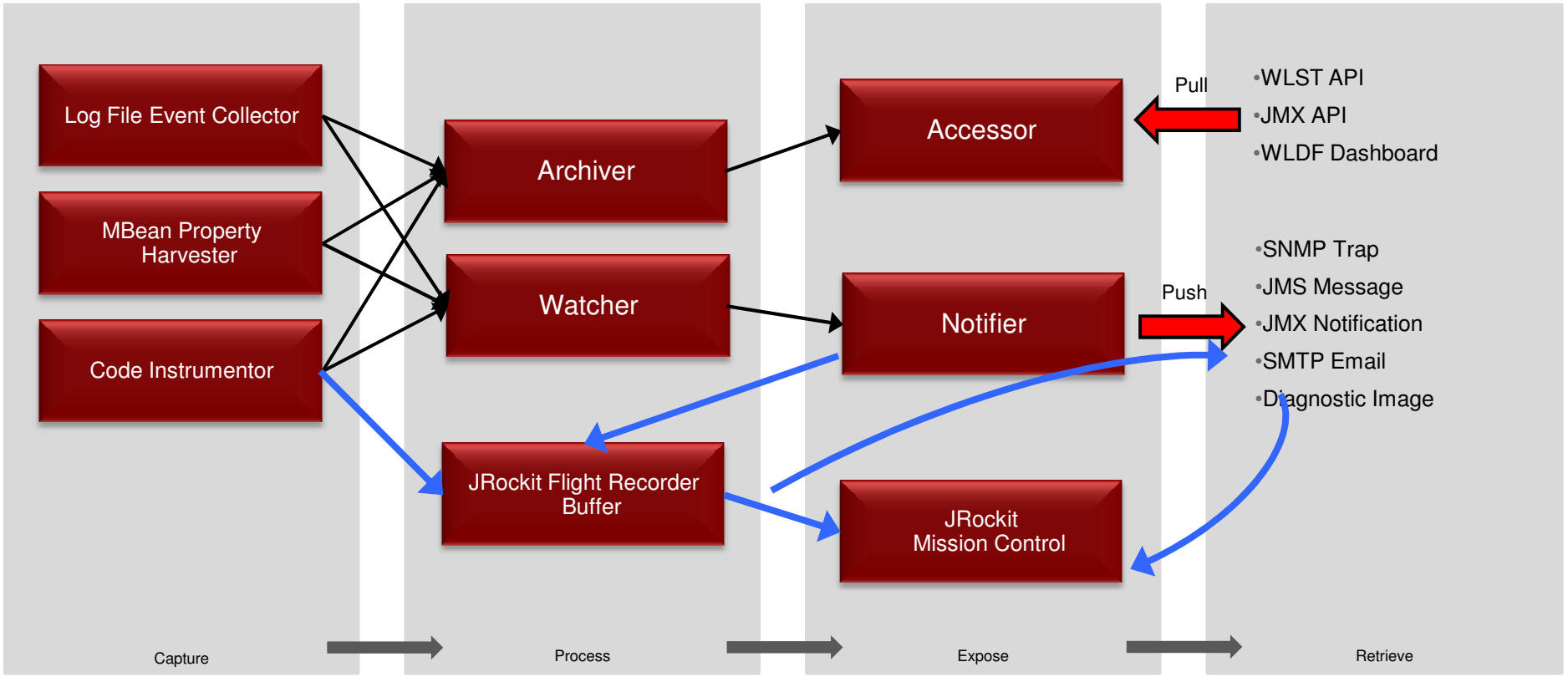
Oracle WebLogic Diagnostics and Troubleshooting

Duško Vukmanović
Principal Sales Consultant, FMW

What is the WebLogic Diagnostic Framework?

- A framework for diagnosing problems that happen at runtime in a WebLogic Server environment and in the applications that are deployed on the server(s)
- Provides a watch/notification system with many options for defining conditions to watch for
- Provides pointcut weaving for *advanced* code instrumentation and troubleshooting
- Enables you to dump a diagnostic image to disk and capture a large set of runtime data from WebLogic, JRockit and your applications
- Integrates with JRockit Flight Recorder to capture WebLogic Events for offline or after-the-fact viewing

Overview of the WebLogic Diagnostics Framework



WLDF Components

Configuring and Using the WebLogic Diagnostic Framework

- **Server-Level Configuration** (domain config.xml)
  - Diagnostic Image Capture
  - Diagnostic Archives
- The following WLDF components are configured as **Diagnostic System Modules**, or resources, that can be deployed to server instances:
  - Harvester (for collecting metrics)
  - Watches and Notifications
  - Instrumentation
- **Application-Level Configuration**
  - The WLDF Instrumentation component can be used with applications
  - The Instrumentation component is configured in a resource descriptor file deployed with the application in the applications archive file

But, why should I care?

- This will allow you to monitor and notify on WebLogic internal metrics that may be causing problems. Common examples include…
- JDBC connection pool metrics like:
  - Average/Max Connection Pool Wait Time
  - Average/Max Number In Use
  - Prepared Statement Cache Hit/Miss Ratio
- JMS Server Metrics like:
  - Number of Messages
  - Number of consumers >=1

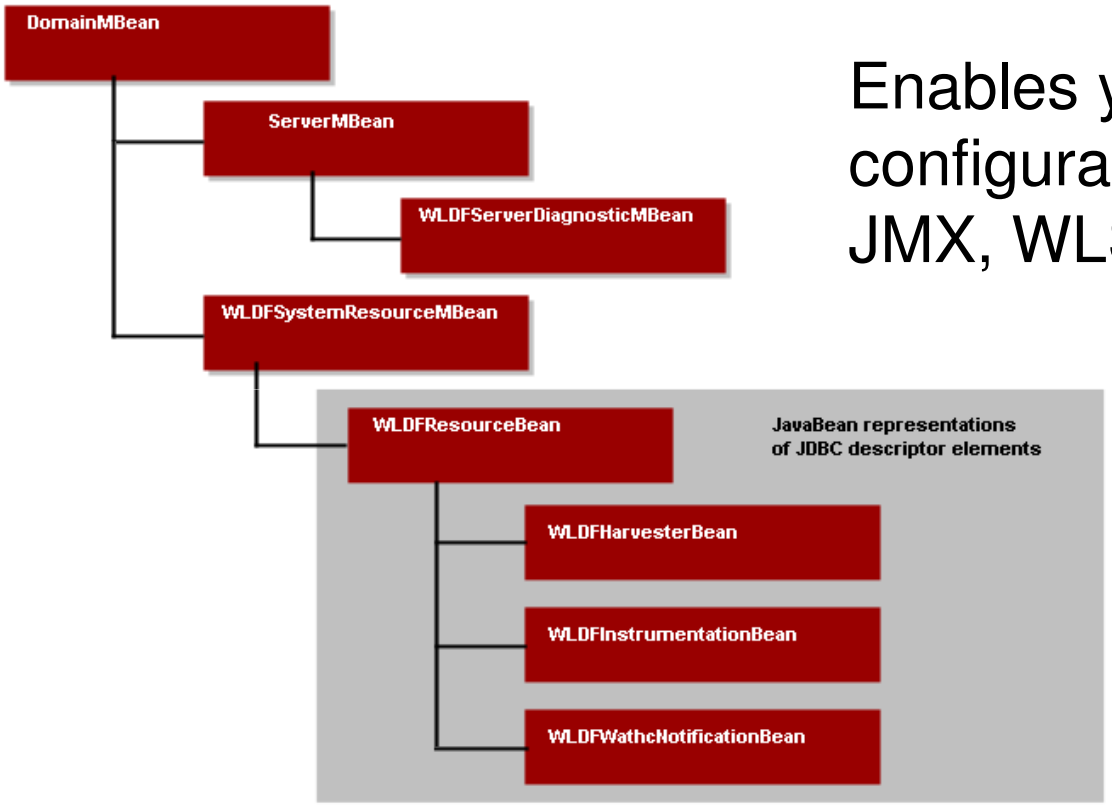
Configuring Diagnostic System Modules

Diagnostic System Module Overview

Configuring Diagnostic System Modules

- A Diagnostic System Module (DSM) is a configuration container for one or more of the following:
  - **Harvester** - for collecting metrics from MBeans
  - **Watches and Notifications** – conditions to watch for and the notification action to take
  - **Instrumentation** - The Instrumentation component of the WebLogic Diagnostic Framework (WLDF) provides a mechanism for adding diagnostic code to WebLogic Server® instances and the applications running on them
- You create a diagnostic system module through the Administration Console or the WebLogic Scripting Tool (WLST)
- DSM's are globally available for targeting servers and clusters and can be targeted to multiple servers and clusters

WLDF Configuration MBeans



Enables you to view & edit the configuration information using JMX, WLST

Managing Diagnostic System Modules

- A DSM can be targeted to zero, one, or more servers, although a given server can have only one module targeted to it at a time.
- You can create multiple DSM's that monitor different aspects of your system and then choose which module to target to a server or cluster, based on what you want to monitor at that time.
- Because you can target the same module to multiple servers or clusters, you can write general purpose modules that you want to use across a domain.
- You can change the target of a diagnostic module without restarting the server instance(s) to which it is targeted or untargeted.
- This gives you considerable flexibility in writing and using diagnostic monitors that address a specific diagnostic goal, without interfering with the operation of the server instances themselves.

The Harvester

Harvester – Configuring and Managing

Diagnostic System Module Components

- Data must meet certain requirements in order to be ***harvestable***, and it must meet further requirements in order to be ***harvested***:
- ***Harvestable*** *data* is data that can potentially be harvested from ***harvestable entities***, including MBean types, instances, and attributes. To be harvestable, an MBean must be registered in the local WebLogic Server runtime MBean server. Only simple type attributes of an MBean can be harvestable.
- ***Harvested*** *data* is data that is currently being harvested. To be harvested, the data must meet all the following criteria:
  - The data must be *harvestable and* configured to be harvested.
  - For custom MBeans, the MBean must be currently registered with the server
  - The data must not throw exceptions while being harvested.

Harvesting MBean Metrics

Configuring Diagnostic System Modules

- The Harvester is used to collect data from harvestable entities, including MBean types, instances, and attributes
- Only simple type attributes of an MBean can be harvested
- To be harvestable, an MBean must be registered in the local WebLogic Server runtime MBean server
- Data Types
  - A Type (only)
  - An Attribute of a Type (Type+Attributes)
  - An Instance of a Type (Type+Instances)
  - An Attribute of an Instance of a Type (Type+Instances+Attributes)
- Sampling Period

Configuring the Sampling Period

Configuring the Harvester

- The <sample-period> element sets the sample period for the Harvester
- The sample period specifies the time between each cycle.
  - For example, if the Harvester begins execution at time T, and the sample period is I, then the next harvest cycle begins at T+I. If a cycle takes A seconds to complete and if A exceeds I, then the next cycle begins at T+A. If this occurs, the Harvester tries to start the next cycle sooner, to ensure that the average interval is I.

```
<wldf-resource xmlns="http://www.bea.com/ns/weblogic/weblogic-diagnc
  <name>myWLDFResource</name>
  <harvester>
    <enabled>true</enabled>
    <sample-period>5000</sample-period>
```

Harvesting a Type

Configuring the Harvester

- The <sample-period> element sets the sample period for the Harvester
- The sample period specifies the time between each cycle, from the start of the cycle to the start of the next cycle. If a cycle takes longer than the sample-period, the next cycle will start as soon as the current cycle is complete
- If no <harvested-instance> is present, all instances that are present at the time of each harvest cycle are collected.

```
<harvested-type>
   <name>weblogic.management.runtime.ServerRuntimeMBean</name>
</harvested-type>
```

Harvesting an Attribute of a Type

Configuring the Harvester

- The optional <harvested-attribute> element specifies that metrics are to be collected only for the listed attributes of the specified type. An attribute is specified by providing its name. The first character should be capitalized. For example, an attribute defined with getter method getFoo() is named Foo
- The <harvested-attribute> element also supports an expression syntax for “drilling down” into attributes that are complex or aggregate objects, such as lists, maps, simple POJOs (Plain Old Java Objects), and various nestings of these types
- If no <harvested-attribute> is present, all harvestable attributes defined for the type are collected

```
<harvested-type>
    <name>weblogic.management.runtime.WLDFHarvesterRuntimeMBean</name>
    <harvested-attribute>TotalSamplingTime</harvested-attribute>
    <harvested-attribute>CurrentSnapshotElapsedTime</harvested-attribute>
</harvested-type>
```

Watches & Notifications

Watches & Notifications

Configuring Diagnostic System Modules

- A ***Watch*** identifies a situation that you want to trap for monitoring or diagnostic purposes. You can configure three types of watches:
  - **Harvester** - monitor the set of harvestable MBeans in the local runtime MBean server.
  - **Log** - monitor the set of messages generated into the server log.
  - **Instrumentation** (or Event Data) monitor the set of events generated by the WLDF Instrumentation component.

- A ***Notification*** is an action that is taken when a watch rule expression evaluates to true. WLDF supports the following types of notifications
  - **SMTP Notification** – Allows you to send a notification via e-mail using an SMTP Session
  - **JMX Notification** – Allows you to raise a JMX notification on the server's WLDFWatchJMXNotificationRuntimeMBean
  - **JMS Notification** – Allows you to send a JMS message to a Queue or Topic
  - **SNMP Notification** – Allows you to raise an SNMP trap on the server
  - **Image Notification** – Allows you to dump a diagnostic image to disk

Options for All Types of Watches

Configuring Diagnostic System Modules

- **Watch Rule Expression** - a logical expression that specifies what significant events the watch is to trap
- **Notifications for Watch** - Each watch can be associated with one or more notifications that are triggered whenever the watch evaluates to true
- **Alarm Options** - Watches can be specified to trigger repeatedly, or to trigger once, when a condition is met. For watches that trigger repeatedly, you can optionally define a minimum time between occurrences
- **Severity Options** - Watches contain a severity value which is passed through to the recipients of notifications
- **Enabled Options** - Each watch can be individually enabled and disabled, using the sub-element <enabled>.

Configuring Harvester Watches

- A Harvester watch can monitor any runtime MBean in the local runtime MBean server
- Harvester watches are triggered in response to a harvest cycle
- Harvesting does not have to be configured and enabled for harvester watches
- However, configuring the Harvester can provide advantages - for example the data will be archived and can be viewed offline leading up to the watch being triggered

```
<wldf-resource xmlns="http://www.bea.com/ns/weblogic/weblogic-diagnostics" xmlns
  <name>mywldf1</name>

  <watch-notification>
    <watch>
      <name>simpleWebLogicMBeanWatchRepeatingAfterWait</name>
      <enabled>true</enabled>
      <rule-type>Harvester</rule-type>
      <rule-expression>
    (${mydomain:Name=WLDFHarvesterRuntime,ServerRuntime=myserver,Type=
    WLDFHarvesterRuntime,WLDFRuntime=WLDFRuntime//TotalSamplingTime}
      &gt;= 100
      AND
      ${mydomain:Name=myserver,Type=
          ServerRuntime//OpenSocketsCurrentCount} &gt; 0)
      OR
      ${mydomain:Name=WLDFWatchNotificationRuntime,ServerRuntime=
        myserver,Type=WLDFWatchNotificationRuntime,
        WLDFRuntime=WLDFRuntime//Enabled} = true
      OR
      ${myCustomDomain:Name=myCustomMBean3//State} =
          'active')
      </rule-expression>
      <severity>Warning</severity>
      <alarm-type>AutomaticReset</alarm-type>
      <alarm-reset-period>10000</alarm-reset-period>
      <notification>mySMTPNotif</notification>
    </watch>

  </watch-notification>
</wldf-resource>
```

Configuring Log Watches

- Use Log watches to monitor the occurrence of specific messages and/or strings in the server log.
- Watches of this type are triggered as a result of a log message containing the specified data being issued

– <log-watch-severity> specifies the threshold severity for a log watch to be evaluated further

– You can use

<rule-expression> to filter for specific error codes or conditions

```
<wldf-resource xmlns="http://www.bea.com/ns/weblogic/weblogic-d
  <name>mywldf1</name>

  <watch-notification>
    <enabled>true</enabled>
    <log-watch-severity>Info</log-watch-severity>

    <watch>
      <name>myLogWatch</name>
      <rule-type>Log</rule-type>
      <rule-expression>MSGID='BEA-000360'</rule-expression>
      <severity>Info</severity>
      <notification>myMailNotif2</notification>
    </watch>

    <smtp-notification>
      <name>myMailNotif2</name>
      ...
    </smtp-notification>

  </watch-notification>
</wldf-resource>
```

Configuring Notifications

Configuring SMTP Notifications

Configuring Notifications

- Simple Mail Transfer Protocol (SMTP) notifications are used to send messages (e-mail) over the SMTP protocol in response to the triggering of an associated watch
- In order to use SMTP notifications, you must first configure the SMTP session for the domain
- An optional subject and/or body can be provided using sub-elements <subject> and <body> respectively. If these are not provided, they will be defaulted.

```
<!-- An SMTP notifications -->
<smtp-notification>
    <name>myMailNotif</name>
    <enabled>true</enabled>
    <mail-session-jndi-name>myMailSession</mail-session-jndi-name>
    <subject>This is a harvester alert</subject>
    <recipient>username@emailservice.com</recipient>
    <recipient>demo.user@demo.domain.com</recipient>
</smtp-notification>
```

Configuring JMX Notifications

Configuring Notifications

- For each defined JMX notification, WLDF issues JMX events (notifications) whenever an associated watch is triggered.
- Applications can register a notification listener with the server's WLDFWatchJMXNotificationRuntimeMBeans to receive all notifications and filter the provided output.
- You can also specify a JMX "notification type" string that a JMX client can use as a filter

```
<!-- A JMX notification -->
<jmx-notification>
    <name>myJMXNotif</name>
    <notification-type>SOMETHING_BROKE</notification-type>
</jmx-notification>
```

Configuring JMS Notifications

Configuring Notifications

- JMS Notifications can be sent to either a Queue or Topic
- You must specify the Connection Factory and Destination JNDI Names
- For this to work properly you must have a valid JMS configuration for the targets where the DMS is deployed

```
<jms-notification>
    <name>myJMSTopicNotif</name>
    <destination-jndi-name>MyJMSTopic</destination-jndi-name>
    <connection-factory-jndi-name>weblogic.jms.ConnectionFactory</connection-factory-jndi-name>
</jms-notification>

<jms-notification>
    <name>myJMSQueueNotif</name>
    <destination-jndi-name>MyJMSQueue</destination-jndi-name>
    <connection-factory-jndi-name>weblogic.jms.ConnectionFactory</connection-factory-jndi-name>
</jms-notification>
```

Diagnostic Images

Diagnostic Image Capture & Archives

Configuring and Using the WebLogic Diagnostic Framework

- **Diagnostic Image Capture** - Creates a diagnostic snapshot from the server that can be used for post-failure analysis.
- **Diagnostic Archive** - Captures and persists data events, log records, and metrics from server instances and applications

```
<domain>
  <server>
    <name>myserver</name>
    <server-diagnostic-config>
      <image-dir>logs/diagnostic_images</image-dir>
      <image-timeout>3</image-timeout>
      <diagnostic-store-dir>data/store/diagnostics</diagnostic-store-dir>
      <diagnostic-data-archive-type>FileStoreArchive</diagnostic-data-archive-type>
    </server-diagnostic-config>
  </server>
  ...
</domain>
```

Diagnostic Image Information Captured

Configuring and Using the WebLogic Diagnostic Framework

- An image is captured as a single file for the entire server. Each image has a unique name, as follows:
  - diagnostic_image_domain_server_yyyy_MM_dd_HH_mm_ss

- The most common sources of a server state are captured in a diagnostic image capture, including:
  - Configuration
  - Log cache state
  - Java Virtual Machine (JVM)
  - Work manager state
  - JNDI state
  - Harvestable data

Configuring Image Notifications

Configuring Notifications

- An image notification causes a diagnostic image to be generated in response to the triggering of an associated watch.
- Configuration:
  - The directory name relative to the Server directory where images will be generated,
  - Image file names are generated using the current timestamp - for example, diagnostic_image_myserver_2005_08_09_13_40_34.zip
  - The lockout period determines the number of seconds that must elapse before a new image can be generated after the last one.

```
<!-- An Image notification -->
<image-notification>
    <name>myImageNotif</name>
    <enabled>true</enabled>
    <image-lockout>2</image-lockout>
    <image-directory>images</image-directory>
</image-notification>
```

D E M O N S T R A T I O N

Questions?