Oracle® Light Weight Availability Collection Tool User Guide

For Software Release 3.3

Part No: 820–4693
July 2010

ORACLE

100810@24661

Contents

Preface 5

1 **Introduction** 9
Architecture 9
Availability Data Collection and Online Reporting 10
Offline Reporting at Oracle 10
Understanding Availability Datagram Structure 11
Important Fields in the Availability Datagram 12
Capabilities 13

2 **Oracle Lightweight Availability Collection Tool Utilities** 15
`tictimed` Utility 15
`logtime` Utility 16
`ltreport` Utility 16

3 **Installing the Oracle Lightweight Availability Collection Tool** 17
Installation Procedure 17
Downloading the Services Tools Bundle 17
Installing the Oracle Lightweight Availability Collection Tool 18
Configurable Parameters 21

4 **Oracle Lightweight Availability Collection Tool Use Cases** 25
Setting Up Cause Codes 25
cause-code 25
`ltreport` 27

5 Working with the Oracle Lightweight Availability Collection Tool 29
Graphical User Interface 29
Xalan Installation 29
Troubleshooting 34
Oracle Lightweight Availability Collection Tool Error Messages 35

Preface

The *Oracle Lightweight Availability Collection Tool User Guide* is intended for users of the Oracle Lightweight Availability Data Collection Tool. This user guide contains description of the product and its modules, installation information, and provides some troubleshooting guidelines.

Who Should Use This Book

This guide is intended for users of the Oracle Lightweight Availability Collection Tool.

Related Books

The following books contain additional information about the Oracle Lightweight Availability Collection Tool:

- *Oracle Lightweight Availability Collection Tool Release notes*
- *Oracle Lightweight Availability Collection Tool FAQ*

Related Third-Party Web Site References

Third-party URLs are referenced in this document and provide additional, related information.

Note – Oracle is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products, or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused or alleged to be caused by or in connection with use of or reliance on any such content, goods, or services that are available on or through such sites or resources.

Documentation, Support, and Training

See the following web sites for additional resources:

- Documentation (`http://docs.sun.com`)
- Support (`http://www.oracle.com/us/support/systems/index.html`)
- Training (`http://education.oracle.com`) – Click the Sun link in the left navigation bar.

Oracle Welcomes Your Comments

Oracle welcomes your comments and suggestions on the quality and usefulness of its documentation. If you find any errors or have any other suggestions for improvement, go to `http://docs.sun.com` and click Feedback. Indicate the title and part number of the documentation along with the chapter, section, and page number, if available. Please let us know if you want a reply.

Oracle Technology Network (`http://www.oracle.com/technetwork/index.html`) offers a range of resources related to Oracle software:

- Discuss technical problems and solutions on the Discussion Forums (`http://forums.oracle.com`).
- Get hands-on step-by-step tutorials with Oracle By Example (`http://www.oracle.com/technology/obe/start/index.html`).
- Download Sample Code (`http://www.oracle.com/technology/sample_code/index.html`).

Typographic Conventions

The following table describes the typographic conventions that are used in this book.

TABLE P–1 Typographic Conventions

Typeface	Meaning	Example
`AaBbCc123`	The names of commands, files, and directories, and onscreen computer output	Edit your `.login` file. Use `ls -a` to list all files. `machine_name% you have mail.`
`AaBbCc123`	What you type, contrasted with onscreen computer output	`machine_name%` **`su`** `Password:`
aabbcc123	Placeholder: replace with a real name or value	The command to remove a file is `rm` *filename.*

TABLE P–1 Typographic Conventions *(Continued)*

Typeface	Meaning	Example
AaBbCc123	Book titles, new terms, and terms to be emphasized	Read Chapter 6 in the *User's Guide*. A *cache* is a copy that is stored locally. Do *not* save the file. **Note:** Some emphasized items appear bold online.

Shell Prompts in Command Examples

The following table shows the default UNIX system prompt and superuser prompt for shells that are included in the Oracle Solaris OS. Note that the default system prompt that is displayed in command examples varies, depending on the Oracle Solaris release.

TABLE P–2 Shell Prompts

Shell	Prompt
Bash shell, Korn shell, and Bourne shell	`$`
Bash shell, Korn shell, and Bourne shell for superuser	`#`
C shell	`machine_name%`
C shell for superuser	`machine_name#`

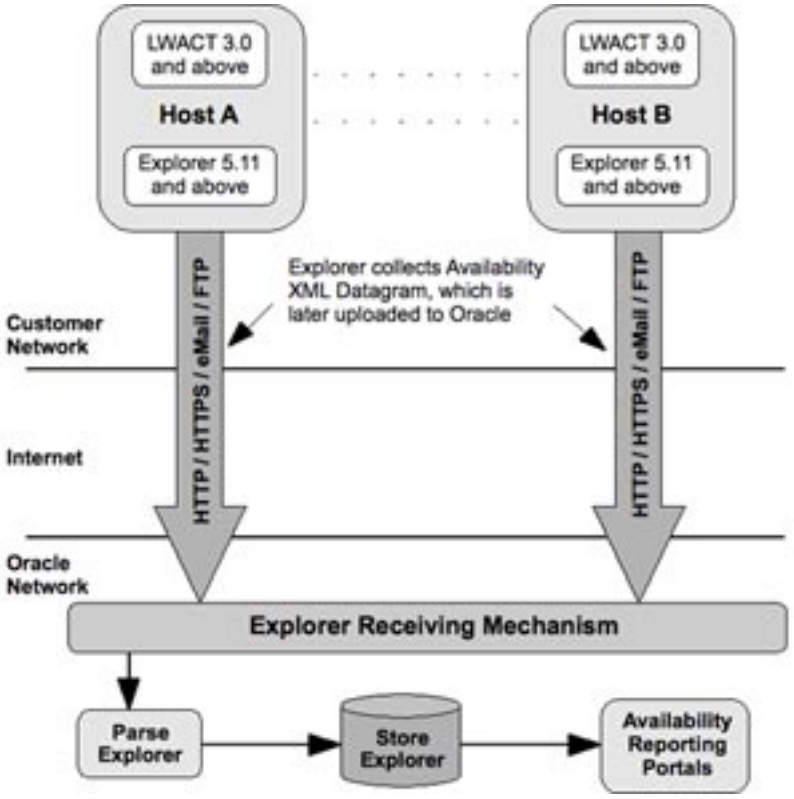
◆ ◆ ◆ CHAPTER 1

Introduction

The Oracle Lightweight Availability Collection Tool is a standalone product that collects availability data. To collect this data, you install it on each Solaris operating system (OS) running on a host. Upon installation, the Oracle Lightweight Availability Collection Tool starts recording the system availability events, such as boot, panic, and halt. You can choose to transport these availability events back to Oracle through the Oracle Explorer Data Collector. This telemetric data is used by Oracle to track system availability and in turn helps improve Oracle products and services. At present the single point solution for system availability data collection is the Oracle Lightweight Availability Collection Tool (LWACT) and has replaced the Configuration and Service Tracker (CST), which has been "End of Service Lifed (EOSLd)."

Architecture

This section explains the architecture. The following graphic shows the end-to-end data flow of the Oracle Lightweight Availability Collection Tool.



Availability Data Collection and Online Reporting

Upon installation of the Oracle Lightweight Availability Collection Tool on the monitored host, the tool spawns a daemon (`tictimed`) that continuously monitors and collects the availability status of the host. This collected availability data is stored in the form of an XML file. The Oracle Lightweight Availability Collection Tool's reporting utility (ltreport) can be used to generate and view command line interface (CLI) based reports from this file. The tool also provides a few XSL sheets to generate HTML-based reports from the datagram (for more information, see "`ltreport` Utility" on page 16).

Offline Reporting at Oracle

The Availability datagram is picked-up and transported back to Oracle by the Oracle Explorer Data Collector and is stored in Oracle's database. This data is used by Oracle to improve its products and is also available for account managers to communicate improvement opportunities with customers.

Understanding Availability Datagram Structure

The availability data collected by the Oracle Lightweight Availability Collection Tool is stored in the form of datagram within the file system of the monitored host. The availability data is embedded in between XML tags. The Availability datagram can be broadly categorized into two sections:

- Monitored System Information
- Availability Data

The following is a sample of an Oracle Lightweight Availability Collection Tool Availability datagram file.

```
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<single_system_availability_results>
    <systemInfo>
       <hostName>bs6-s0</hostName>
       <hostId>83254cb1</hostId>
       <zoneName>global</zoneName>
       <timeZone>US/Mountain</timeZone>
       <sysSerialNumber>unknown</sysSerialNumber>
       <OSName>SunOS</OSName>
       <OSVersion>5.10</OSVersion>
       <cpuArchitecture>sparc</cpuArchitecture>
       <productType>Serverblade1</productType>
       <lwactVersion>3.1</lwactVersion>
    </systemInfo>
  <event type="epoch" utc="1207784519" timeStamp="Wed Apr  9 17:41:59 2008"
    up="0" dwnPlnd="0" dwnUnplnd="0" dwnUndef="0" cksum="13c8" />
  <event type="boot" utc="1207784519" timeStamp="Wed Apr  9 17:41:59 2008"
   up="76820" dwnPlnd="0" dwnUnplnd="0" dwnUndef="0" cksum="13e4" />
  <event type="panic" utc="1207861339" timeStamp="Thu Apr 10 15:02:19
    2008 -06:00" up="0" dwnPlnd="0" dwnUnplnd="1" dwnUndef="0" L1causeCode="Unplanned"
    L2causeCode="Undefined" L3causeCode="Undefined" wasPlanned="2" cksum="2708" />
  <event type="boot" utc="1207861340" timeStamp="Thu Apr 10 15:02:20 2008 -06:00"
   up="8931" dwnPlnd="0" dwnUnplnd="0" dwnUndef="0" cksum="143b" />
  <event type="time" utc="1207870271" timeStamp="Thu Apr 10 17:31:11 2008 -06:00"
   up="85751" dwnPlnd="0" dwnUnplnd="1" dwnUndef="0" elapsed="85752" totAvail="99.999"
    adjAvail="99.999" cksum="1c95" />
</single_system_availability_results>
```

Note – In this sample datagram file, the *epoch* and *boot* events do not have a time zone offset. This might happen if you upgrade the Oracle Lightweight Availability Collection Tool from a pre-3.0 version to a later one. The latest Oracle Lightweight Availability Collection Tool will always contain time zone offset information in the event timestamps.

The information collected in between the tags <systemInfo> and </systemInfo> constitutes the system information section. This section provides details about the monitored host, such as the following:

- hostname

- hostid
- zonename (if present)
- timezone of the host
- system serial number (if known)
- OS name and version
- CPU architecture (sparc/X86)
- product type
- the version of the Oracle Lightweight Availability Collection Tool installed on the host

The Availability Data section contains the availability events (boot, epoch, halt, panic) and its corresponding timestamp. All the availability calculations are done based on the data collected under this section.

Important Fields in the Availability Datagram

This section identifies and describes the important fields of the Availability datagram (listed here in alphabetical order).

Field Name	Definition
Adjusted availability	Represented in percentage as: ((Total uptime + Total Planned downtime) /Total elapsed time) * 100

Note – Planned downtime is considered as uptime in this instance; hence, the term *adjusted availability*.

Downtime

The duration during which the host was out of run level 3 is considered as *downtime* (that is, the difference in coordinated universal time (UTC) between the outage event and its corresponding boot event). Downtime is recorded as a part of the outage event (panic/halt). It is decided by the *wasPlanned* field. *wasPlanned* can be one of the following designations:

- Undefined (value of 0)
- Planned (value of 1)
- Unplanned (value of 2)

In the sample datagram (above), event #2 is a panic event, and event #3 is its corresponding boot event; the difference in UTC of event #3 and event #2 is the downtime.
Therefore, downtime = 1207861340 - 1207861339 (= 1 sec)

Since the *wasPlanned* flag is 2, the downtime is marked against the field *dwnUnplnd* (Unplanned downtime)

Total availability Represented in percentage as: (Total uptime/Total elapsed time) * 100

Types of Events The following types of events are recorded in the Availability datagram by the Oracle Lightweight Availability Collection Tool:

- epoch

  Marks the beginning of event tracking. It is recorded only once in the Availability datagram (at the inception). The UTC of this event marks the inception time of the Oracle Lightweight Availability Collection Tool on the monitored host.

- boot

  Whenever the host returns to run level 3, a *boot* event is recorded in the datagram along with the corresponding timestamp.

- halt

  Whenever the host leaves run level 3 to any other level, a *halt* event is created with the time of halt being the time the host left run level 3.

- panic

  If the host encounters an un-natural downing such as system crash, upon the subsequent boot of the host (that is, a return to run level 3), a panic event is recorded where the time of the *panic* event is the time at which the Oracle Lightweight Availability Collection Tool stopped running.

- time

  Indicates the last recorded UTC for offline reporting. This event contains the consolidated uptime and downtime information. It also reports the elapsed time (measured as the duration in UTC that the Oracle Lightweight Availability Collection Tool is monitoring this host since inception). Apart from this information, the *time* event also reports system availability in two forms: Total availability and Adjusted availability.

Uptime The difference in UTC between the current outage event and the last event before it, which would be a *boot event*, is measured as uptime.

In the sample datagram (above), if the uptime field in event #1 (boot event) is calculated as the difference in UTC between event #3 and event #2
uptime = 1207861339 - 1207784519 (= 76820 secs)

Capabilities

Key capabilities of the Oracle Lightweight Availability Collection Tool are as follows:

- Supported on Solaris 10 Containers/Zones
- Supports both SPARC and x86/x64 platforms
- Stores the data in universally accepted datagram format
- Does not generate network traffic

- Tracks boot, halt, and panic events to a granularity of one second
- Facilitates segregation of planned, unplanned, or undefined downtime for finer tracking
- Deploys easily in Sun package format
- Is very Lightweight on system resources
- Facilitates both online and offline reporting

Basic reporting functionality is provided through the `ltreport` command line interface, which is part of the Oracle Lightweight Availability Collection Tool package. Additionally, the datagram enables a wide range of reporting options that are independent of any reporting database or applications. This enables availability reports to be generated on-site or through any report generating portals at Oracle.

Browser-based graphical reporting can also be performed at a system level. To enable such report generation capabilities, a predefined set of XSL style sheets are provided when the Oracle Lightweight Availability Collection Tool is installed. An XSL translator is required to generate the HTML reports using these style sheets. The XSL translator is not part of the Oracle Lightweight Availability Collection Tool, see "Xalan Installation" on page 29 for more information about an XSL translator.

◆ ◆ ◆ CHAPTER 2

Oracle Lightweight Availability Collection Tool Utilities

The Oracle Lightweight Availability Collection Tool consists of the following three main binary utilities:

- "`tictimed` Utility" on page 15
- "`logtime` Utility" on page 16
- "`ltreport` Utility" on page 16

`tictimed` Utility

`tictimed` is a heartbeat daemon for the Oracle Lightweight Availability Collection Tool. It changes the modified time (the UTC) of the log file once a second, and updates the `time` event once a minute. This utility starts automatically via the `/etc/rc2.d/S95lwact` script. An entry in `/etc/inittab` makes sure that it is re-spawned even if it is killed or it crashes for an unknown reason. It writes the system halts, panic, and boot records to a log file to track system availability. If the update file (`lwact.update`) is present under the update directory, the `tictimed` also modifies the event to update cause codes.

`tictimed` captures the following five event types:

1. `epoch` - the beginning of event tracking
2. `boot` - UTC when system leaves run-level 2
3. `halt` - UTC when system exits run-level 3
4. `panic` - a boot event without a preceding halt recorded. Last modified time of the log file is used as the panic UTC
5. `time`- the last recorded UTC for off-line reporting

The Oracle Lightweight Availability Collection Tool has *init* scripts which are invoked by the system during run level changes. If the you try to invoke these scripts manually, it logs the appropriate info log to `/var/adm/messages`:

- LWACT is started — Indicates that a user has used `/etc/init.d/lwact` script to re-initialize the init tab
- LWACT is going down — Indicates a user has stopped the Oracle Lightweight Availability Collection Tool using the `/etc/init.d/lwact` script. This causes the `tictimed` daemon to respawn and re-write the lock file under `/var/spool/locks` with the new `tictimed pid`

`logtime` Utility

The `logtime` utility is used by the *root* user to update the cause code for events. This utility is also used by the system to create and update `boot` and `halt` events. Using `-M` option of this utility, only the *root* user can modify the cause code for the `halt` and `panic` events, whereas `-B` and `-H` options are used by the system (host process, such as `init`) itself. `logtime` can be executed in interactive and non-interactive mode. In the interactive mode, the user does not need to provide the cause code string; whereas in the non-interactive mode it does require the event number and cause code string (comma separated, triplet—For more information, see "Setting Up Cause Codes" on page 25).

`ltreport` Utility

`ltreport` is a command line, binary executable reporting tool that reads the datagram and calculates the system availability. The output is written to `stdout`.

`ltreport` calculates the following two availability figures:

- Total - Total availability is a raw calculation whereby total uptime is divided by total elapsed time.
- Adjusted - Adjusted availability is the sum of total uptime and total planned down time, divided by total elapsed time. Here, any planned downtime is accounted as uptime of the system.

`ltreport` reports three downtime categories:

- Planned
- Unplanned
- Undefined

For more information about these categories, see "`ltreport`" on page 27.

◆ ◆ ◆ CHAPTER 3

Installing the Oracle Lightweight Availability Collection Tool

The Oracle Lightweight Availability Collection Tool is a package that has to be installed on each instance of Solaris. It is bundled with Oracle Services Tools Bundle (STB) for Sun Systems. By downloading STB and running ./install_stb.sh, you can add the Oracle Lightweight Availability Collection Tool with other deliverables in the STB.

Installation Procedure

The Oracle Lightweight Availability Collection Tool is delivered as part of the STB. This section explains how to download STB and install the Oracle Lightweight Availability Collection Tool.

Downloading the Services Tools Bundle

The Oracle Lightweight Availability Collection Tool must be installed through STB and is made available via its download link. Use the following procedure to download the latest Services Tools Bundle:

1. Go to the STB site at: `http://www.sun.com/service/stb/index.jsp` and click the *Software Download and Documentation* link in the Resources section.
2. In the drop-down lists, select the appropriate Platform and Language for your download.
3. Review the STB License Agreement and mark the *I agree* check box to proceed with downloading.

   The Sun Download Center might require you to log in before proceeding.
4. Click install_stb.sh to download the installer.

To finish the installation, complete the instructions in the "Installing the Oracle Lightweight Availability Collection Tool" on page 18 section.

Installing the Oracle Lightweight Availability Collection Tool

To install the Oracle Lightweight Availability Collection Tool using STB, complete the following steps when requested during the installation process:

Note – Though questions not pertaining specifically to the Oracle Lightweight Availability Collection Tool are asked, this section does not address these questions. You must decide whether you want these tools installed and answer the questions accordingly.

At the beginning of the installation, the following is displayed on your screen:

```
-bash-3.00# ./install_stb.sh
Services Tools Bundle(STB) v6.0 Installer

Checksumming...

List of Components and Corresponding Selection

1. Install SNEEP Tool v2.9 ? (y/n) y
   Already Installed Sneep Tool has Version (2.9)

   Sneep Tool details can be found at <http://www.sun.com/sneep> and local
   system documentation reference is available at /opt/SUNWsneep/Docs

2. Install Service Tags v1.1.5,REV=2009.09.23.10.58 ? (y/n) y
   Already Installed Service Tags has Version (1.1.5,REV=2009.09.23.10.58)

   Service Tags details can be found at
   <http://wikis.sun.com/display/SunInventory/FAQ> and
   <http://wikis.sun.com/display/SunInventory/Discovery+and+Registration>

3. Install Explorer v6.5,REV=2010.07.02.12.51 ? (y/n) y

   Explorer details can be found at
   <http://docs.sun.com/app/docs/coll/1554.2> and local system
   documentation reference is available at /opt/SUNWexplo/doc

4. Install Lightweight Availability Collection Tool v3.3 ? (y/n) y

   Lightweight Availability Collection Tool details can be found at
   <http://docs.sun.com/app/docs/coll/1811.1>

Would you like to (I)nstall, (X)tract component selections, or
(E)xit ? I(default)
```

1. Accept the default: **I**

   The installation proceeds with the default options:

   ```
   Would you like to (I)nstall, (X)tract component selections, or
   (E)xit ? I(default)
   ```

```
STB is installing all selected modules and their dependencies.
Details of this will be in /var/log/install_stb-v6.0.log
Please wait.....

Installing Oracle Sneep .....

---- Already Installed Sneep Packages has current Version (2.9)

All sneep data sources are consistent.

Installing Service Tags and Product Serial Number Package .....

---- Checking Service Tags dependency packages...

---- Service Tags dependency check passed

---- Already Installed Product Serial Number Package has current Version
(1.1.4,REV=2008.04.25.10.21)

---- Already Installed Service Tags Packages has current Version
(1.1.5,REV=2009.09.23.10.58)

---- Already Installed Hardware Service Tag Registration Package has
current Version (1.0,REV=2009.09.23.11.02)

Installing Oracle Explorer Data Collector .....

Modifying /etc/opt/SUNWexplo/xscfinput.txt
Modifying /etc/opt/SUNWexplo/tapeinput.txt
Modifying /etc/opt/SUNWexplo/t3input.txt
Modifying /etc/opt/SUNWexplo/srscinput.txt
Modifying /etc/opt/SUNWexplo/se6920input.txt
Modifying /etc/opt/SUNWexplo/se6320input.txt
Modifying /etc/opt/SUNWexplo/se3kinput.txt
Modifying /etc/opt/SUNWexplo/scinput.txt
Modifying /etc/opt/SUNWexplo/saninput.txt
Modifying /etc/opt/SUNWexplo/ipmiinput.txt
Modifying /etc/opt/SUNWexplo/indyinput.txt
Modifying /etc/opt/SUNWexplo/ilomsnapshotinput.txt
Modifying /etc/opt/SUNWexplo/ilominput.txt
Modifying /etc/opt/SUNWexplo/b1600switchinput.txt
Modifying /etc/opt/SUNWexplo/b1600input.txt
Modifying /etc/opt/SUNWexplo/alominput.txt
Modifying /etc/opt/SUNWexplo/acinput.txt
Modifying /etc/opt/SUNWexplo/Tx000input.txt
Modifying /etc/opt/SUNWexplo/1280input.txt
Copyright (c) 2010, Oracle and/or its affiliates. All rights reserved.
All sneep data sources are consistent.
Installation of Oracle Explorer Data Collector <6.5,REV=2010.07.02.12.51>
was successful

Installing Oracle Lightweight Availability Collection Tool .....
It may take a few minutes to complete postinstall..
It may take a few minutes to complete postinstall..
Installation of Lightweight Availability Collection Tool <3.3> was
successful

STB v6.0 installation is complete...
```

Note – In order to leverage complete functionality, the Oracle Lightweight Availability Collection tool requires Explorer at a minimum release of 6.0. STB will fail to install the tool if this minimum requirement is not met.

2. Verify that the Oracle Lightweight Availability Collection Tool completed successfully by reviewing the following conditions:
   - The output of the `pkginfo -l SUNWlwact` command reflects *completely installed* in the *STATUS* field.
   - Immediately upon successful installation of the Oracle Lightweight Availability Collection Tool package, the tool starts the `tictimed` daemon. This is the daemon responsible for continuous monitoring of the availability status of the system. You can check for the existence of this daemon by executing: `/usr/bin/ps -eaf | grep tictimed`.
   - `/etc/inittab` contains a new entry for `tictimed` under the ID *LT*.
   - Availability datagram is created in the default location (defined by the configurable parameter *LOGDIR*).

As soon as the Oracle Lightweight Availability Collection Tool package is installed, it will kick-off the process `tictimed` which monitors availability related events. You can check this with:

```
# ps -eaf | grep tictime
              root  4817     1  0 16:47:43 ?        0:00 /usr/sbin/tictimed
```

Any availability-related event is logged by the `tictime` daemon to `/var/log/<hostid>.lwact.xml`

A local report based on this file can be viewed in a user friendly format with the following command:

```
# /usr/bin/ltreport -v
```

The file can also be viewed in its raw XML format using the following command:

```
# /usr/bin/ltreport -x
```

To prevent tampering with the file, each event that is logged has a checksum. If the file is manipulated, the checksum will become invalid and a message will be logged to `/var/adm/messages`.

Configurable Parameters

There are a set of configurable parameters for the Oracle Lightweight Availability Collection Tool, which enables you to set default actions based on local site policies. The tool configures itself using the parameters defined in `/etc/default/lwact` file. The following parameters are configurable:

- *LOGDIR* parameter

  LOGDIR specifies the path where availability data (`hostid.lwact.xml`) will be collected. By default, it is collected in `/var/log`. You can change the value to a different path and the tool will start logging the availability metrics into this new path after the tool is restarted. To retain the availability data collected thus far, you must ensure that the log file is manually copied into the new location; otherwise, the tool will start logging availability data in the new location afresh and the old data will be lost.

  Note – Before you restart the tool, be sure you retain the availability data already collected. To retain this data, manually copy the log file to the new location. If you do not copy the log file, the new data will be logged to the new location, but the old data will not be carried over to this new location when the tool restarts.

- *BACKUP* parameter

  BACKUP specifies the path where the Oracle Lightweight Availability Collection Tool will store a backup copy of the log file. By default this entry has the path set to `/var/tmp` and is commented; therefore, no backup will be stored. If you want a backup, you can un-comment the entry and change the path to your preferred location. The backup file will be found under the path you specify.

- *UPDATE* parameter

  UPDATE specifies the path where the `lwact.update` file can be found. By default, the path is `/var/tmp`. You can modify this path.

  `lwact.update` is a feature provided by the Oracle Lightweight Availability Collection Tool to auto-update predefined cause codes for any outage. You can use this feature to update a cause code to a single or bulk of hosts for an outage.

  For example, an outage might have occurred on number of hosts within your site due to a power failure. Hence, you might want to update a common cause code across all these hosts for that particular outage. Instead of manually updating the cause code for that event after the outage on each host, you can push this `lwact.update` file into all these hosts soon after this activity is carried out. The Oracle Lightweight Availability Collection Tool will automatically pick the cause codes mentioned in the `lwact.update` file and set the cause codes to the outage event accordingly. After completing this update, the file is automatically deleted. By using this feature, you no longer need to manually log into each of the hosts to update the cause code after an outage occurs.

The structure of the `lwact.update` file is as follows:

```
# This file contains the cause codes for the outage
<L1CauseCodeIndex>, <L2CauseCodeIndex>, <L3CauseCodeIndex>
```

For example:

```
$ cat lwact.update
1,2,7
```

Based on the file in this example, after the outage, the tool will set the cause codes as follows: L1=1, L2=2, L3=7

- *L1CC*, *L2CC*, *L3CC* parameters

  The *L1CC*, *L2CC*, *L3CC* enable you to define default cause codes for L1, L2 and L3.

  By default, the Oracle Lightweight Availability Collection Tool logs *halt* event cause codes as:

  ```
  L1=Planned
  L2=Undefined,
  L3=Undefined
  ```

  By default, it logs the *panic* event's cause codes as:

  ```
  L1=Unplanned
  L2=Undefined
  L3=Undefined
  ```

  The structure of the *L1CC*, *L2CC*, *L3CC* parameters is as follows:

  ```
  L1CC=<L1CauseCodeString>
  L2CC=<L2CauseCodeString>
  L3CC=<L3CauseCodeString>
  ```

  By default, there are no entries for cause codes in this file. So L1 cause code for halt and panic events are logged as *Planned* and *Unplanned* respectively and L2 and L3 cause codes are logged as *Undefined*. If cause codes are explicitly set for different levels, they override the default cause codes for outage events (both halt and panic). (For specific information about setting cause code values, see "Setting Up Cause Codes" on page 25.)

  Note – If any of the *L1CC, L2CC, L3CC* values are not valid, then the Oracle Lightweight Availability Collection Tool detects this and logs a corresponding log message in `/var/adm/message` and sets the invalid cause code entry as *Undefined*.

Upon installation, the configurable parameters in the `/etc/default/lwact` file have the following default values:

```
LOGDIR=/var/log
#BACKUP=/var/tmp
UPDATE=/var/tmp
```

Note – For any changes to take effect, you must restart the Oracle Lightweight Availability Collection Tool.

◆ ◆ ◆ CHAPTER 4

Oracle Lightweight Availability Collection Tool Use Cases

This chapter describes the Oracle Lightweight Availability Collection Tool use cases and explains how to setup the cause codes.

Setting Up Cause Codes

There are two cause codes you can setup:

- "cause-code" on page 25
- "ltreport" on page 27

cause-code

You can set the cause-codes for the *outages post event*. This enables the user to have more control in maintaining the availability metrics for the host. There are two modes you can use for a cause code:

Note – The following cause code example is the same in both the interactive and non-interactive modes.

- Interactive mode

  In interactive mode, a list of all available cause codes for that level is displayed from which you can choose the appropriate code. The following shows the information displayed on the screen when using the Interactive mode.

```
## Interactive mode##
$logtime -M 2

Level-1 Cause Codes
- - - - - - - - - -
   0) - Undefined
```

```
   1) - Planned
   2) - Unplanned

Section? 1

Level-2 Cause Codes
- - - - - - - - - -
   0) - Undefined
   1) - System Management
   2) - System Hardware
   3) - System Software
    4) - Application
   5) - Network
   6) - Environmental

Selection? 4

Level-3 Cause Codes
- - - - - - - - - -
   0) - Undefined
   1) - Patch Installation
   2) - Application Upgrade
   3) - Dabase Maint
   4) - App/Db Schema Mgnt
   5) - Configuration
   6) - Batch Processing
   7) - Application Install
   8) - Application Removal
   9) - Other

Selection ? 8
Modification of event was successful.
```

- Non-interactive mode

  In non-interactive mode, you can set the cause codes using the CLI by invoking `logtime`. The following shows the command line display when using the non-interactive mode:

```
## Non-Interactive mode##
## Same functionality of setting the codes L1=1, L2=4, L3=8 is achieved by
non-interactive invocation of logtime##

$ logtime -M 2,1,4,8
<Modification of event was Successful.
```

For example, you should mark a weekly planned upgrade or a planned power outage as a Planned outage with the appropriate reason behind it. In such cases, you can use the `logtime` utility to modify the originally logged outage event and attach a suitable cause code to it. You can attach up to three levels of cause codes for an outage event.

1. You can modify any outage event that has already occurred by invoking the `logtime` as shown:

   `logtime -M <event# that has to be modified> <L1,L2,L3 causecodes>`

   This can be done interactively or non-interactively, as described above.

2. There is also another invocation of `logtime` with `-L` option. This allows you to modify the cause code for the last occurred outage.

```
logtime -M -L <L1, L2, L3 cause codes>
```

3. You can get a list of all permissible cause codes for each level by invoking the `logtime` as shown:

```
logtime -M
```

ltreport

You can utilize the CLI ltreport to generate simple availability reports and view them locally for a single host. It can also be used to generate and view availability reports of other hosts' data using the `-i` option for one host at a time. Thus, this utility facilitates an offline reporting mechanism. The following are examples of various invocations of the `ltreport`:

```
bash-3.00# ltreport -v

     Hostname: bs6-s0 Hostid: 8325cb1    Zone:global
     -----------------EVENTS-----------------
     Event[  0]: epoch    Thu Apr 24 08:36:44 2008 -06:00
     Event[  1]: boot   Thu Apr 24 08:36:44 2008 -06:00
Availability: 100.000% (total) 100.000% (adjusted)
   Monitored: 00d-00-00m-00      Since: Thu Apr 24 08:36:44 2008 -06:00s
      Uptime: 00d-00-00m-00
    Downtime: 00d-00h-00m-00s
           Planned: 00d-00h-00m-00s
         Unplanned: 00d=00h-00m-00s
         Undefined: 00d-00h-00m-00s

bash-3.005# ltreport -x
<?xml version="1.0" encoding="UTF-8" standalone="no"?>
<single_system_availability_results>
    <systemInfo>
       <hostName>bs06-s0</hostName>
       <hostId>83254cb1</hostId>
       <zoneName>global</zoneName>
       <timeZone>US/Mountain</timeZone>
       <sysSerialNumber>3254CB1\uffff</sysSerialNumber>
       <OSName>SunOS</OSName>
       <OSVersion>5.10</OSVersion>
       <cpuArchitecture>sparc</cpuArchitecture>
       <productType>Serverblade1</productType>
       <lwactVersion>3.1</lwactVersion>
     </systemInfo>
  <event type="epoch" utc="1209047804" timeStamp="Thu Apr  24 08:36:44 2008 -06:00"
    up="0" dwnPlnd="0" dwnUnplnd="0" dwnUndef="0" cksum="1448" />
  <event type="boot" utc="1209047804" timeStamp="Thu Apr  24 08:36:44 2008 -06:00"
    up="556" dwnPlnd="0" dwnUnplnd="0" dwnUndef="0" cksum="143d" />
   <event type="time" utc="1209048360" timeStamp="Thu Apr  24 08:36:44 2008 -06:00"
up="556" dwnPlnd="0" dwnUnplnd="0" dwnUndef="0" elapsed="556"
totAvail="100.000" adjAvail="100.000" cksum="1c0e" />
</single_system_availability_results>
bash-2.05# ltreport -i
bash-2.05# cp /var/log/83cde40d.lwact.xml  /tmp/myhost.xml
bash-2.05# ltreport -i /tmp/myhost.xml
```

```
     Hostname: noyal Hostid: 83cde40d

Availability: 100.000% (total) 100.000% (adjusted)
   Monitored: 02d-20h-28m-03s  Since: Tue Apr  1 04:32:39 2008 -25200 ..isdst=1
      Uptime: 02d-20h-28m-03s
    Downtime: 00d-00h-00m-00s
           Planned: 00d-00h-00m-00s
         Unplanned: 00d-00h-00m-00s
         Undefined: 00d-00h-00m-00s
```

For information about installing an XML to HTML converter on your system and viewing this report in a browser, see "Graphical User Interface" on page 29.

◆ ◆ ◆ CHAPTER 5

Working with the Oracle Lightweight Availability Collection Tool

This chapter contains the following sections:

- "Graphical User Interface" on page 29
- "Troubleshooting" on page 34

Graphical User Interface

The Oracle Lightweight Availability Collection Tool comes with a browser-based GUI display that is used to display the recorded availability results so they are easy to understand. You must configure, and run a web server on the host where the Oracle Lightweight Availability Collection Tool is installed. You can then use simple web services to display the Availability datagram report in a web browser.

A predefined set of XSL style sheets are provided when the Oracle Lightweight Availability Collection Tool is installed. An XSL translator is required to translate the datagrams to HTML and to display it in the web browser.

A web services Perl script is provided as a reference application when the package is installed.

In order to display the availability reports in the GUI, you must have an XML to HTML converter. Xalan is one such software. It is available as a C based binary as well as java based jar file.

Xalan Installation

To install Xalan, decide whether you want to install the C based binary or the java based jar file, and complete the steps in the appropriate section:

- "Installation and Configuration of the C Based Xalan Software" on page 30
- "Installation and Configuration of the Java based Xalan" on page 33

Note – The prerequisite (common for both C and Java based Xalan) is that the Oracle Lightweight Availability Collection Tool and an Apache web server must be installed in the system. The Apache web server can be downloaded from `http://httpd.apache.org/download.cgi`.

Installation and Configuration of the C Based Xalan Software

To configure the GUI for the Oracle Lightweight Availability Collection tool, you need to have *Xalan* and *Xerces* binaries installed in the host.

Download and Unpack the Software

1. Download the latest Xalan-c software from `http://www.apache.org/dyn/closer.cgi/xml/xalan-c`.

   This URL provides links to different mirror sites. The binaries are available under the Binaries folder on any of these mirror sites. It can be downloaded in the form of *.tar.gz according to system specification (32 or 64 bit).

   For these instructions assume that the download path is `/export/lwact/installables/xalan` and that you are going to download the Xalan binary `Xalan-C_1_10_0-solaris_28_64-cc_62.tar.gz` for a 64 bit Solaris system.

2. To verify download of the `Xalan-c` software file to `/export/lwact/installables/xalan`, type the following commands:

```
bash-2.05# pwd
/export/lwact/installables/xalan
bash-2.05# ls
Xalan-C_1_10_0-solaris_28_64-cc_62.tar.gz
```

3. Unpack the archive to install Xalan in the system; type the following commands:

   Note – The path for binary Xalan is `/export/lwact/installables/xalan/Xalan-C_1_10_0-solaris_28-cc_62/bin` and the path for Xalan library is `/export/lwact/installables/xalan/Xalan-C_1_10_0-solaris_28-cc_62/lib`.

```
bash-2.05# gunzip Xalan-C_1_10_0-solaris_28_64-cc_62.tar.gz
bash-2.05# ls
Xalan-C_1_10_0-solaris_28_64-cc_62.tar
bash-2.05# tar -xvf Xalan-C_1_10_0-solaris_28_64-cc_62.tar
bash-2.05# ls
Xalan-C_1_10_0-solaris_28_64-cc_62
bash-2.05# cd Xalan-C_1_10_0-solaris_28_64-cc_62
bash-2.05# ls
KEYS LICENSE NOTICE bin/ include/ lib/ samples/
bash-2.05# cd bin
bash-2.05# ls
CompileStylesheet SerializeNodeSet StreamTransform UseStylesheetParam
DocumentBuilder SimpleTransform ThreadSafe Xalan
```

```
ExternalFunction SimpleXPathAPI TraceListen XalanTransform
ParsedSourceWrappers SimpleXPathCAPI TransformToXercesDOM XalanTransformerCallback
bash-2.05# pwd
/export/lwact/installables/xalan/Xalan-C_1_10_0-solaris_28-cc_62/bin
bash-2.05# cd ../lib
bash-2.05# pwd
/export/lwact/installables/xalan/Xalan-C_1_10_0-solaris_28-cc_62/lib
```

4. Download the latest `Xerces-c` library file from `http://www.apache.org/dyn/closer.cgi/xml/xerces-c`.

   This URL provides links to different mirror sites. The binaries are available under the Binaries folder of any mirror site. It can be downloaded in the form of *.tar.gz according to system specification (32 or 64 bit).

   For these instructions, assume that the download path is `/export/lwact/installables/xerces` and that you are going to download the Xerces binary `xerces-c_2_7_0-solaris_28-cc_62.tar.gz` for a 64 bit Solaris system.

5. Type the following commands to download the `Xerces-c` library file to `/export/lwact/installables/xerces`

   ```
   bash-2.05# pwd
   /export/lwact/installables/xerces
   bash-2.05# ls
   xerces-c_2_7_0-solaris_28-cc_62.tar.gz
   ```

6. Unpack the archive to install `Xerces-c` in the system; type the following commands:

   Note – The path for `xerces` library is `/export/lwact/installables/xerces/xerces-c-solaris_28-cc_62/lib`.

   ```
   bash-2.05# gunzip xerces-c_2_7_0-solaris_28-cc_62.tar.gz
   bash-2.05# ls
   xerces-c_2_7_0-solaris_28-cc_62.tar
   bash-2.05# tar -xvf xerces-c_2_7_0-solaris_28-cc_62.tar
   bash-2.05# ls
   xerces-c-solaris_28_64-cc_62
   bash-2.05# cd xerces-c-solaris_28_64-cc_62
   bash-2.05# ls
   LICENSE Readme.html credits.txt etc lib samples version.incl
   NOTICE bin doc include msg scripts
   bash-2.05# cd lib
   bash-2.05# ls
   libxerces-c.so libxerces-c.so.27.0 libxerces-depdom.so.27
   libxerces-c.so.27 libxerces-depdom.so libxerces-depdom.so.27.0
   bash-2.05# pwd
   /export/lwact/installables/xerces/xerces-c-solaris_28-cc_62/lib
   ```

Provide 755 Access Permissions

1. To provide 755 access permissions to the `/opt/SUNWlwact/xsl` directory, type the following command:

   ```
   bash-2.05# chmod -R 755  /opt/SUNWlwact/xsl
   ```

2. Type the following command to copy the directory `pacifico` from `/opt/SUNWlwact/misc/` to `/var/apache/htdocs/`

   ```
   bash-2.05# cp -rf /opt/SUNWlwact/misc/pacifico /var/apache/htdocs/
   ```

3. Type the following commands to provide 755 access permission to the `cgi-bin` directory:

   ```
   bash-2.05# chmod 755 /var/apache/cgi-bin
   ```

4. Type the following commands to provide 755 access permission to the `pacifico` directory:

   ```
   bash-2.05# chmod -R 755 var/apache/htdocs/pacifico
   ```

   The web browser-based Availability report for a single host is created by the `/opt/SUNWlwact/sdk/my_uptime.cgi` script, which is part of Oracle Lightweight Availability Collection Tool package installation.

Complete the Installation

1. Copy `my_uptime.cgi` file from `/opt/SUNWlwact/sdk/` to `/var/apache/cgi-bin/`

   ```
   bash-2.05# cp /opt/SUNWlwact/sdk/my_uptime.cgi /var/apache/cgi-bin/
   ```

2. Type the following command to provide 755 access permission to the file `/var/apache/cgi-bin/my_uptime.cgi`:

   ```
   bash-2.05# chmod 755 /var/apache/cgi-bin/my_uptime.cgi
   ```

3. Modify `/var/apache/cgi-bin/my_uptime.cgi` to include correct path for Xalan, Xerces libraries, the XSL path and the Xalan binary path:

   ```
   bash-2.05# vi /var/apache/cgi-bin/my_uptime.cgi
   ```

4. At the beginning of the *Variable Initialization* section, type the following statement to include the libraries for Xalan and Xerces:

   ```
   $ENV{'LD_LIBRARY_PATH'} =
   "/export/lwact/installables/xalan/Xalan-C_1_10_0-solaris_28-cc_62/lib:
   /export/lwact/installables/xerces/xerces-c-solaris_28-cc_62/lib:
   $ENV{'LD_LIBRARY_PATH'}";
   ```

5. Make sure that the input file is included properly by verifying value of *$input*:`$input="/var/log/$hostId.lwact.xml";`

   If you manually modify the path for the log file (`LOGDIR` in `/etc/default/lwact` file), the value of *$input* must be modified accordingly. The value set to the *$xalan* variable must be modified to point to Xalan in the system:

   ```
   $xalan="/export/lwact/installables/xalan/Xalan-C_1_10_0-solaris_28-cc_62/bin/Xalan";
   ```

6. Make sure that the variable *$xslt* points to `singleLWACT.xsl` in `/opt/SUNWlwact/xsl`: `$xslt="/opt/SUNWlwact/xsl/singleLWACT.xsl";`

7. Replace avoidance.central with the Fully Qualified Domain Name (FQDN) of the your host in the subroutine display_results():

   ```
   print "<meta http-equiv=\"Refresh\" content=60;
   URL=http://avoidance.central/cgi-bin/my_uptime.cgi\">\n";
   ```

8. To view the GUI-based Oracle Lightweight Availability Collection Tool availability report, type the following URL in the address field of your web browser:
   http://<FQDN>/cgi-bin/my_uptime.cgi

Installation and Configuration of the Java based Xalan

Download and Unpack the Software

1. Download the Xalan jar from http://www.apache.org/dyn/closer.cgi/xml/xalan-j

   This URL provides links to different mirror sites and the Xalan jar can be downloaded in the form of *.tar.gz.

   For these instructions assume that the download path is /export/lwact/installables/xalanand that you are going to download the Xalan xalan-j_2_7_1-bin-2jars.tar.gz.

2. Type the following commands to verify download of the Xalan jar to the /export/lwact/installables/xalan

```
bash-2.05# pwd
/export/lwact/installables/xalan
bash-2.05# ls
xalan-j_2_7_1-bin-2jars.tar.gz
```

3. To unpack the archive and install Xalan in the system, type the following command:

```
bash-2.05# gunzip xalan-j_2_7_1-bin-2jars.tar.gz

bash-2.05# ls
xalan-j_2_7_1-bin-2jars.tar
bash-2.05# tar -xvf xalan-j_2_7_1-bin-2jars.tar
bash-2.05# ls
xalan-j_2_7_1
bash-2.05# cd xalan-j_2_7_1
bash-2.05# ls
docs NOTICE.txt samples xalan.jar xml-apis.jar
LICENSE.txt readme.html serializer.jar xercesImpl.jar xsltc.jar
bash-2.05# pwd
/export/lwact/installables/xalan/xalan-j_2_7_1
```

Note – The path for xalan.jar is the following: /export/lwact/installables/xalan/xalan-j_2_7_1

Provide 755 Access Permissions

1. Copy the directory pacifico from /opt/SUNWlwact/misc/ to /var/apache/htdocs/

```
bash-2.05# cp -rf /opt/SUNWlwact/misc/pacifico /var/apache/htdocs/
```

2. Provide 755 access permission to the /opt/SUNWlwact/xsl

```
bash-2.05# chmod -R 755 /opt/SUNWlwact/xsl
```

3. Provide 755 access permission to the directory /var/apache/htdocs/pacifico

```
bash-2.05# chmod -R 755 /var/apache/htdocs/pacifico
```

Complete the Installation

The browser-based Availability report for single host is created by `/opt/SUNWlwact/sdk/my_uptime.cgi` script which is part of the Oracle Lightweight Availability Collection Tool package installation.

1. Copy `my_uptime.cgi` file from `/opt/SUNWlwact/sdk/ to /var/apache/cgi-bin/`

   ```
   bash-2.05# cp /opt/SUNWlwact/sdk/my_uptime.cgi /var/apache/cgi-bin/
   ```
2. Type the following command to provide 755 access permission to the file `/var/apache/cgi-bin/my_uptime.cgi`:

   ```
   bash-2.05# chmod 755 /var/apache/cgi-bin/my_uptime.cgi
   ```
3. Modify `/var/apache/cgi-bin/my_uptime.cgi` to include correct path for Xalan, the XSL path, and the Xalan binary path:

   ```
   bash-2.05# vi /var/apache/cgi-bin/my_uptime.cgi
   ```
4. Set the value of the *$xalan* variable to point to the path where the `xalan.jar` exists in the system:

   ```
   $xalan="/export/lwact/installables/xalan/xalan-j_2_7_1;
   ```
5. Set the value of the *$syscmd* variable:

   ```
   $syscmd="/usr/bin/java -jar $xalan/xalan.jar -in $input -xsl $xslt -out $output";
   ```

   If you manually modify the path for the log file (`LOGDIR` in `/etc/default/lwact` file), the value of *$input* must be modified accordingly: `$input="/var/log/$hostId.lwact.xml";`
6. Ensure that the *$xslt* variable points to `singleLWACT.xsl` in `/opt/SUNWlwact/xsl/`:
   `$xslt="/opt/SUNWlwact/xsl/singleLWACT.xsl";`
7. Replace avoidance.central with the Fully Qualified Domain Name (FQDN) of your host in the subroutine display_results():

   ```
   print "<meta http-equiv=\"Refresh\" content=60;
   URL=http://avoidance.central/cgi-bin/my_uptime.cgi\">\n";
   ```
8. To view the GUI-based Oracle Lightweight Availability Collection Tool availability report, type the following URL in the address field of your web browser:
   `http://<FQDN>/cgi-bin/my_uptime.cgi`

Troubleshooting

This section explains the troubleshooting steps for the Oracle Lightweight Availability Collection Tool.

Oracle Lightweight Availability Collection Tool Error Messages

This section lists the various errors logged by the Oracle Lightweight Availability Collection Tool, their functional meaning, and any actions that should be taken when these errors are displayed either in `/var/adm/message` or on the screen.

[logtime] Default causecode [XX] at LWACT configuration file is invalid

Indicates that an invalid cause code entry is in the `/etc/default/lwact` file. The user can set up to three levels of default cause codes for the outage events in this file. The cause code level that contains incorrect entry is logged in the error message with the square brackets ([]); that is, XX can take either `[L1CC]` or `[L2CC]`, or `[L3CC]` based on the level of cause code that is invalid.

Action: Enter the valid set of cause codes against the `L1CC`, `L2CC`, `L3CC` fields in the `/etc/default/lwact` file. Use `logtime -M` command to get the list of valid cause codes for all three levels.

Invalid event number

Indicates that a user has tried to modify the cause code for an invalid event number; that is, a non-outage event. Users can modify/assign the cause codes in only the halt and panic outage events.

Action: Use the ltreport `-v` command to display the list of outage events along with their corresponding event numbers.

Invalid Level-X cause code: Invalid cause code entered

Indicates that a user has entered invalid an cause code for the level displayed in the message. X can be either 1, 2 or 3.

Action: For each level 1 cause code, there is a corresponding umbrella of level 2 and level 3 cause codes under it. The only valid cause codes for that level is listed under the umbrella. To obtain the valid list of cause codes, use the `logtime -m` command.

[logtime] event entry X was modified

Indicates a user has successfully modified the event number *X*. In this message, X is the event number.

Action: No action is required. Informational only.

[tictimed]: stopping on SIGTERM or SIGPWR

This message is logged when the Oracle Lightweight Availability Collection Tool terminates (for example, in the case of `pkgrm`).

Action: No action is required. Informational only.

[tictimed] Daemon instance already running

Indicates a user has tried to start the `tictimed` daemon that is already running.

Action: No action is required. Informational only.

[tictimed] Catastrophic file error - zero length

LWACT is removing the zero byte file and starting afresh. Occurs when the availability datagram file turns to 0 bytes in size for an unknown reason.

Action: For pre-LWACT 3.2 installation, remove the zero byte file, `tictimed` will recreate it. For LWACT 3.2 or higher versions, no action is required. LWACT will automatically remove the zero byte file.

[tictimed] datagram file corruption detected

The entire message is as follows:

```
[tictimed] datagram file corruption detected. LWACT is quarantining the
corrupted file and starting afresh. If required user can pick up the uncorrupted
datagram file from the last run explorer output in-rder to avoid considerable
data loss.
```

Whenever the Availability datagram is found to be corrupted, the Oracle Lightweight Availability Collection Tool automatically quarantines it to the same folder where the Availability datagram is present with a filename of the format: lwact_corrupted_<UTC at which the corruption was detected> (for example: lwact_corrupted_1208531225). Quarantining the Availability datagram causes a data loss in the Oracle Lightweight Availability Collection Tool. Old data, collected before the file corruption occurred, will not be taken into account by the tool during the availability calculation.

Action: In order to minimize this data loss, you can manually obtain the uncorrupted copy of the datagram from the previous Explorer image.

[tictimed] Unable to update timestamp on log file

If the Availability datagram is lost or deleted for some reason, `tictimed`, which periodically updates the timestamp on the log file, will not be able to carry out this activity. Hence, it logs the error message. A few possible cases where this error can occur are the following:

- The datagram file is corrupted and `tictimed` has quarantined it.

- The Availability datagram file has been deleted by the user for some reason.

Action: No action is required. `tictimed` will automatically recreate the file afresh if it does not find it.

Attempting to start LWACT. Respawning inittab

Indicates that user has attempted to start LWACT manually using the init script.

Action: No action required. Information only.

LWACT is already running

Indicates that user has attempted to start LWACT which was already running.

Action: No action required. Information only.

LWACT is going down

Indicates that user has attempted stop LWACT manually using the `init` script.

Action: No action required. Information only.

ATTENTION Event generation not in chronological order. It can affect availability metrics

The entire message is as follows:

```
**ATTENTION** Event generation not in chronological order. It can affect
availability metrics. Sudden fall back in system date may have caused this. Check and
correct system date. Otherwise, quarantine current datagram to start monitoring
availability afresh.
```

Occurs when the availability events are recorded out-of-sequence in the availability datagram. out-of-sequence events can occur due to sudden fall back in system date (for example, system shutdown today and boots back to a date from last week). In such cases, LWACT detects the sudden shift in time and records the message indicating the exact time when the system fell back in time. The affected system can report incorrect availability metrics.

Action: You can check and correct the system date or quarantine the current datagram to start monitoring the availability of the system afresh. Please note that old availability metrics will be lost during when the datagram is quarantined.

Failed to list SAVECORE dir contents

Indicates that the `SAVECORE` directory is clean from any core dumps and therefore LWACT was unable to get the contents of this directory.

Action: No action required. Information only.