Accounting in EntireX Broker

This chapter describes the accounting records for Broker that can be used for several purposes, including:

- **application chargeback**
  for apportioning EntireX resource consumption on the conversation and/or the application level;
- **performance measurement**
  for analyzing application throughput (bytes, messages, etc.) to determine overall performance;
- **trend analysis**
  for using data to determine periods of heavy and/or light resource and/or application usage.

This chapter covers the following topics:

- EntireX Accounting Data Fields
- Using Accounting under UNIX and Windows
- Using Accounting under z/OS
- Example Uses of Accounting Data

EntireX Accounting Data Fields

In the EntireX Accounting record, there are various types of data available for consumption by applications that process the accounting data:

Field Name	Accounting Version	Type of Field	Description
Record Write Time	1	z/OS: I4I4 timestamp Other platforms: A14 timestamp	z/OS: SMF timestamp in format I4I4 (time in hundredths of seconds followed by date in format X'0CYYDDDF' (packed decimal number)). Other platforms: The time this record was written to the accounting file in "YYYYMMDDHHMMSS" format.
EntireX Broker ID	1	A32	Broker ID from attribute file.
EntireX Version	1	A8	Version information, *v.r.s.p* where *v* =version *r* =release *s* =service pack *p* =patch level for example 9.7.0.00.

Field Name	Accounting Version	Type of Field	Description
Platform of Operation	1	A32 (A8 under z/OS)	Platform where EntireX is running.
EntireX Start Time	1	z/OS: I4I4 timestamp Other platforms: A14 timestamp	z/OS: The time EntireX was initialized in format I4I4 (time in hundredths of seconds followed by date in format X'0CYYDDDF' (packed decimal number)). Other platforms: The time EntireX was initialized in "YYYYMMDDHHMMSS" format.
Accounting Record Type	1	A1	It is always C for conversation. Future Types will have a different value in this field.
Client User ID	1	A32	USER-ID ACI field from the client in the conversation.
Client Token	1	A32	`TOKEN` field from the ACI from the client.
Client Physical ID	1	A56	The physical user ID of the client, set by EntireX.
Client Communication Type	1	I1	Communication used by client: 1 = Net-Work 2 = TCP/IP 3 = APPC 4 = WebSphere MQ 5 = SSL
Client Requests Made	1	I4	Number of Requests made by client.
Client Sent Bytes	1	I4	Number of bytes sent by client.
Client Received Bytes	1	I4	Number of bytes received by client.
Client Sent Messages	1	I4	Number of messages sent by client.
Client Received Messages	1	I4	Number of messages received by client.
Client Sent UOWs	1	I4	Number of UOWs sent by client.
Client UOWs Received	1	I4	Number of UOWs received by client.
Client Completion Code	1	I4	Completion code client received when conversation ended.
Server User ID	1	A32	`USER-ID` ACI field from the server in the conversation.
Server Token	1	A32	`TOKEN` field from the ACI from the server.

Field Name	Accounting Version	Type of Field	Description
Server Physical ID	1	A56	The physical user ID of the server, set by EntireX.
Server Communication Type	1	I1	Communication used by Server: 1 = Entire Net-Work 2 = TCP/IP 3 = APPC 4 = WebSphere MQ 5 = SSL
Server Requests Made	1	I4	Number of requests made by server.
Server Sent Bytes	1	I4	Number of bytes sent by server.
Server Received Bytes	1	I4	Number of bytes received by server.
Server Sent Messages	1	I4	Number of messages sent by server.
Server Received Messages	1	I4	Number of messages received by server.
Server Sent UOWs	1	I4	Number of UOWs sent by server.
Server Received UOWs	1	I4	Number of UOWs received by server.
Server Completion Code	1	I4	Completion code server received when conversation ended.
Conversation ID	1	A16	`CONV-ID` from ACI.
Server Class	1	A32	`SERVER-CLASS` from ACI.
Server Name	1	A32	`SERVER-NAME` from ACI.
Service Name	1	A32	`SERVICE` from ACI.
CID=NONE Indicator	1	A1	Will be N if `CONV-ID=NONE` is indicated in application.
Restarted Indicator	1	A1	Will be R if a conversation was restarted after a Broker shutdown.
Conversation Start Time	1	z/OS: I4I4 timestamp Other platforms: A14 timestamp	z/OS: The time the conversation began in format I4I4 (time in hundredths of seconds followed by date in format X’0CYYDDDF’ (packed decimal number)). Other platforms: The time the conversation began in "YYYYMMDDHHMMSS" format.

Field Name	Accounting Version	Type of Field	Description
Conversation End Time	1	z/OS: I4I4 timestamp Other platforms: A14 timestamp	z/OS: The time the conversation was cleaned up in format I4I4 (time in hundredths of seconds followed by date in format X'0CYYDDDF' (packed decimal number)). Other platforms: The time the conversation was cleaned up in "YYYYMMDDHHMMSS" format.
Conversation CPU Time	1	I4	Number of microseconds of CPU time used by the conversation
Client Security Identity	2	A32	Actual identity of client derived from authenticated user ID.
Client Application Node	2	A32	Node name of machine where client application executes.
Client Application Type	2	A8	Stub type used by client application.
Client Application Name	2	A64	Name of the executable that called the broker. Corresponds to the Broker Information Service field `APPLICATION-NAME`.
Client Credentials Type	2	I1	Mechanism by which authentication is performed for client.
Server Security Identity	2	A32	Actual identity of server derived from authenticated user ID.
Server Application Node	2	A32	Node name of machine where server application executes.
Server Application Type	2	A8	Stub type used by server application.
Server Application Name	2	A64	Name of the executable that called the broker. Corresponds to the Broker Information Service field `APPLICATION-NAME`.
Server Credentials Type	2	I1	Mechanism by which authentication is performed for server.
Client RPC Library	3	A128	RPC library referenced by client when sending the only/first request message of the conversation.
Client RPC Program	3	A128	RPC Program referenced by client when sending the only/first request message of the conversation.
Server RPC Library	3	A128	RPC library referenced by server when sending the only/first response message of the conversation.

Field Name	Accounting Version	Type of Field	Description
Server RPC Program	3	A128	RPC Program referenced by server when sending the only/first response message of the conversation.
Client IPv4 Address	4	A16	IPv4 address of the client.
Server IPv4 Address	4	A16	IPv4 address of the server.
Client Application Version	4	A16	Application version of the client.
Server Application Version	4	A16	Application version of the server.
Client IPv6 Address	5	A46	IPv6 address of the client.
Server IPv6 Address	5	A46	IPv6 address of the server.

Note:
Accounting fields of any version greater than 1 are created only if the attribute `ACCOUNTING-VERSION` value is greater than or equal to the corresponding version. For example: accounting fields of version 2 are visible only if `ACCOUNTING-VERSION=2` or higher is specified.

Using Accounting under UNIX and Windows

- Broker Attribute File Settings
- Retrieving Accounting Data

Broker Attribute File Settings

ACCOUNTING = NO | YES | (YES, SEPARATOR=Separator Characters) (Default is NO)

Set this parameter to "NO" (i.e., do not create accounting data) or "YES" to create accounting data. Up to seven separator characters can specified using the SEPARATOR suboption, for example ACCOUNTING = (YES, SEPARATOR=;). If no separator character is specified, the comma character will be used.

Retrieving Accounting Data

The accounting file will be located in the Broker's installed directory. The file's name is based on the ETB_LOG environment variable and the current date and time (for uniqueness). Example: If ETB_LOG is set to BROKER1.LOG, the accounting data file will be named BROKER1_*YYYYMMDDHHMMSS*.csv. If ETB_LOG is not set, the Broker's ID will be used, with an extension of CSV (e.g. ETB048_*YYYYMMDDHHMMSS*.csv). See *Environment Variables in EntireX*.

Using Accounting under z/OS

The `ACCOUNTING` attribute indicates if accounting records will be generated. Accounting records are written upon successful completion of a conversation. A conversation ending in an application error (such as a timeout) is considered to be a successful conversation.

- Attribute File
- Retrieving Accounting Records
- Accounting Record Layouts
- Notes

Attribute File

ACCOUNTING={NO|128-255}

Set this parameter to "NO" (i.e., do not create accounting records) or to a number between 128 and 255, which specifies the SMF record type to use when writing the accounting records. In order to avoid conflicts with other applications that also produce SMF records, check with your z/OS systems programmer for an appropriate number. In addition, check with your z/OS systems programmer to ensure that the selected SMF record number is set up to be written.

Default value: NO

Retrieving Accounting Records

The standard IBM IFASMFDP utility program may be used to selectively offload Broker SMF records. Analysis and report routines - either user-written or those available from IBM or various software vendors - may subsequently be used to process the offloaded records.

```
//* Copies selected records from the "live" SMF data sets
//*
//* Replace nnn (OUTDD parameter) with a valid SMF record type
//*
//* Note: the "DISPLAY SMF" operator command will show the names of the
//* SMF data sets
//*
//IFASMFDP EXEC PGM=IFASMFDP
//SYSPRINT DD SYSOUT=*
//MAN1 DD DISP=SHR,DSN=SYS1.MAN1
//MAN2 DD DISP=SHR,DSN=SYS1.MAN2
//MAN3 DD DISP=SHR,DSN=SYS1.MAN3
//OUTPUT DD DISP=(MOD,CATLG),
// UNIT=SYSDA,SPACE=(TRK,(15,15),RLSE),
// DCB=(RECFM=VBS,LRECL=32760,BLKSIZE=0),
// DSN=EXX.SMF.RECORDS
//SYSIN DD *
 DATE(2002001,2099366)
 START(0000)
 END(2359)
 INDD(MAN1,OPTIONS(DUMP))
 INDD(MAN2,OPTIONS(DUMP))
 INDD(MAN3,OPTIONS(DUMP))
 OUTDD(OUTPUT,TYPE(nnn))
//*
```

Note:
The IBM publication *MVS System Management Facilities (SMF)* provides complete information on SMF.

Accounting Record Layouts

EntireX provides three mappings for its accounting records in the following members, all located in the EXX970.SRCE data set:

- EXXCACT - A C language include file that maps the accounting record;
- EXXACTR - An Assembler language MACRO that will generate a DSECT of the accounting record;
- EXXSACT - An SAS DATA step that will read in a file with the appropriate field names.

Notes

- Since there is no server for Broker Command and Information Services, no server data is generated in the SMF records for Command and Information Services conversations.
- The unit for CPUTIME is expressed in microseconds.

Example Uses of Accounting Data

- Chargeback
- Trend Analysis
- Tuning for Application Performance

Chargeback

Customers can use the EntireX accounting data to perform chargeback calculations for resource utilization in a data center. Suppose EntireX Broker is being used to dispatch messages for three business departments: Accounts Receivable, Accounts Payable, and Inventory. At the end of each month, the customer needs to determine how much of the operation and maintenance cost of EntireX Broker should be assigned to these departments. For a typical month, assume the following is true:

Department	Amount of Data	Percentage	Messages Sent	Percentage	Average Percentage
Accts Payable	50 MB	25	4000	20	22.5
Accts Receivable	40 MB	20	6000	30	25
Inventory	110 MB	55	10000	50	52.5

The use of Broker resources here is based upon both the amount of traffic sent to the Broker (bytes) as well as how often the Broker is called (messages). The average of the two percentages is used to internally bill the departments, so 52.5% of the cost of running EntireX Broker would be paid by the Inventory Department, 25% by the Accounts Receivable Department, and 22.5% by the Accounts Payable Department.

Trend Analysis

The Accounting Data can also be used for trend analysis. Suppose a customer has several point-of-sale systems in several stores throughout the United States that are tied into the corporate inventory database with EntireX. The stubs would be running at the stores, and the sales data would be transmitted to the Broker, which would hand it off to the appropriate departments in inventory. If these departments wish to ascertain when the stores are busiest, they can use the accounting data to monitor store transactions. Assume all of the stores are open every day from 9 AM to 10 PM.

Local Time	Average: Weekday Transactions per Store	Maximum Weekday Transactions in any Store	Average Weekend Transactions per Store	Maximum Weekend Transactions in any Store
9 AM	7.3	27	28.2	83
10 AM	11.2	31	29.3	102
11 AM	14.6	48	37.9	113
12 noon	56.2	106	34.8	98
1 PM	25.6	65	34.2	95
2 PM	17.2	52	38.5	102
3 PM	12.1	23	42.7	99
4 PM	18.3	34	43.2	88
5 PM	26.2	47	45.2	93
6 PM	38.2	87	40.6	105
7 PM	29.6	83	39.2	110
8 PM	18.6	78	28.6	85
9 PM	11.2	55	17.5	62

The owner of the stores can examine the data and make decisions based upon the data here. For example, on weekdays, he or she can see that there is little business until lunchtime, when the number of transactions increase. It then decreases during lunch hour; then there is another increase from 5 PM to 8 PM, after people leave work. Based on this data, the owner might investigate changing the store hours on weekdays to 10 AM to 9 PM. On the weekend the trends are different, and the store hours could be adjusted as well, although there is a more regular customer flow each hour on the weekends.

Tuning for Application Performance

Assume that a customer has two applications that perform basic request/response messaging for similar sized messages. The applications consist of many Windows PC clients and Natural RPC Servers on UNIX. An analysis of the accounting data shows the following:

Application Type	Class	Server	Service	Average Server Messages Received per Conversation	Average Client Messages Received per Conversation
Application 1:	CLASS1	SERVER1	SERVICE1	10.30	10.29
Application 2:	CLASS2	SERVER2	SERVICE2	10.30	8.98

A further analysis of the accounting data reveals that there are a lot of non-zero response codes in the records pertaining to Application 2, and that a lot of these non-zero responses indicate timeouts. With that information, the customer can address the problem by modifying the server code, or by adjusting the timeout parameters for SERVER2 so that it can have more time to get a response from the Service.