BEA TSAM

Reference Guide

Version 1.1
Document Released: September 28, 2007

Contents

CHAPTER 1

BEA TSAM Reference Guide

The BEA TSAM Reference Guide describes, system processes and commands delivered with the BEA SALT software.

Table 1 BEA TSAM System Processes and Commands

Name	Description
tpgetcallinfo	Routine for getting monitoring attributes of a message in a call path
LMS (Local Monitor Server)	The BEA TSAM Agent Local Monitor Server

tpgetcallinfo

Name

tpgetcallinfo()— Routine for getting monitoring attributes of a message in a call path monitoring

Synopsis

```
int tpgetcallinfo(const char *msg, FBFR32 **obuf, long flags)
```

Description

tpgetcallinfo() is BEA TSAM API this use for call path monitoring only.tpgetcallinfo() supports the following parameters:

`msg`
The typed buffer use for measurement.

`obuf`
The FML32 buffer used to contain the fields.

`flags`
Reserved for future use.

For monitored calls, `tpgetcallinfo()`uses the following fields: following fields:

`tpgetcallinfo()`
Retrieves the message monitoring attributes when call path monitoring is enabled.

`tpgetcallinfo()`
Can be used in different scenarios to accomplish different functions. Typical usage is as follows:

- Application server calls `tpgetcallinfo()` to check the requested message monitoring attributes. It can provide the following information:

  - Correlation ID of the request. It is generated by the caller

  - Begins time stamping when the monitoring initiator starts the call

  - Last stop time stamp on the call path tree of the monitored request. Usually, it is used to measure the process requested message waiting time for a service

  - Workstation client address if the request is from a Tuxedo workstation client

- The monitoring initiator calls `tpgetcallinfo()` to get the end to the monitored call end time.

  Note: `tpgetcallinfo()` can be called at any time for a reply buffer after a reply is received. This is especially useful with `tpacall`/`tpgetrply`.

Table 2 lists the FML monitor initiator field names.

Table 2 Monitor Initiator Field Names

Field Name	Type	Description	Service	Monitoring Initiator
TA_MONCORRID	string	The monitored call correlation ID. It is a critical call path monitoring metric.	Y	Y
TA_MONLASTTIMESEC	long	Timestamp for the last stop on the call path tree in seconds.	Y	Y

Table 2 Monitor Initiator Field Names

TA_MONLASTTIMEUSEC	long	Ttimestamp of last stop on the call path tree in microseconds.	Y	Y
TA_MONSTARTTTIMESEC	long	Timestamp when the monitoring initiator starts the call in seconds.	Y	Y
TA_MONSTARTTIMEUSEC	long	Timestamp of the monitoring initiator starts the call in microseconds.	Y	Y
TA_MONCLTADDR	string	The workstation client address	Y	N
TA_MONTOTALTIME	long	The end-to-end time used for a monitored call in milliseconds.	N	Y

Return Values

Upon successfully getting a FML32 buffer containing the monitoring attributes, returns 0.

Upon failure, `tpgetcallinfo()` returns -1 and sets tperrno to indicate the error condition.

Errors

Upon failure, `tpgetcallinfo()` sets tperrno to one of the following values,

`[TPEINVAL]`
: Invalid arguments were given (for example, msg is NULL or obuf is not a valid FML32 buffer)

`[TPESYSTEM]`
: The message input does not contain monitoring attributes. Usually this is because the call path monitoring is not turned on for the message.

`[TPEOS]`
: An operating system error has occurred.

Example(s)

Listing 1 Service-Side tpgetcallinfo Example

```
#include <stdio.h>
#include <atmi.h>
#include <userlog.h>
#include <fml32.h>
```

```
#include <tpadm.h>
#if defined(__STDC__) || defined(__cplusplus)
tpsvrinit(int argc, char *argv[])
#else
tpsvrinit(argc, argv)
int argc;
char **argv;
#endif
{

       /* tpsvrinit logic */
#ifdef __cplusplus
extern "C"
#endif
void
#if defined(__STDC__) || defined(__cplusplus)
APPSVC(TPSVCINFO *rqst)
#else
TOUPPER(rqst)
TPSVCINFO *rqst;
#endif
{
       FBFR32 *metainfo;
       int len = 0;
       /* Allocate the metainfo space */
       metainfo = tpalloc("FML32", NULL, 1024);
       if (metainfo == NULL ) {
              userlog("Memory allocation failed");
              tpreturn(TPFAIE, 0, 0, 0);
       }
       /* Get the monitoring attributes*/
       if ( tpgetcallinfo(rqst->data, &metainfo, 0) == 0 )
       {
              char *corrid;
              long laststopsec, starttimesec;
              if ((corrid = Ffind32(metainfo, TA_MONCORRID, 0, &len) ) {
                     userlog("Correlation ID = %s", corrid);
              }
```

```
                len = sizeof(starttimesec);
                if (( Fget32(metainfo, TA_MONSTARTTIMESEC, &starttimesec,
&len) == 0) {
                        userlog("Message beginning time = %ld", starttimesec);
                }
}
                len = sizeof(lasttimesec);
        if (( Fget32(metainfo, TA_MONLASTTIMESEC, &lasttimesec, &len)  == 0)
{
                userlog("Message entering my queue time = %ld", lasttimesec);
        }
        }
        tpfree(metainfo);
        /* rest of service processment */
        °¦ °¦
}
```

LMS (Local Monitor Server)

Name

LMS—The BEA TSAM Agent Local Monitor Server

Synopsis

```
LMS SRVGRP="identifier" SRVID="number" [other_parms]
CLOPT= "-A -- -l tsam-manager-dataserver-url [-t heartbeat-interval]
[-x internal-queue-size-limit] [-e log-warning-interval]"
```

Description

LMS is a BEA TSAM Agent Tuxedo server. It provides the following functions:

- Acts as the local Tuxedo domain data collection proxy

  The performance metrics collected by the BEA TSAM framework are passed to the plug-in. BEA TSAM default plug-in sends the data to the LMS using the Tuxedo service infrastructure.

- The plug-in metrics are stored in the LMS before being sent to the BEA TSAM manager data server. The LMS and BEA TSAM manager use the HTTP+XML protocol.
- Other management information exchanges between the LMS and BEA TSAM manager.

The LMS must be configured in the UBBCONFIG file and set with the proper options. Only one LMS can be deployed per Tuxedo machine.

Options

`-l`

Mandatory parameter. It is followed by the BEA TSAM manager data server. The host address and port number are set based on your BEA TSAM manager installation.

Note: `tsam/dataserver` is the data server address. It strongly recommended that `tsam/dataserver` is not changed.

`-x`

Optional parameter. It limits the LMS max internal queue size. Its range is `[1001-999999]`. The messages sent from the BEA TSAM default plug-in are stored in the queue first. If the system load is heavy, the queue may consume a lot of memory. The `-x` option allows you to adjust the amount of memory used by LMS. The default is `100000`.

If the internal queue max size is reached, the oldest message is dropped and the latest one is added.

`-t`

Optional parameter. It specifies the time interval in seconds that LMS should connect to the BEA TSAM manager if there is no performance activity. Its range is `[1-60]`. The default value is 30 seconds.

`-e`

Optional parameter. It specifies the time interval LMS sends a warning message to ULOG if performance metrics data is dropped due to queue size limit. Its range is `[1-65535]`. The default value is equal the `-t` value. The warning message reports how many messages have been lost during the past interval.

Environment Variable(s)

`TM_MON_REPORT_LOG`

Controls the reporting time interval for the following message in the event the plug-in fails to call the LMS server:

`LIBTUX_CAT:6774: WARN: LMS` server is unavailable currently

Note: You must manually set `TM_MON_REPORT_LOG`.

TM_MON_REPORT_LOG has a range (in seconds) of [0,65535]. The the time 60s. TM_MON_REPORT_LOG=0, indicates that the warning message is disabled.

Example(s)

Listing 2 LMS in UBBCONFIG

```
...
*SERVERS
LMS SRVGRP=LMSGRP SRVID=1 MINDISPATCHTHREADS=1 MAXDISPATCHTHREADS=5
CLOPT="-A -- -l tsamweb.abc.com:8080/tsam/dataserver -x 200000 -t 60 -e 120"
...
```

LMS Notes

LMSSVC and Security

The communication channel between a plug-in and the LMS is based on the Tuxedo service infrastructure. LMSSVC is the service advertised by LMS to receive requests. If ACL and MANDATORY ACL are used in UBBCONFIG, LMSSVC should be configured to allow all monitored processes access.

How Many LMS Deployed in One Domain

It is recommend that you deploy one LMS for each machine so that IPC queue communication through local processes to LMS is limited.

Note: A single LMS can be used by multiple machines, but it will consume network bandwidth and extend the BRIDGE computing cycle. The overall Tuxedo application performance may be significantly impacted.

CHAPTER 2

BEA TSAM Error Messages

BEA TSAM Error Messages

Table 2-1 lists the BEA TSAM Error Messages.

Table 2-1 BEA TSAM Error Messages

Message Code	Description
TSAM-0001	Cannot connect to database
TSAM-0002	Error occurred when execute SQL statement: {0}
TSAM-0003	SQL exception
TSAM-0101	Performance metrics message parsing error
TSAM-0102	No PERFDATA element found
TSAM-0103	No LOCATION element found
TSAM-0104	Acesse database error: {0}
TSAM-0105	DataServer failed to initialize: {0}
TSAM-0106	Configuration message parsing error: {0}
TSAM-0107	No DOMAIN element found
TSAM-0108	No DOMAINID element found

Table 2-1 BEA TSAM Error Messages

Message Code	Description
TSAM-0109	No MASTER element found
TSAM-0110	No LMID element found in MACHINCE section
TSAM-0111	No RELEASE element found in MACHINE section
TSAM-0112	No PMID element found in MACHINE section
TSAM-0113	No GRPNAME element found in GROUP section
TSAM-0114	No GRPNO element found in GROUP section
TSAM-0115	No SRVNAME element found in SERVER section
TSAM-0116	No SRVID element found in SERVER section
TSAM-0117	No SVCNAME element found in SERVICE section
TSAM-0118	Original message{0} dropped
TSAM-0119	Heartbeat message parsing error: {0}
TSAM-0120	RequestServer failed to initialize: {0}
TSAM-0121	Monitoring policy reply message parsing error: {0}
TSAM-0122	No SEQUENCE found for monitoring policy reply message
TSAM-0123	Invalid Tuxedo message: {0}
TSAM-0124	No HEADER element found
TSAM-0125	No TYPE element found
TSAM-0126	No supported message type: {0}
TSAM-0127	TSAM Data Server started
TSAM-0128	TSAM Request Server started
TSAM-0129	JDBC Driver: {0}
TSAM-0130	JDBC URL: {0}
TSAM-0131	JDBC Username: {0}

Table 2-1 BEA TSAM Error Messages

Message Code	Description
TSAM-0132	Invalid topnsvc {0} web.xml value
TSAM-0133	Invalid minapptime {0} web.xml value
TSAM-0134	Invalid recsperpage {0} web.xml value
TSAM-0135	Invalid timetrans {0} web.xml value
TSAM-0136	Invalid windowsvc {0} web.xml value
TSAM-0137	Invalid windowsrv {0} web.xml value
TSAM-0138	Invalid minapppattern {0} web.xml value
TSAM-0139	Invalid windowevt {0} web.xml value
TSAM-0140	Invalid windowhs {0} web.xml value
TSAM-0141	Invalid maxappactive {0} web.xml value
TSAM-0142	Invalid maxappdone {0} web.xml value
TSAM-0143	Invalid maxappsize {0} web.xml value
TSAM-0144	Invalid timeoutwithtuxedo {0} web.xml value
TSAM-0145	Failed to update Tuxedo objects cache from database
TSAM-0146	Failed to update Alert objects cache from database
TSAM-0160	ERROR: Unknown plug-in event type {0}
TSAM-0161	ERROR: XML data parsing error
TSAM-0162	ERROR: Missing required XML data field {0}
TSAM-0163	Plug-in generated event
TSAM-0164	INFO: {0} events deleted.
TSAM-0501	Time Zone is set to {0}
TSAM-0502	Effective Time Zone is {0}
TSAM-0503	Unknown Transport {0}

Table 2-1 BEA TSAM Error Messages

Message Code	Description
TSAM-0504	INFO: Nothing found
TSAM-0505	BIRT not initialized
TSAM-0506	BIRT engine not available
TSAM-0507	BIRT data source problem
TSAM-0508	BIRT encountered problem
TSAM-0509	Cleanup BIRT temporary image files periodically
TSAM-0510	INFO: No monitoring data
TSAM-0511	INFO: Monitoring policy {0} is applied to Tuxedo components {1}
TSAM-0512	INFO: Monitoring policy {0} is created for Tuxedo components {1}
TSAM-0513	INFO: Monitoring set {0} is created
TSAM-0514	INFO: Monitoring set {0} is edited
TSAM-0515	INFO: Please enter Correlation ID
TSAM-0516	ERROR: Call Path data for correlation id {0} in database is corrupted
TSAM-0517	INFO: Please enter Global Transaction ID (GTRID)