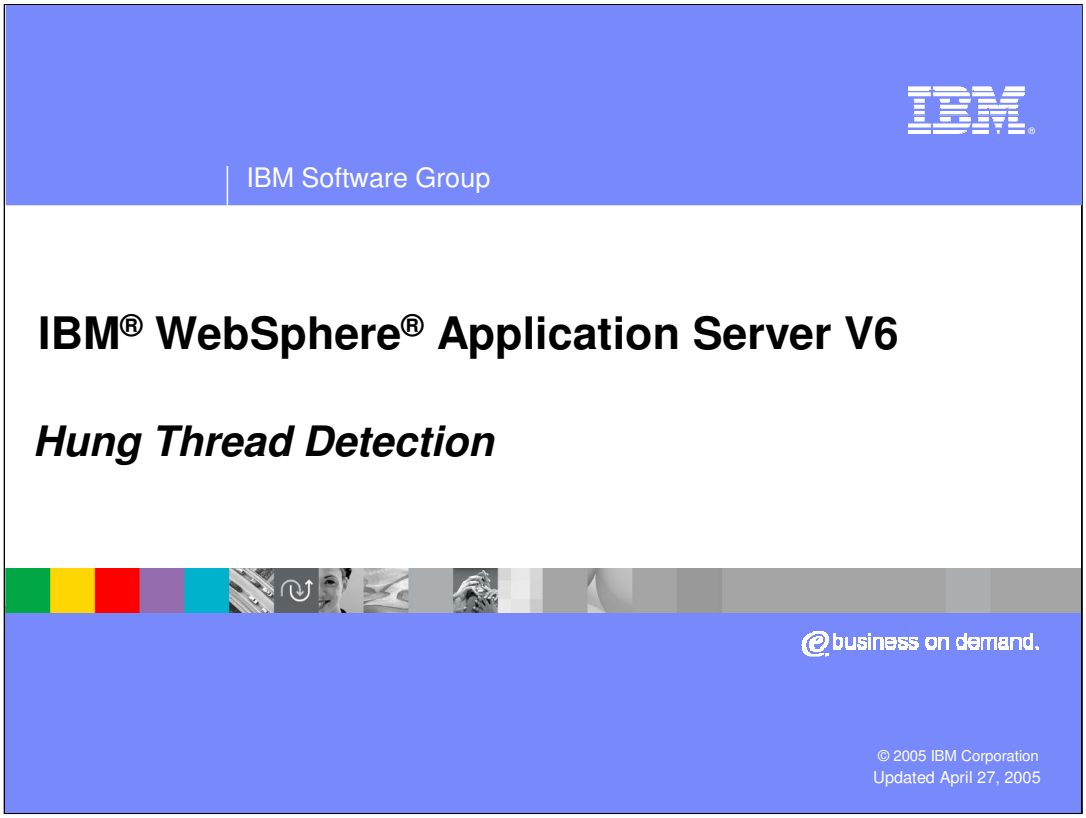
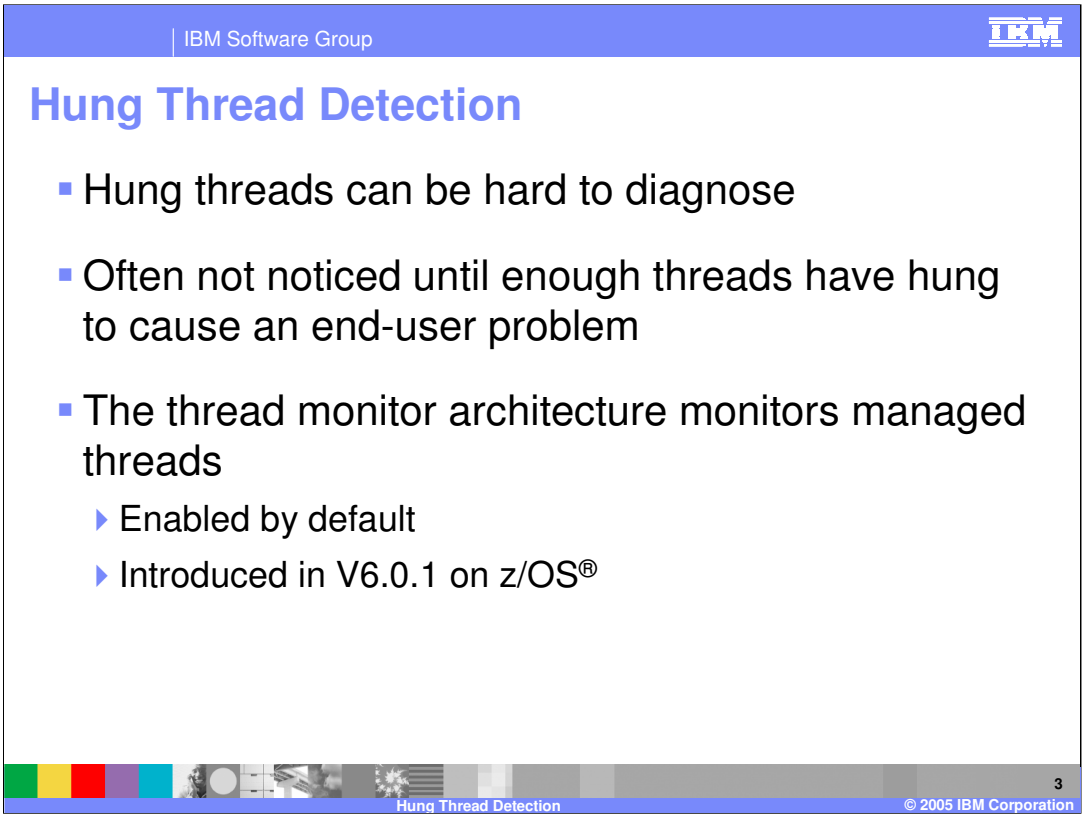


This presentation will focus on the hung thread detection feature of WebSphere Application Server V6.

Goals

- Explain the hung thread detection feature of WebSphere Application Server V6
  - Functionality
  - Configuration

The goal of this presentation is to explain how hung thread detection works in WebSphere Application Server V6, and how you can customize the hang detection policy.



Hung threads have traditionally been hard to diagnose. This is usually because hung threads will go unnoticed until enough threads have hung to cause a problem for your users, such as an application not responding even though it appears to be running. The thread monitor architecture, which was introduced in version 6.0.1 on z/OS, monitors managed threads, meaning threads that belong to a thread pool, and issues a notification if it detects any hung threads. Threads created by application code cannot be monitored by the thread monitor.

Hung Thread Detection (cont.)

- No action taken to kill the thread. Only a notification mechanism
- When a thread is suspected to be hung, notification is sent 3 ways:
  - Java™ Management Extensions (JMX) notification for JMX listeners
  - ThreadPool metric for Performance Monitoring Infrastructure (PMI) clients
  - Message written to Sysprint

```
Trace: 2005/03/12 17:40:04.930 01 t=8CF658 c=UNK key=P8 (13007002)
  ThreadId: 00000038
  FunctionName: com.ibm.ws.runtime.component.ThreadMonitorImpl
  SourceId: com.ibm.ws.runtime.component.ThreadMonitorImpl
  Category: WARNING
  ExtendedMessage: BBOO0221W: WSVR0605W: Thread "WebSphere:ORB.thread.pool
t=008cca60" (00000028) has been active for 608089 milliseconds and may be
hung.  There is/are 1 thread(s) in total in the server that may be hung.
```


The thread monitor doesn't try to deal with the hung threads, it just issues notifications, so that you can then take action to resolve the issue.

When a hung thread is detected, three notifications are sent: a JMX notification for JMX listeners, PMI Thread Pool data is updated for tools like the Tivoli® Performance Viewer, and a message is written to SystemPrint to indicate the name and ID of the hung thread, as well as the time for which it has been hung.

Hung Thread Detection: Internals

- When the thread pool gives work to a thread, it notifies the thread monitor
  - Thread monitor notes thread ID and timestamp
- Thread monitor compares active threads to timestamps
  - Threads active longer than the time limit are marked "potentially hung"
- Performance impact is minimal (< 1%)

When the thread pool issues work to a thread, it sends a notification to the thread monitor, which notes the thread ID and the time in a list.

At user-configurable intervals, the thread monitor looks at the active threads, and compares them to the list, to determine how long each thread has been active. If a thread has been active longer than the user-specified threshold, the thread is marked as "potentially hung", and the notifications are sent as discussed on the previous slide.

The performance impact of this monitoring is less than 1%.

Hung Thread Detection: Internals (cont.)

- What about false alarms?
  - e.g., a thread that takes several minutes to complete a long-running query
- If a thread previously reported to be hung completes its work, a notification is sent:

```
[7/15/04 15:03:47:684 EDT] 37c18e37 ThreadMonitor W WSVR0606W: Thread
"Servlet.Engine.Transports : 0" (37c18e37) was previously reported to be
hung but has completed.  It was active for approximately 105,021
milliseconds.  There are 0 threads in total in the server that still may
be hung.
```

- The monitor has a self-adjusting system to make a best effort to deal with false alarms

It's possible that a thread could actually be running for longer than the specified threshold for legitimate reasons. For example, a thread could be executing a large database query that takes several minutes to return.

The thread monitor is built to recognize false alarms and adjust itself automatically. When a thread that was previously marked as "potentially hung" completes its work and exits, a notification is sent. After a certain number of false alarms, the threshold is automatically increased by 50% to account for these long-running threads. The idea is that if there are several threads that are routinely active for 20 minutes, the threshold will eventually adjust itself to be higher than 20 minutes, so as to not mark those threads as hung.

Hung Thread Detection: Configuration

- Create custom properties on the application server:

Property	Units	Default	Description
com.ibm.websphere.threadmonitor.interval	secs.	180	interval at which the thread pools will be polled for hung threads
com.ibm.websphere.threadmonitor.threshold	secs.	600	the length of time that a thread can be active before being marked as “potentially hung”
com.ibm.websphere.threadmonitor.false.alarm.threshold	N/A	100	the number of false alarms that can occur before automatically increasing the threshold by 50%.

- Can also be configured on the fly using wsadmin
  - Information Center shows an example of this

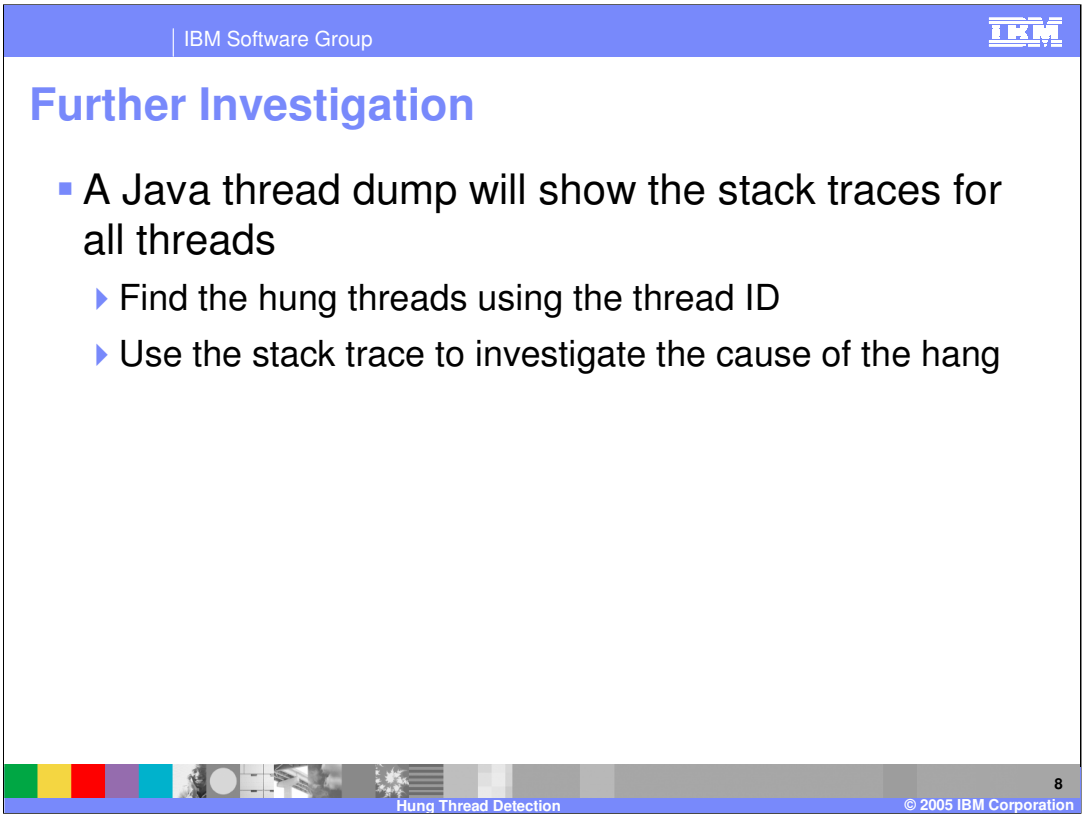
The hang detection policy can be configured by creating custom properties for the application server.

com.ibm.websphere.threadmonitor.interval is the interval at which the thread pools will be polled for hung threads (in seconds). It defaults to 180 seconds, which is 3 minutes.

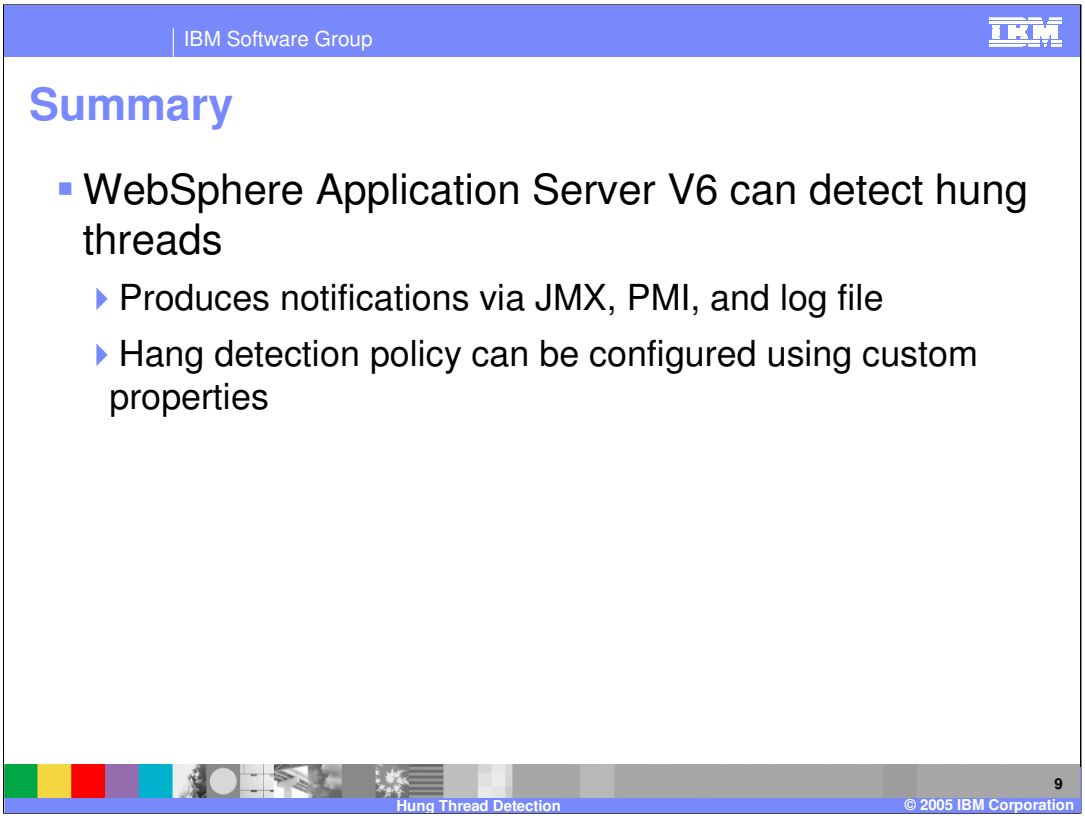
com.ibm.websphere.threadmonitor.threshold is the length of time that a thread can be active before being marked as “potentially hung”. The default value is ten minutes.

com.ibm.websphere.threadmonitor.false.alarm.threshold is the number of false alarms that can occur before automatically increasing the threshold by 50%. The default value is 100. Automatic adjustment can be disabled altogether by setting this property to zero.

The application server must be restarted for these changes to take effect. To adjust the hang detection policy on the fly, use wsadmin. The WebSphere Application Server Information Center has detailed instructions on adjusting the policy with wsadmin.



After being notified of a hung thread problem, the most logical next step is to trigger a Java thread dump. Use the thread ID from the notification to find the hung thread in the dump, and use the stack trace information to begin investigating the cause of the problem.



In summary, this presentation has focused on the new hung thread detection feature of WebSphere Application Server V6. The thread monitor monitors thread pools in an Application Server and notifies you of hung threads via JMX, PMI, and the SystemOut.log file. The policy that governs the behavior of the thread monitor can be configured using custom properties.

Trademarks, Copyrights, and Disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM	CICS	IMS	MQSeries	Tivoli
IBM(logo)	Cloudscape	Informix	OS/390	WebSphere
e(logo)business	DB2	iSeries	OS/400	xSeries
AIX	DB2 Universal Database	Lotus	pSeries	zSeries

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

Intel, ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel 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.

Linux is a registered trademark of Linus Torvalds.

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

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements and/or changes in the product(s) and/or program(s) described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NONINFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (e.g., IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.