or restores original tracing after diagnostic data is captured on a local or remote managed system. Only one managed system can be specified.

To set a trace dynamically, you need to specify the managed system where you want to set the trace, one of the properties, and any corresponding option for the property. The description is optional.

A hub Tivoli Enterprise Monitoring Server can be specified either with its **CMS_NODEID** value or as ***HUB**.

To undo any tracing changes, you must specify the managed system where you want to store the trace, one of the properties and **-r** to restore to the original value.

Help for the tacmd settrace command (2 of 3)

- KBB_RAS1 supports more complex options; for help with KBB_RAS1, use these resources:
 - "Setting traces" in the *IBM Tivoli Monitoring Troubleshooting Guide*
 - Use the settings that are provided by your IBM Software Support representative
 - The `-o` flag is not required if you specify `-r`
 - `-d|--description` Specifies an optional description text string to help identify the reason for changing the trace variable. If a description string contains any embedded blanks, the entire string must be enclosed in double quotation marks
 - `-r|--restore` Specifies that the RAS1 trace property should be restored back to the original value it had at product startup. If the `-r` flag is set, the `-o` and `-d` flags are ignored
 - **Examples**
 - `- tacmd settrace -m AIX RTEMS -p KBB_RAS1 -o "ERROR (UNIT:kpx Error State)"`
 - `- tacmd settrace -m SystemA:Agent1 -p KBB_RAS1 -r`
 - `- tacmd settrace -m *HUB -p KDC_DEBUG -o Y -d "PMR 12345"`
- ```
./tacmd settrace -m itmmt100:TEPS -p KBB_RAS1 -o "ERROR (UNIT:kcq
DETAIL)"
KUISTR000I: Command was performed successfully
```

This slide shows a few examples of setting the trace and the rules regarding KBB\_RAS1 string specifications.

## Help for the tacmd settrace command (3 of 3)

### Notes:

- You can specify a hub Tivoli Enterprise Monitoring Server either with its CMS\_NODEID value or as \*HUB
- Three rules for KBB\_RAS1 string specifications:
  - (1) If there is no default class before the first (UNIT:xxx) parameter, the software assumes a default class of **ERROR**
  - (2) If the string contains embedded blanks, you must enclose the entire string in double quotation marks
  - (3) If the string contains embedded double quotation marks, you must enclose the entire string in outer double quotation marks

### Example

```
tacmd settrace -m zOS:CMS -p KBB_RAS1 -o "ERROR
(UNIT:KFASTINH,ENTRY:"KFA_InsertNodestatus" ALL) "
```

This slide shows a few final notes on the command.



## Help for tacmd listtrace command

- The **listtrace** command queries the RAS1 logging level on a managed system
- Usage
 

```
tacmd listtrace
 {-m|--system} SYSTEM
 {-p|--property} PROPERTY
```
- Where:
  - **-m|--system** specifies which managed system to send the command.
  - **-p|--property** specifies the RAS1 variable name being queried.
- Example:
 

```
- tacmd listtrace -m TVT6044 -p KBB_RAS1
 KUILTR000I: The current value of KBB_RAS1 at TVT6044 is:
 - ERROR (UNIT:kfastini ER ST)(UNIT:kqmarm ER ST)(UNIT:ko4srv ER
 ST)(UNIT:ko4bkgnd ER ST) (UNIT:kdsma ALL) (COMP:SMAF ALL)
 (unit:Managed detail)
- tacmd listtrace -m TVT6044 -p KDC_DEBUG
 KUILTR000I: The current value of KDC_DEBUG at TVT6044 is: Y
```

The **listtrace** command queries the RAS1 logging level on a managed system.

To list the trace, you must specify the managed system where you want to list the trace and one of the properties names that you want to query. This example lists the value for the property KBB\_RAS1 and KDC\_DEBUG.

## Usage scenario

- Dynamic KBB\_RAS1 changes are replacements and not added to the existing value, for example:
  - `.ini` file contains `KBB_RAS1=ERROR`
  - Level 2 support requests that you turn on **kxx** and **kyy** for **<MSN>**
  - `tacmd settrace -m <MSN> -p KBB_RAS1 -o "ERROR (UNIT:kxx ALL) (UNIT:kyy ALL) "`
- An additional request to add **kzz** tracing:
  - `tacmd settrace -m <MSN> -p KBB_RAS1 -o "ERROR (UNIT:kxx ALL) (UNIT:kyy ALL) (UNIT:kzz ALL) "`
  - Notice that the command is the same as the previous command plus the **kzz**
- To avoid extra typing, you can copy-and-paste the output from a **tacmd listtrace** command against **KBB\_RAS1** when you set the modified string:
  - After the diagnostic data has been collected, you can revert to the minimal **ERROR-level** tracing
  - `tacmd settrace -m <MSN> -p KBB_RAS1 -o ERROR`
- All prior (UNIT:xxx) filters are cleared; it is not necessary to specify **ANY** to clear prior filters

10

Dynamically modify trace settings with the tacmd command

© 2013 IBM Corporation

Here you see how you can set dynamic tracing in a typical customer environment with the CLI.

A dynamic **KBB\_RAS1** change is a replacement; it does not add to the existing value.

For example, the `.ini` file contains `KBB_RAS1=ERROR`. Level 2 support asked for you to turn on the **kxx** and **kyy** traces for **<MSN>**. Run the command **`tacmd settrace -m <MSN> -p KBB_RAS1 -o "ERROR (UNIT:kxx ALL)(UNIT:kyy ALL)"`**.

Later support requests for you to add the **kzz** tracing. Remember that the change is a replacement. The command is identical to the previous command plus the additional **kzz** information. The command looks like **`tacmd settrace -m <MSN> -p KBB_RAS1 -o "ERROR (UNIT:kxx ALL) (UNIT:kyy ALL) (UNIT:kzz ALL)"`**.

To avoid extra typing, you can copy-and-paste the output from a **tacmd listtrace** command against **KBB\_RAS1** to set the modified string.

After the diagnostic data is collected, you can revert to the minimal **ERROR-level** tracing. To revert, run the command **`tacmd settrace -m <MSN> -p KBB_RAS1 -o ERROR`**. This command clears all prior (UNIT:xxx) filters. It is not necessary to specify **ANY** to clear prior filters.

## Known limitations and things to consider

- Only one managed system can be specified per command
- The **tacmd listtrace** command requires the “-v” option to query the RAS1 logging level. There is no default option to query the full list of trace settings
- There is a limitation of 1000 bytes for the KBB\_RAS1 RAS1 property
- Dynamic KBB\_RAS1 changes are replacements
- Significantly increasing tracing levels with the dynamic tracing might wrap up the RAS1 logs quickly and overlay important trace messages. You must use appropriate parameters for the KBB\_RAS1\_LOG variable to avoid the wrapping up the logs
- Dynamic tracing does not help for lock ups that have no prior warning symptoms, for startup errors, or infrequent failures
- You must be judicious in the use of dynamic tracing calls, do not set KBB\_RAS1=ALL when an error occurs
- You must remember to turn off extra RAS1 tracing once the problem is resolved

11

Dynamically modify trace settings with the tacmd command

© 2013 IBM Corporation

Here are the known limitations and the important aspects you should consider before setting the trace:

Only one managed system can be specified per command.

The **tacmd listtrace** command requires the “-p” option to query the RAS1 logging level. There is no default option to query the full list of trace settings.

There is a limitation of 1000 bytes for the KBB\_RAS1 RAS1 property.

Dynamic KBB\_RAS1 changes are replacements.

Significantly increasing tracing levels with dynamic tracing might wrap up the RAS1 logs quickly and overlay important trace messages. You must use appropriate parameters for the KBB\_RAS1\_LOG variable to avoid wrapping up the logs.

Dynamic tracing does not help for lock ups that have no prior warning symptoms, for startup errors, or infrequent failures.

You must be judicious in the use of dynamic tracing calls; do not set KBB\_RAS1=ALL when an error occurs.

You must remember to turn off extra RAS1 tracing after the problem is resolved.

## Debugging

- The RAS1 log file contains either of these messages after you successfully turn on traces with **tacmd**:
  - \*\*\* KBB\_RAS1=ERROR (UNIT:kdy ALL) is now in effect
  - \*\*\* KDC\_DEBUG=Y is now in effect
- If you do not see either of the preceding trace message in the RAS1 log, perform the steps in the next bullet
- To debug the **tacmd settrace**, you can perform these steps:
  - 1.Windows®: %itm\_install%\cms\kbbenv  
UNIX® or Linux®: \$itm\_install\config\\_ms\_<temsname>.config  
z/OS®: &rhilev.&rte.RKANPARU(KppENV)
  - 2.Add this trace parm **KBB\_RAS1=ERROR (UNIT:KDY ALL)**
  - 3.Stop and start the hub Tivoli Enterprise Monitoring Server
  - 4.Re-create the problem by entering the **tacmd settrace** command
  - 5.Collect problem information with **pdcollect** and send this information to IBM support

If you do not see either of the standing trace message in the RAS1 log, set the trace manually. Use the steps that are shown and contact IBM support with the information received from running **pdcollect** on the hub Tivoli Enterprise Monitoring Server.

## Summary

Now that you have completed this module, you can perform these tasks:

- Set the trace setting with the **tacmd** command
- List the existing trace setting with the **tacmd** command

Now that you have completed this module, you can set and list the trace settings with the **tacmd** CLI command.

## Trademarks, disclaimer, and copyright information

IBM, the IBM logo, ibm.com, Tivoli, and z/OS are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of other IBM trademarks is available on the web at "[Copyright and trademark information](http://www.ibm.com/legal/copytrade.shtml)" at <http://www.ibm.com/legal/copytrade.shtml>

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Windows, and the Windows logo are registered trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

THE INFORMATION CONTAINED IN THIS PRESENTATION IS PROVIDED FOR INFORMATIONAL PURPOSES ONLY. WHILE EFFORTS WERE MADE TO VERIFY THE COMPLETENESS AND ACCURACY OF THE INFORMATION CONTAINED IN THIS PRESENTATION, IT IS PROVIDED "AS IS" WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. IN ADDITION, THIS INFORMATION IS BASED ON IBM'S CURRENT PRODUCT PLANS AND STRATEGY, WHICH ARE SUBJECT TO CHANGE BY IBM WITHOUT NOTICE. IBM SHALL NOT BE RESPONSIBLE FOR ANY DAMAGES ARISING OUT OF THE USE OF, OR OTHERWISE RELATED TO, THIS PRESENTATION OR ANY OTHER DOCUMENTATION. NOTHING CONTAINED IN THIS PRESENTATION IS INTENDED TO, NOR SHALL HAVE THE EFFECT OF, CREATING ANY WARRANTIES OR REPRESENTATIONS FROM IBM (OR ITS SUPPLIERS OR LICENSORS), OR ALTERING THE TERMS AND CONDITIONS OF ANY AGREEMENT OR LICENSE GOVERNING THE USE OF IBM PRODUCTS OR SOFTWARE.

© Copyright International Business Machines Corporation 2013. All rights reserved.