cmlmdish: Telemetry Data Collector Version 1

CML00077-01

Code Magus Limited (England reg. no. 4024745)
Number 6, 69 Woodstock Road
Oxford, OX2 6EY, United Kingdom
www.codemagus.com

December 15, 2020

Contents

List of Figures

1 Code Magus Dish Guide

1.1 Introduction

Code Magus Dish collects telemetry data by listening for and recording metrics from remote metric producers. Additionally it can act as a forwarding agent by forwarding the telemetry data on to a remote metric consumer either in real time or later from recorded telemetry data.

See figure 1 on page 2 for a pictorial overview of the environment that `cmlmdish` operates within.

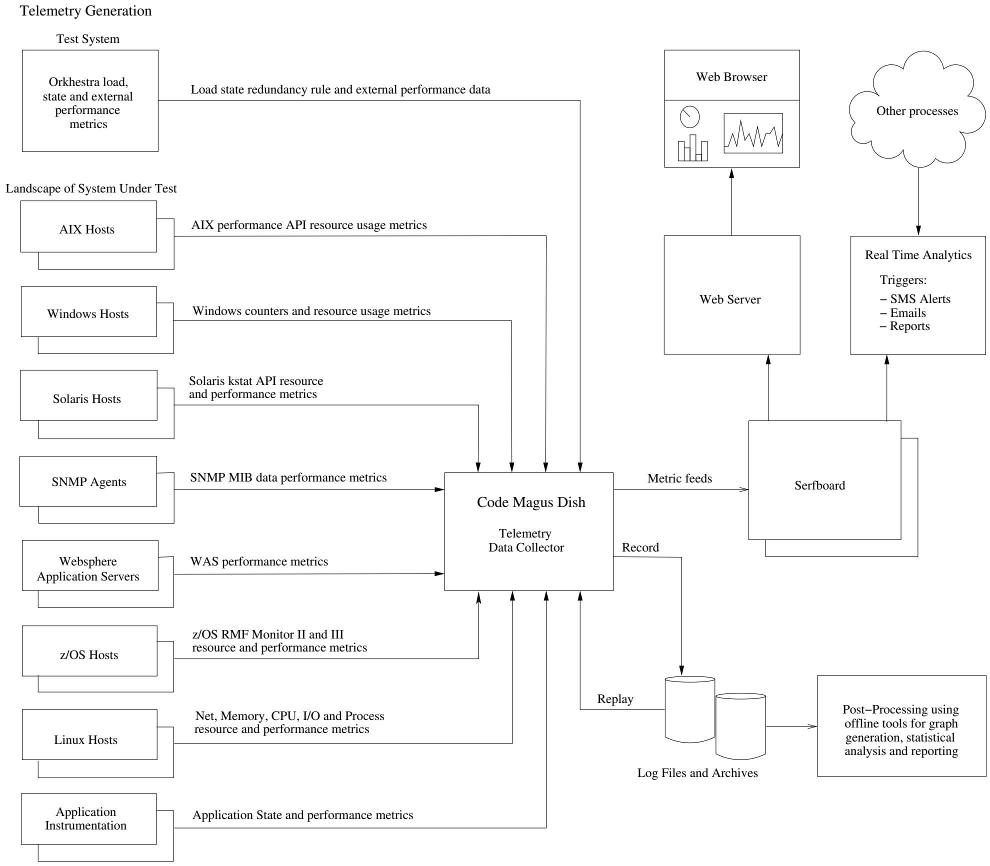
Telemetry Generation
Test System
Orkhestra load, state and external performance metrics
Load state redundancy rule and external performance data
Landscape of System Under Test
AIX Hosts
AIX performance API resource usage metrics
Windows Hosts
Windows counters and resource usage metrics
Solaris Hosts
Solaris kstat API resource and performance metrics
SNMP Agents
SNMP MIB data performance metrics
Websphere Application Servers
WAS performance metrics
z/OS Hosts
z/OS RMF Monitor II and III resource and performance metrics
Linux Hosts
Net, Memory, CPU, I/O and Process resource and performance metrics
Application Instrumentation
Application State and performance metrics
Code Magus Dish
Telemetry Data Collector
Metric feeds
Record
Replay
Web Browser
Other processes
Web Server
Real Time Analytics
Triggers:
– SMS Alerts
– Emails
– Reports
Serfboard
Log Files and Archives
Post–Processing using offline tools for graph generation, statistical analysis and reporting

Figure 1: Overview of the **Code Magus Dish** environment.

1.2 Environment

Code Magus Dish exists within an environment of metric producers and consumers. It collects all telemetry data it receives and records it. In contrast a metric producer only generates metrics from one source and a metric consumer only processes those metrics it is configured to recognise.

1.2.1 Metric Producers

Metric producers generate telemetry data from a variety of sources such as:

- An application enabled to produce metrics.
  An example would be a real time network proxy or concentrator. As these applications handle many transactions a second, operators use the telemetry data produced to monitor the throughput and overall performance of such systems.
- A non-functional or stress test environment.
  Code Magus Orkhestra [2] is an example of one such system. The metrics produced enable system testers to analyse the throughput and performance of the system under test.
- Code Magus Metric Probes.
  These utility programs actively query machine, system level software and supporting applications for performance metrics.
  - cmlxsnmp: SNMP Metric Probe [3].
    Uses the Simple Network Management Protocol (SNMP) library to query any network device that supports SNMP.
  - cmlxaixp: AIX Performance Metric Probe [4].
    Queries AIX machine performance metrics.
  - cmlxwinp: Windows Performance Metric Probe [5].
    Queries Windows machine performance metrics.
  - cmlxwasp: Websphere Application Server Performance Metric Probe [6].
    Queries Websphere Application Server performance metrics.
  - cmlxsolp: Solaris Performance Metric Probe [7].
    Queries Solaris machine performance metrics.

Using the above together, for example a system under test and probes on the actual machines that affect it, a detailed analysis of the performance of the system can be achieved.

1.2.2 Metric Consumers

Metric consumers process telemetry data in order to inform about the performance of the systems, machines and applications that generated the telemetry data. Some examples are:

- An application that renders the telemetry data as a dashboard.
  An example of this is `Serfboard` [1] that renders the information graphically on demand via a dashboard on a web browser.
- Offline Utilities.
  These utilities are either Code Magus products or other non-propriety systems such as R[8] and gnu plot[9]. They usually operate on the recorded telemetry data to produce analysis, trends and graphs that describe the performance of those systems that produced the telemetry data.

1.3 Summary

Code Magus Dish is primarily a telemetry data collector.

It can act as a metric consumer under normal operation mode by collecting and recording the telemetry data it receives. It can also act as a metric producer when forwarding telemetry data either in real time or in replay mode. In replay mode it reads a previously recorded telemetry data log and forwards the telemetry data to a metric consumer in the same manner (chronological order and timing) as they were recorded. The time stamp in each metric is updated to the current time.

2 Code Magus Dish Reference

2.1 Usage

Code Magus Dish (`cmlmdish`) is started from the command line preferably in the background as a daemon process and should not be associated with the terminal initiating it.

Under normal operation mode `cmlmdish` will receive telemetry data from metric producers, record them to a log file and optionally forward them on to a metric consumer. `cmlmdish` will accept both TCP/IP and UDP connections. Any duplicate messages received via UDP are automatically discarded and therefore not forwarded on or recorded.

If `cmlmdish` is started with the parameter '`--help`' it will only display the command line parameters it accepts and a short description of each. The list of parameters along with a complete description of each can be found in the following section.

```
Code Magus Limited Metric Recorder V1.1: build 2012-12-18-11.32.47
[./cmlmdish] $Id: cmlmdish.c,v 1.2 2012/12/21 11:34:00 janvlok Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
 [Contact: stephen@codemagus.com].
Usage: cmlmdish [OPTION...]
  -p, --port=<port>                       Port number for connections
  -h, --forward-host={localhost|<host>}   Host name for forwarding metrics
                                          to
  -P, --forward-port=<port>               Port number for forwarding
                                          metrics to
      --forward-host2={localhost|<host>}  2nd Host name for forwarding
                                          metrics to
      --forward-port2=<port>              2nd Port number for forwarding
                                          metrics to
  -u, --udp                               Use UDP connection for forwarding
  -l, --log=<file name>                   Recording log file name
  -r, --replay-log                        Read the log file and forward to
                                          host
  -v, --verbose                           Verbose processing
  -t, --trace                             Trace all messages to stdout

Help options:
  -?, --help                              Show this help message
      --usage                             Display brief usage message
```

2.2 Parameters

- '`-p|--port`' Specifies the port number to listen on for incoming connections from metric producers.

- '`-h|--forward-host`' Specifies the host IP address of a metric consumer server process to forward the telemetry data to.
- '`-p|--forward-port`' Specifies the port number of the metric consumer server process to forward the telemetry data to.
- '`--forward-host2`' Specifies the host IP address of a second metric consumer server process to forward the telemetry data to.
- '`--forward-port2`' Specifies the port number of the second metric consumer server process to forward the telemetry data to.
- '`-u|--udp`' Specifies that telemetry data are to be forwarded to the consumer(s) using UDP. If this is not specified then the telemetry data are forwarded using TCP.
- '`-l|--log`' Specifies the name of the file that the telemetry data are recorded to. This file is written using the Code Magus `rdwlog` library.
- '`-r|--replay-log`' Specifies to read the log file of recorded telemetry data and forward them to the metric consumer server process.
- '`-v|--verbose`' Directs `cmlmdish` to output diagnostic information during execution.
- '`-t|--trace`' Directs `cmlmdish` to output the maximum amount of diagnostic information during execution.

2.3 Examples

2.3.1 `cmlmdish` as a Collector, Recorder and Forwarder

The following invocation will accept telemetry data on the port `6400`, record the telemetry data to the file `metric_log.txt` using the `rdwlog` library and at the same time forward them over UDP to the server `192.168.17.17` using port `6500`.

```
cmlmdish  --verbose \
               --log=metric_log \
               --port=6400 \
               --forward-host=192.168.17.17 \
               --forward-port=6500 \
               --udp
```

```
Code Magus Limited Metric Recorder V1.1: build 2011-01-05-12.10.43
[./cmlmdish] $Id: cmlmdish.c,v 1.2 2012/12/21 11:34:00 janvlok Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
 [Contact: stephen@codemagus.com].
2011-01-05-12.13.24.653181: Feeder (0.0.0.0:6400) - Waiting connection
2011-01-05-12.13.24.654057: Opened log: text(metric_log.txt,mode=w)
```

```
2011-01-05-12.13.24.654105: Forwarding: Connecting to 192.168.17.17:6500 UDP
2011-01-05-12.13.24.654193: Forwarding host Connected
2011-01-05-12.13.27.841391: Shutdown
```

2.3.2 `cmlmdish` in Replay Mode as a Metric Producer

The following example shows an invocation of the `cmlmdish` in replay mode. It reads the file `test.metric.rdw` and forwards the telemetry data on to the server `localhost` on port `12345`.

```
cmlmdish --replay-log --log=test.metric.rdw \
      --forward-host=localhost --forward-port=12345
Code Magus Limited Metric Recorder V1.1: build 2012-12-18-11.32.47
[./cmlmdish] $Id: cmlmdish.c,v 1.2 2012/12/21 11:34:00 janvlok Exp $
Copyright (c) 2010 by Code Magus Limited. All rights reserved.
 [Contact: stephen@codemagus.com].
2012-12-19-16.52.21.674406: Forwarding: Connecting to localhost:12345 TCP
2012-12-19-16.52.21.677971: Forwarding host localhost:12345 Connected
Replayed 27 records
```

References

[1] Serfboard User Guide Version 1. CML Document CML00027-01, Code Magus Limited, July 2008. PDF.

[2] orkhestra: Configuration and User Reference Version 1. CML Document CML00041-01, Code Magus Limited, June 2011. PDF.

[3] cmlxsnmp: SNMP Metric Probe. CML Document CML00044-01, Code Magus Limited, June 2009. PDF.

[4] cmlxaixp: AIX Performance Metric Probe. CML Document CML00045-01, Code Magus Limited, June 2009. PDF.

[5] cmlxwinp: Windows Performance Metric Probe. CML Document CML00048-01, Code Magus Limited, June 2009. PDF.

[6] cmlxwasp: Websphere Application Server Performance Metric Probe. CML Document CML00049-01, Code Magus Limited, June 2009. PDF.

[7] cmlxsolp: Solaris Performance Metric Probe. CML Document CML00065-01, Code Magus Limited, June 2009. PDF.

[8] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2015.

[9] Thomas Williams, Colin Kelley, and many others. Gnuplot: an interactive plotting program. `http://gnuplot.sourceforge.net/`, December 2012.