IBM® Security Guardium®
Database Activity Monitor (DAM)
&
Db2® for i
Advanced Filtering Techniques
Version 1.2

Scott Forstie
forstie@us.ibm.com

The master version of this document can be found here:
https://ibm.biz/GuardiumDAMonIBMi

Note: These features are available on IBM i 7.2 and higher.

Table of Contents

Preface:

This document describes the server-side filtering improvements to Guardium DAM with Db2 for i.

Technical Contacts:

When this document doesn't address all your questions, problems or customer requests for enhancement, contact the following:

Author:
Scott Forstie
Title: Db2 for i Business Architect
Email: forstie@us.ibm.com

Version History:

Version 1.0 – Original document
Version 1.1 – Edits
Version 1.2 – More examples added

Resources

Refer to and use the following education resources.

- **Guardium Data Monitoring - Db2 for i fact page**
  - https://ibm.biz/GuardiumDAMonIBMi

- **Guardium Data Monitoring - Db2 for i developerWorks article**
  - http://www.ibm.com/developerworks/ibmi/library/i-infosphere_guardium_db2

- **Guardium Activity Monitor & Db2 for i Serviceability Guide**
  - https://ibm.biz/GuardiumOniServiceabilityGuide

IBM i side Filtering enhancements (Advanced filtering)

DAM on Db2 for i filtering has been enhanced in several ways to provide improved, granular control over the auditing detail captured by the IBM i Audit Server and stored on the Guardium collector.

Important: Use of advanced filtering should be used *only* in the case where the existing filtering capability is not sufficient to enable the Guardium collector to handle the load. When using this feature, all configuration and control of the filtering policy occurs directly on the IBM i. There is no Guardium interface to manage more than a single iSTAP filtering rule.

The Guardium Db2 for i S-TAP Configuration report and corresponding API (get_istap_config and update_istap_config) can be used to review and change the first filtering rule.

Filtering enhancements

DAM on Db2 for i filtering has been enhanced in several ways to allow customers to have improved, granular control over the auditing detail captured by the Audit Server and stored on the Guardium collector.

1. **Instead of having a single server-side filtering configuration,**
   DAM on i now accommodates up to nine (9) configurations, which are referred to as "filtering rules".

   Multiple policies are OR'd together. When an audit journal entry is produced or an SQL statement is executed, the Audit server will examine the policies to determine if the filtering rule criteria is "matched". If a match is found, the instance is handed to the audit server where it is processed and sent to the collector.

2. **Optional NOT EQUAL support has been added to the following Audit Server filters found within QSYS2/SYSAUDIT:**
   a. FILTER_USER
   b. FILTER_JOB
   c. FILTER_TABLE

3. **Extend the DAM on i audit journal support to be filter-aware.**
   When FILTER_USER, FILTER_JOB or FILTER_TABLE are specified, the filtering criteria applies to audit journal entries, as long as the audit journal entry type does not correspond with a security- oriented audit journal entry.

   Audit journal entry types of type CD, CO, DO, OM, ZC and ZR are the filtering eligible audit entry types.

DAM on i Audit journal entry types:

AD Auditing change	**OR Object restored**
AF Authority failure	**OW Change owner**
AX Row Permissions & Column Masks	**PG Primary group change**
CA Authority change	**PW Invalid password or user ID**
CD Command execution	**RA Restore authority change**
CP Change Profile	**RO Restore owner change**
CO Create object	**RZ Restore primary group change**
DO Delete object	**SV System Value change**
GR General purpose audit record	**ZC Change object**
OM Object moved or renamed	**ZR Read object**

Service level requirements

The best practice is for IBM i clients to remain current with PTF Groups. Db2 for i has its own PTF Group, which is updated periodically each year. To see the schedule and version history of Db2 PTF Groups, look here: https://www.ibm.com/developerworks/ibmi/techupdates/db2/groupptf

To use these filtering enhancements, the following IBM i service needs to be installed:

1. DB2 PTF Group SF99703 – **Level 2 or higher**
   or
   DB2 PTF Group SF99702 – **Level 13 or higher**

   The DB2 PTF Group level can be examined using the WRKPTFGRP command.
   For example: **WRKPTFGRP PTFGRP(SF99702)**
   or
   STRSQL

   SELECT CHAR(PTF_GROUP_NAME,7) as GRPPTF, PTF_GROUP_LEVEL FROM QSYS2.GROUP_PTF_INFO WHERE PTF_GROUP_NAME = 'SF99702' AND PTF_GROUP_STATUS = 'INSTALLED' ORDER BY PTF_GROUP_LEVEL DESC FETCH FIRST 1 ROWS ONLY

2. The Guardium collector must be using V10 (or higher).

Configuring Audit Server Policies

The Audit Server configuration file (QSYS2/SYSAUDIT) is extended to support advanced filtering. If a client has previously used Guardium DAM for i, they already have a copy of QSYS2/SYSAUDIT and the SYSAUDIT file will be upgraded to the format below when the audit server is first started after application of the enabling IBM i service. After being upgraded to the new SYSAUDIT format, the configuration file will still have only a single filtering rule because only a single row exists within the table.

To add additional policies, use the INSERT SQL statement to insert additional rows into QSYS2/SYSAUDIT. To use the INSERT, UPDATE, or DELETE SQL statements against the QSYS2/SYSAUDIT *FILE, the user needs to have the INSERT, UPDATE, or DELETE SQL privilege.

Simply stated, each row in the table corresponds to a unique filtering rule.

Whenever the contents of QSYS2/SYSAUDIT is changed, a restart of the audit server must be performed to promote the changes. Any time you make changes to the SYSAUDIT file, it is recommended that you review the status after restarting the audit server. Restarting the audit server is the only method which confirms that the SYSAUDIT rules are syntactically valid.

QSYS2/SYSAUDIT columns:

Column Name	Column Definition	Description
The following columns are only permitted in the ONE row of SYSAUDIT that contains a non-null value of SERVERNAME. If non-null values are found in other rows, an error will be observed when attempting to start the audit server.		
SERVERNAME	VARCHAR(128) CCSID 1208	This contains the TCP/IP address of the Guardium appliance
FILTER_RDB	VARCHAR(1290) CCSID 1208	The specified relational database filter, if any. Up to 10 relational database names can be specified.
FILTER_AUDIT_ENTRY_TYPES	VARCHAR(1000)	The specified QAUDJRN audit entry filter, if any. Specifies which audit journal entry types should be processed. When DAM for i installed, this column is set to the default of 'AD AF AX CA CO CP DO GD OM OR OW PG PW RA RO RZ SV ZC ZR' Note: You cannot specify NULL or an empty string for this column in the 1st filtering rule. If you do not want Guardium DAM to capture ANY audit journal activity, choose one of the audit journal entry values that is never (or rarely) generated.

		The complete list of supported audit journal entry types: AD – Auditing change AF – Authority failure AX – Row & Column Access Control CA – Authority change CD – Command string CO – Create object CP – Change Profile DO – Delete object GR – General purpose audit record OM – Object moved or renamed PG – Primary group change PW – Invalid password or user ID OW – Change owner OR – Object restored RA – Restore authority change RO – Restore owner change RZ – Restore primary group change SV – System Value change ZC – Change object ZR – Read object
ITAP_PARAM	VARCHAR(1000) NOT NULL	Parameters for controlling ITAP.
DEBUG	CHAR(1)	Debug flag - used to send additional debug messages (Y or N). The default value is DEFAULT 'N'.

PREVENT_SKIPPED_ENTRIES	CHAR(1)	Prevent skipped entries flag - used to prevent entries from being skipped when the queue gets full. The default value is 'N'.
The following columns are the filters that support the option EQUAL and NOT EQUAL filtering controls.		
FILTER_USER	VARCHAR(160) CCSID 1208	A list of up to 10 user, group or generic names, separated by spaces. To use NOT EQUAL filtering, parenthesis must be placed around each filter pair. **Examples:** **-- Capture DBATEAM group and all user names that begin with "JOE"** 'DBATEAM JOE*' **-- Capture the DBATEAM group but avoid activity by the user FRANK** '(DBATEAM *EQ) (FRANK *NE)'
FILTER_JOB	VARCHAR(34) CCSID 1208	A single qualified or generic job name. To use NOT EQUAL filtering, parenthesis must be placed around the job name. **Examples:** **-- Capture activity for QZDASOINIT jobs** '*ALL/*ALL/QZDASOINIT' **-- Capture activity for all jobs by avoid activity for QSQSRVR jobs**' ('*ALL/*ALL/QSQSRVR *NE)'
FILTER_TABLE	VARCHAR(5240) CCSID 1208	A list of up to ten (10) library and table names, separated by spaces. The library name must be used, long SQL schema names are not supported. The table name can be either the SQL table name or short system (file) name. To use NOT EQUAL filtering, parenthesis must be placed around each filter pair. **Examples:** **-- Capture activity for all tables within PRODLIB and the DAM configuration table** '(PRODLIB/*ALL) (QSYS2/SYSAUDIT)' **-- Capture activity for all tables within PRODLIB but avoid capturing activity for table names starting with ACCT** '(PRODLIB/*ALL *EQ) (PRODLIB/ACCT* *NE)
This grouping of filters can be specified to reduce the amount of auditable events collected. The policy name column can optionally be used to uniquely name each filtering rule (row) with a descriptive name.		
POLICY_NAME	VARCHAR(128) CCSID 1208	A new column, containing an optional name the user or the Guardium appliance can use to uniquely identify the filtering rule.
FILTER_TCPIP	VARCHAR(254) CCSID 1208	The specified TCP/IP filter, if any. Only one TCP/IP address can be specified.
FILTER_PORT	INT	The specified port filter, if any. Only one port filter can be specified.
FILTER_CLIENT_ACCTING	VARCHAR(128)	The specified client accounting filter, if any.

	CCSID 1208	Only one client accounting filter can be specified.
FILTER_CLIENT_APPLNAME	VARCHAR(128) CCSID 1208	The specified client application filter, if any. Only one client application filter can be specified.
FILTER_CLIENT_PROGRAMID	VARCHAR(128) CCSID 1208	The specified client program filter, if any. Only one client program filter can be specified.
FILTER_CLIENT_USERID	VARCHAR(128) CCSID 1208	The specified client user filter, if any. Only one client user filter can be specified.
FILTER_CLIENT_WRKSTNNAME	VARCHAR(128) CCSID 1208	The specified client workstation filter, if any. Only one client workstation filter can be specified.
FILTER_SYSTEM_SQL	CHAR(1)	The specified system SQL statement filter. Specifies whether system SQL statements should be audited (Y or N) . The default is Y.
The remaining columns contain values supplied by the Audit Server when started. The values only appear in the row which contains the non-null value for SERVERNAME. These columns will contain NULL for all other rows.		
START_JOB	CHAR(26)	The job name of the most recently started Audit server.
START_TIME	TIMESTAMP	The timestamp when the most recently started Audit server was started.
MONITOR_ID	CHAR(10)	The Database Monitor identifier associated with this filtering rule instance.
START_USER	CHAR(10)	The user that last started the Audit server. This column can be updated to change the user profile responsible for execution of the audit server. The changed value will be used when the Audit server is restarted.

Filtering details:

- For each auditing event, the policies are examined in order until a match is found or the filtering rule list is exhausted. Therefore, to achieve the best performance, the policies should be ordered from least filtered → greatest filtered. In other words, the first row in the table should correspond to the filter that will capture the greatest amount of activity.
- Within a filtering rule (i.e. each row in QSYS2/SYSAUDIT), the different filtering columns are logically ANDed. For example, if a filtering rule is defined where FILTER_JOB contains '*ALL/*ALL/QZDASOINIT' and FILTER USER contains 'MJA', the filtering rule will only send entries to the collector for user MJA activity within QZDASOINIT jobs.
- Within a filter column of a filtering rule, the elements without a NOT EQUAL are logically ORed. For example, if FILTER_USER contains 'MJA SCOTTF', the filtering rule will only send entries to the collector for user MJA **or** SCOTTF.
- Within a filter column of a filtering rule, the elements with a NOT EQUAL are logically ANDed with the other elements. For example, if FILTER_USER contains **'(B*) (BURRICHTER *NE) (SCOTTF)'**, the filtering rule will only send entries to the collector for user SCOTTF and users whose name starts with a B, but not when the user is BURRICHTER.
- Within a filter column of a filtering rule, the “name” elements will be folded to uppercase IF the names are not delimited names.
- Wild cards (*) are only supported at the end of a string.
- *ALL is a special value that can be used for certain elements.
- For each list that does not contain an optional *NE (NOT EQUAL), elements may be separated with one or more blanks and may be enclosed in parenthesis.

- For each list that does contain an optional *NE (NOT EQUAL), elements may be separated with one or more blanks but MUST be enclosed in parenthesis if there is more than a single element in the list.

Protecting the Audit Server Configuration File

Note: The following information is provided to be illustrative of some of the considerations and configuration choices. Every client is encouraged to employ a security expert or to contract with a security expert consultant prior to deploy or changing the security configuration.

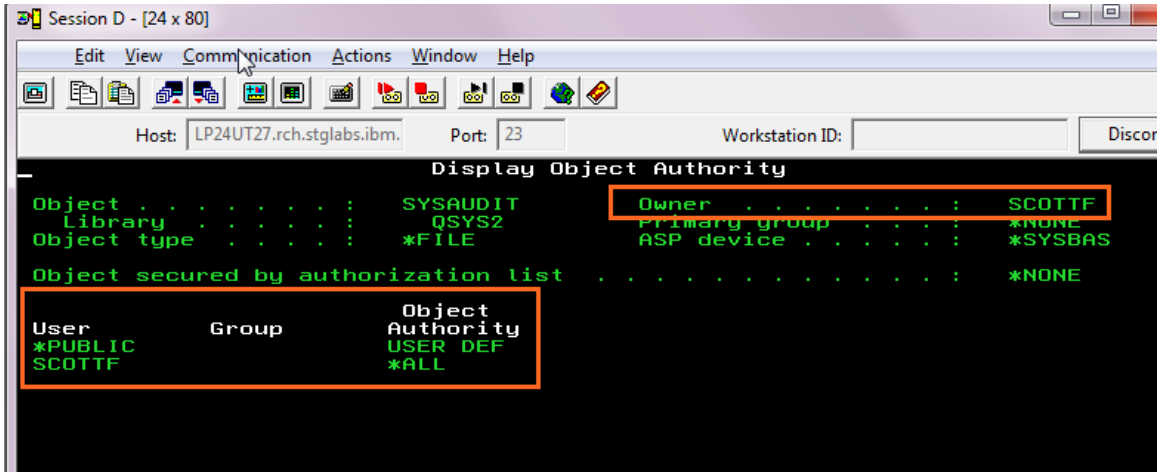
The QSYS2/SYSAUDIT *FILE indicates which database activity (users, jobs, tables, SQL queries, etc...) are monitored by the audit server. Therefore, the SYSAUDIT table is a critical security resource and needs to be both protected and audited.

To protect the QSYS2/SYSAUDIT you need to regularly review the following security settings:

- **Ownership** – By default, the SYSAUDIT table will be owned by whichever user installed the Audit Server PASE program. The owner is permitted to query, change, alter or remove the SYSAUDIT *FILE.
- **Private Authority** – By default, the owner of the file is granted *ALL authority to SYSAUDIT.
- **Public Authority** – By default, any user (i.e. the public) can query SYSAUDIT and discover the filtering strategy.

To review the security authorization of the QSYS2/SYSAUDIT *FILE:
DSPOBJAUT OBJ(QSYS2/SYSAUDIT) OBJTYPE(*FILE) AUTTYPE(*OBJECT)

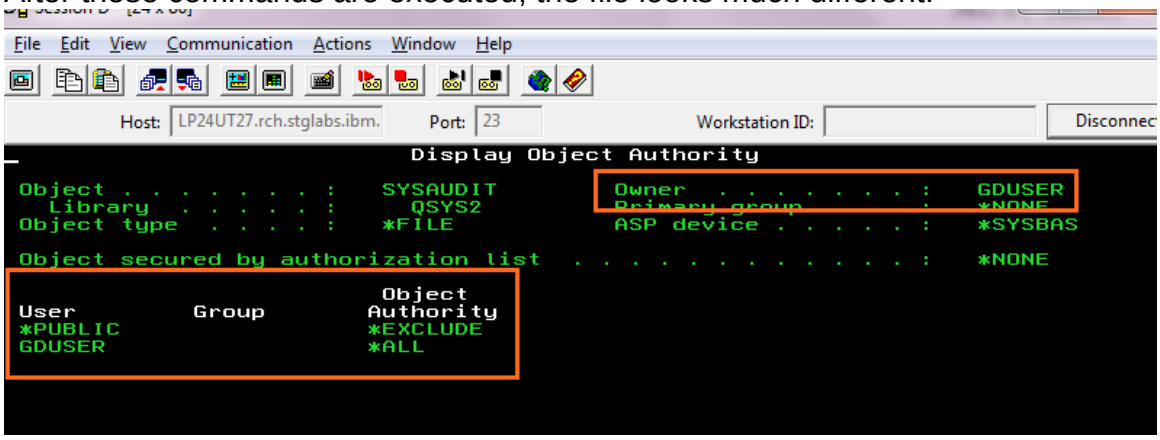
As we see in the image, the user SCOTTF is both the owner and also has private authorities. Also, by default, any user is allowed to query the contents of QSYS2/SYSAUDIT.



To completely lock down and limit access to the SYSAUDIT, we can use the following commands:

- CHGOBJOWN OBJ(QSYS2/SYSAUDIT) OBJTYPE(*FILE) NEWOWN(GDUSER)
- GRTOBJAUT OBJ(QSYS2/SYSAUDIT) OBJTYPE(*FILE) USER(GDUSER) AUT(*ALL)
- RVKOBJAUT OBJ(QSYS2/SYSAUDIT) OBJTYPE(*FILE) USER(*PUBLIC) AUT(*ALL)

After these commands are executed, the file looks much different:



Even though the SYSAUDIT table now is protected from many potential security related exposures, we still need to take additional steps to audit changes to the file. Why? Any user with *ALLOBJ user special authority can change the contents of this file. By enabling SYSAUDIT to generate audit records, we will be able to see any changes appear on the Guardium activity report.

To configure object auditing:

- CHGOBJAUD OBJ(QSYS2/SYSAUDIT) OBJTYPE(*FILE) OBJAUD(*ALL)

Of course, auditing relies upon several other things being set up correctly. For this reason, it is again recommended that you engage a security expert, construct a Guardium activity report for QSYS2/SYSAUDIT, and conduct tests to confirm that this file is properly protected, audited and monitored.

Multiple Filtering Rule – example 1

The following example demonstrates how to configure the audit server to have a multiple filtering rule deployment. In this example, there are two (2) filtering rules:

1. The first row establishes the audit server base configuration, indicates which journal entries are eligible to be captured and uses the FILTER_USER support to track all users within the DBATEAM group profile.
2. The second row simply tracks all activity on the QSYS2/SYSAUDIT configuration file, which is a best practice.

```
create table qgpl.saved_sysaudit as (select * from
qsys2.sysaudit) with data;

delete from qsys2.sysaudit;
```

```
insert into qsys2.sysaudit (FILTER_AUDIT_ENTRY_TYPES,
FILTER_TABLE, FILTER_JOB, FILTER_USER, SERVERNAME, START_USER,
POLICY_NAME)
values('AD AF AX CA CP GR OW PG PW RA RZ SV ZR ZC OM OW OR',
NULL, NULL, 'DBATEAM',
'9.5.12.126', 'GDUSER',
'Security tracking and DBA team tracking');

insert into  qsys2.sysaudit (FILTER_TABLE, POLICY_NAME)
values('(QSYS2/SYSAUDIT *EQ)',
'Guardium configuration table tracking');
```

Multiple Filtering Rule – example 2

The following example demonstrates how to configure the audit server to capture all SQL activity over a set of database files. The SQL activity of some user profiles is filtered (i.e. not captured). In this example, there are three (3) filtering rules:

1. The first row establishes the audit server base configuration, indicates which journal entries are eligible to be captured and uses the FILTER_USER support to track all users within the DBATEAM group profile.
2. The second row simply tracks all activity on the QSYS2/SYSAUDIT configuration file, which is a best practice.

```
create table qgpl.saved_sysaudit as (select * from
qsys2.sysaudit) with data;

delete from qsys2.sysaudit;


insert into  qsys2.sysaudit (SERVERNAME, FILTER_AUDIT_ENTRY_TYPES,
FILTER_USER, FILTER_TABLE,
                             FILTER_SYSTEM_SQL, FILTER_RDB,
START_USER, POLICY_NAME)
values('guardroc.rch.stglabs.ibm.com', 'AX',
'(MIMIX* *NE) (QBRMS *NE) (QSYS *NE) (RBTNETPT *NE) (RBTUSER *NE)
(RBTWEB *NE) (MWIPROD *NE) (PLCM2ADM *NE)',
'(PRODLIB1/FILE1) (PRODLIB1/TABLE2) (PRODLIB1/TABLE3)
(PRODLIB1/TABLE4) (PRODLIB1/TABLE5) (PRODLIB1/TABLE6)
(PRODLIB1/TABLE7) (PRODLIB1/TABLE8) (PRODLIB1/TABLE9)
(PRODLIB1/TABLE10)',
'N', 'LP02UT28 IASPONE', 'GDUSER',
'Rule One');

insert into  qsys2.sysaudit (FILTER_USER, FILTER_TABLE,
POLICY_NAME)
values(
'(MIMIX* *NE) (QBRMS *NE) (QSYS *NE) (RBTNETPT *NE) (RBTUSER *NE)
(RBTWEB *NE) (MWIPROD *NE) (PLCM2ADM *NE)',
'(PRODLIB1/FILE11) (PRODLIB1/TABLE12) (PRODLIB1/TABLE13)
(PRODLIB1/TABLE14) (PRODLIB1/TABLE15) (PRODLIB1/TABLE16)
(PRODLIB1/TABLE17) (PRODLIB1/TABLE18) (PRODLIB1/TABLE19)
(PRODLIB1/TABLE20)',
'Rule Two');

insert into  qsys2.sysaudit (FILTER_USER, FILTER_TABLE,
POLICY_NAME)
values(
```

```
'(MIMIX* *NE) (QBRMS *NE) (QSYS *NE) (RBTNETPT *NE) (RBTUSER *NE)
(RBTWEB *NE) (MWIPROD *NE) (PLCM2ADM *NE)',
'(PRODLIB1/FILE21) (PRODLIB1/TABLE22) (PRODLIB1/TABLE23)
(PRODLIB1/TABLE24) (PRODLIB1/TABLE25) (PRODLIB1/TABLE26)
(PRODLIB1/TABLE27) (PRODLIB1/TABLE28) (PRODLIB1/TABLE29)
(PRODLIB1/TABLE30)',
'Rule Three');

-- Restart the audit server
call sysproc.sysaudit_start_batch('');

-- Review the audit server status
call sysproc.sysaudit_status();
select * from qtemp.sysaudsts
  order by STATUS_TIME DESC;
```

Debugging filter rule user errors

When the configuration file (QSYS2/SYSAUDIT) contains incorrect values or syntax, a failure will occur when an attempt is made to start/restart the audit server. To self-diagnose and correct these failures, you will need to find and study the failure joblog.

If your audit server job description is configured to discard joblogs, change the job description to use *SECLVL. This change will take hold the upon the next restart of the Audit server.

CHGJOBD JOBD(QGPL/GDAUDIT) LOG(4 0 *SECLVL)

Once the failed joblogs are being retained, use the following command to find the joblogs.

WRKSPLF GDUSER

Informative message will appear in the joblog, guiding you to discover which filtering rule or filter criteria is invalidly specified.

Audit Server Status

The status procedure is unchanged. The status and statistics relate to all active DAM policies.

call sysproc/sysaudit_status()
select * from qtemp/sysaudsts

The status file has many columns, so it might be beneficial to only query those column names that are of interest.

General status detail:

STATUS_TIME TIMESTAMP – Time that the sysproc/sysaudit_status() procedure was called to output this row.

SERVER_STARTED CHAR(4) - YES or NO

START_TIME TIMESTAMP – When the server was started (or restarted after a system IPL)

SERVER_JOB CHAR(26) – The jobname of the audit server job.
Note that the format of this column does not match the formatting needed when you work with IBM i commands.

For example: QDFTJOBD RUIYU 410890
Refers to the qualified jobname: 410890/RUIYU/QDFTJOBD

SQL Monitor detail:
This detail does not include the SQL statements that were filtered at the source through the use of one or more database monitor (STRDBMON) filters.

NUMBER_JOBS_AUDITED_USING_SQL BIGINT – The count of the number of different jobs that have sent at least one instance of audit detail to the PASE program.

NUMBER_PROCESSED_SQL_STATEMENTS BIGINT - The count of the number of SQL statements that have been received by the Instead of Trigger program. (QSQGDIOT)

NUMBER_ENQUEUED_SQL_STATEMENTS BIGINT - The count of the number of SQL statements that have been sent to the PASE program by the Instead of Trigger program. (QSQGDIOT)

NUMBER_SKIPPED_SQL_STATEMENTS BIGINT – Indicates the number of SQL statements that could have been sent to the PASE program, but have not been sent. Under normal conditions this value will be zero. When PREVENT_SKIPPED_ENTRIES is set to 'N', each job will attempt to an SQL statement on the queue up to three times then will give up (typically because queue is full).

NUMBER_PROCESSED_VARIABLE_SETS BIGINT – The total number of 3010 variable sets received by the Instead of Trigger program. (QSQGDIOT) The variable sets are the data needed to populate the "Bind Variables Values" column on the Guardium client.

NUMBER_SKIPPED_VARIABLE_SETS BIGINT - The total number of variable sets received, but could not be handed off to the Audit Server. Under normal conditions this value will be zero. When PREVENT_SKIPPED_ENTRIES is set to 'N', we keep a buffer of up to 300 variable sets before we begin to skip variable sets.

QAUDJRN (security journal) detail:
This information does not include the audit entry filtering based upon the configured audit entry types.

NUMBER_PROCESSED_QAUDJRN_ENTRIES BIGINT – Number of individual audit entries received from the QAUDJRN audit journal.

NUMBER_ENQUEUED_QAUDJRN_ENTRIES BIGINT – Number of audit entries sent to the PASE program. Does not include any audit entries received (processed), but deemed not necessary to send to the Guardium collector

NUMBER_SKIPPED_QAUDJRN_ENTRIES BIGINT – Number of audit entries which could not be sent to the Audit Server (PASE program). Under normal conditions this value will be zero. Does not include any audit entries received (processed), but deemed not necessary to send to the Guardium collector. When PREVENT_SKIPPED_ENTRIES is set to 'N', audit entries can be discarded if the Audit Server is unable to receive the detail.

QUEUE detail:
The queue referred to here is the message queue being used to communicate the auditable entries between the server and the PASE program that sends the detail to the Guardium Collector.

QUEUE_DAMAGED CHAR(3) – YES or NO. If YES, level 3 should be contacted.

NUMBER_MESSAGES_ON_QUEUE INTEGER – The number of messages that the Guardium PASE program has NOT consumed. Under normal situations, the number of messages on the queue will be zero indicating that the PASE program is able to keep up with the audit data.

SIZE_OF_MESSAGES_ON_QUEUE INTEGER – Similar to NUMBER_MESSAGES_ON_QUEUE, but a different metric. Frequently zero.

MAXIMUM_SIZE_OF_QUEUE INTEGER – Not configurable and you should see 16,777,216.

TOTAL_ENQUEUING_THREADS INTEGER – Indicates how many different threads are placing audit detail into the message queue.

LAST_DEQUEUE_TIME TIMESTAMP – Indicator that the queue is working.

LAST_ENQUEUE_TIME TIMESTAMP - Indicator that the queue is working.

QUEUE_OWNER CHAR(10) – Not interesting

Monitor end detail:

LAST_END_MONITOR_JOB CHAR(26) – The jobname of the previous instance of the audit server.

LAST_END_MONITOR_USER CHAR(10) – The user name of the user that ended the audit server. When the customer IPLs the machine, the audit server will be stopped and automatically restarted. The user ID that started the audit server will appear here on an IPL.

Notices

2017-September 28

2017-August 25

2015-June 30