Audit and Logging in OpenEMR

Version 2.0
By ViCarePlus Team

Prepared by

ViSolve Inc.,
Contact: 408.666.4320
EMail: vicareplus_engg@visolve.com
www.visolve.com

April 10, 2010

1. Introduction to Logging

OpenEMR is enhanced with extensive logging and auditing functionalities by Visolve. Through logging features, all the events and transactions can be logged for future auditing purposes.

Audit related configurations in globals.php

```
$GLOBALS["enable_auditlog"]=0;
$GLOBALS["audit_events"]=array("patient-record"=>1,
                    "scheduling"=>1,
                    "query"=>0,
                    "order"=>1,
                    "security-administration"=>1,
                    "backup"=>1,
                    );
```

Through enable_auditlog, we can enable/disable the auditing feature by setting that to 1/0. The corresponding audit events can be made enable/disable by configuring the audit_events array.

For ex, if the variable ‘patient-record’ is set to 1, all the events related to patient record will be logged. Like, if ‘query’ is enabled, all the view information (select statements) will be logged.

The type of info logged by the audit events are described below

Patient-record
- Patient Registration
- Adding issues/allergies/diagnosis
- Encounters
- Immunizations
- Billing

Security Administration
- GACL calls
- Facility/User creation
- Practice settings
- ICD9/CPT4/HCPCS configurations

Scheduling
- Calendar categories modification
- Appointment scheduling

Backup

Order
    Prescriptions

2. How logging works

By calling “auditSQLEvent” in sql.inc functions like sqlStatement, sqlInsert and sqlQuery.

Based on the sql query passed, auditSQLEvent identifies the type of audit event and inserts the following information into the “log” table

Date & Time
User
Groupname
Event name
Details of the event

3. Logging new events

The new sql queries which passes through sqlStatement, sqlInsert and sqlQuery will be logged automatically with the event type “others”.

New event names can be introduced by inserting a new entry in the “tables” array in auditSQLEvent function.

For example, if the queries to “lab-results” table need to be identified as “patient-record” event, the following array element can be inserted into the “tables” array.
    “lab-results” => “patient-record”

4. Constraints

While using LAST_INSERT_ID and mysql_insert_id in our future works, we need to make sure that the correct ids are generated since the value of mysql_insert_id will get impacted by the last audit statement executed

Example 1
sqlStatement($query1)
$lastid=mysql_insert_id(GLOBALS[‘dbh’]

sqlStatement will call ‘auditSQLEvent’ which in turn inserts the details of the last database call into the ‘log’ table (insert into log values etc) and might have changed the mysql_insert_id value since “auto increment” field is present in the “log” table.

To avoid this, the value of mysql_insert_id before the audit call is stored in GLOBAL['lastidado'] before executing the audit log call.

sqlInsertClean_audit() in sql.inc function which executes the "log" calls.

```
$lastid=mysql_insert_id($GLOBALS['dbh']);;
$GLOBALS['lastidado']=$lastid;
  //----------run a mysql insert, return the last id generated
  $ret = mysql_query($statement, $GLOBALS['dbh']);
```

We need to keep this in our mind for our future enhancements. Statements like "Example 1" can be rewritten as

```
sqlStatement($query1)
$lastid=$GLOBALS['lastidado']
```

How we've taken care of this in our existing openemr code is given in Appendix A.

Appendix A

Approach followed

Storing the mysql_insert_id in GLOBALS[""] before the audit call
Use the GLOBALS[""] wherever mysql_insert_id is used

Section A - Handling "LAST_INSERT_ID" calls

root@vicare-laptop:/home/selvi/TEST/SF-CVS/openemr# grep -r "LAST_INSERT_ID" .

./interface/main/calendar/pnadodb/drivers/adodb-mysql.inc.php: var $_genIDSQL = "update %s set id=LAST_INSERT_ID(id+1);";
./gacl/adodb/drivers/adodb-mysql.inc.php: var $_genIDSQL = "update %s set id=LAST_INSERT_ID(id+1);";
./gacl/adodb/drivers/adodb-mysqli.inc.php: var $_genIDSQL = "update %s set id=LAST_INSERT_ID(id+1);";
./library/adodb/drivers/adodb-mysql.inc.php: var $_genIDSQL = "update %s set id=LAST_INSERT_ID(id+1);";
./library/adodb/drivers/adodb-mysqli.inc.php: var $_genIDSQL = "update %s set id=LAST_INSERT_ID(id+1);";

- LAST_INSERT_ID calls returns the value of the expression (id) only.
- Impact – No
- Changes - No

./gacl/adodb/drivers/adodb-mysql.inc.php: return ADOConnection::GetOne('SELECT LAST_INSERT_ID()');

- This is called in _InsertID() function.
- Impact – No
- Changes – No
- **We believe that this is not used anywhere.**

/phpmyadmin/libraries/dbi/mysql.dbi.lib.php: return PMA_DBI_fetch_value('SELECT LAST_INSERT_ID();', 0, 0, $link);

- We are not logging phpmyadmin calls.
- Impact – No
- Changes - No

./phpmyadmin/libraries/sqlparser.data.php: 'LAST_INSERT_ID',
./phpmyadmin/libraries/sqlparser.data.php: 'LAST_INSERT_ID',

- Reserved words array. In this context it is a String constant and not a function.
- Impact – No
- Changes - No

./phpmyadmin/libraries/config.default.php: 'LAST_INSERT_ID',
./phpmyadmin/libraries/config.default.php: 'LAST_INSERT_ID',
./phpmyadmin/libraries/config.default.php: 'LAST_INSERT_ID',

- Config array. In this context it is a String constant and not a function.
- Impact – No
- Changes - No

./phpmyadmin/tbl_replace.php: 'LAST_INSERT_ID',

- It is defined as a string constant in an array and it is not called as a function.
- Impact – No
- Changes - No

Section B - Handling "mysql_insert_id" calls

root@vicare-laptop:/home/selvi/TEST/SF-CVS/openemr# grep -r "mysql_insert_id" .

B1 – GenID calls

B11. ./library/adodb/drivers/adodb-mysql.inc.php: $this->genID = mysql_insert_id($this->_connectionID);

- **Here mysql_insert_id() is handled. When the audit is enabled, the value in the GLOBALS is returned; If disabled then mysql_insert_id() is returned.**
- **Impact – Yes**
- **Changes done - Yes**

B12. ./interface/main/calendar/pnadodb/drivers/adodb-mysql.inc.php: return mysql_insert_id($this->_connectionID);

B13. ./gacl/adodb/drivers/adodb-mysql.inc.php: $this->genID = mysql_insert_id($this->_connectionID);

- mysql_insert_id is called in GenID() function after the &Execute() function in gacl/adodb/adodb.inc.php, where we have no log calls.

For B12 & B13, we aren't sure of the impact of the GenID calls made

In order to avoid those GenID calls impact, we've ignored the logging of "sequences" database calls

B2. log.inc and sql.inc files

B21. ./library/log.inc: $rid = mysql_insert_id($GLOBALS['dbh']);

- @ line 484, mysql_insert_id() is called in sql_checksum_of_modified_row() function. This function is called before the audit insert query, which is @ line 653. Obviously, it will always get the last_insert_id() of the query that executed before the audit log insert query.
- Impact – No
- Changes - No

B22. ./library/sql.inc

- **Handled in sqlInsert() & sqlInsertClean_audit() – with the introduction of GLOBALS variable (which holds the mysql_insert_id() before audit log and returning the same).**
- **The function sqlLastID() is not used anywhere in the application.**
- **Impact – Yes**
- **Changes - Yes**

```
root@priya:/var/www/oemr_audit# grep -r "sqlLastID" .
Binary file ./library/.sql.inc.swp matches
./library/sql.inc:function sqlLastID() {
root@priya:/var/www/oemr_audit#
```

B3. Other mysql_insert_id calls

The contrib folder is like a place holder, wherein most of the customizable forms are in /contrib/forms folder. The mysql_insert_id() in these files are also handled. So when a form which is listed in the following is included into the interface/forms, the mysql_insert_id() will not cause any mishap when the audit is enabled.

B 31 ./contrib/forms/phone_exam/save.php:$newid=mysql_insert_id($GLOBALS['dbh']); // last id

- **Impact – Yes**

- **Changes - Yes**

B 32 ./contrib/forms/documents/save.php:$newid=mysql_insert_id($GLOBALS['dbh']); // last id

- **Impact – Yes**
- **Changes - Yes**

B 33 ./contrib/forms/lab_results/save.php:$newid=mysql_insert_id($GLOBALS['dbh']); // last id

- **Impact – Yes**
- **Changes - Yes**

B 34 ./contrib/forms/habits/save.php:$newid=mysql_insert_id($GLOBALS['dbh']); // last id

- **Impact – Yes**
- **Changes - Yes**

B. 35 ./library/patient.inc: $db_id = mysql_insert_id();

- Impact – No
- Changes – No

Reason : $db_id variable gets the return value of mysql_insert_id() function. But this variable is not used. So there is no impact of mysql_insert_id() function used at newPatientData() function in library/patient.inc file when audit feature is enabled.