Subject: Application Crashes Due to Duplicated Concurrent INSERT Transactions
Posted by kong on Tue, 20 Sep 2016 05:25:55 GMT
View Forum Message <> Reply to Message

This is further on the issue that was brought up here http://radicore.org/fud/index.php?t=msg&th=2283&goto=5883&a mp;a mp;a mp;a mp;#msg_5883

Multiple submits of the same ADD1 forms will lead to concurrent INSERT transactions in the database. If the table uses non auto-increment primary keys or has other unique key constraints, this could result in fatal error thrown by the database (in this case MySQL) and application crash.

The checks built into the framework as described here http://www.tonymarston.net/php-mysql/functions-and-variables.html#n otes._dml_insertrecord should have prevented such crash and presented the user with user-friendly error messages instead. However, under certain race conditions which are hard to replicate, multiple submits of the same form in rapid succession could result in concurrent transactions invoked by calling startTransaction() function followed by insertRecord() method in the controller script for every click on the submit button. If you are unlucky, you could end up with these concurrent insertRecord transactions all working off the same pre-transaction snapshot of the database table and hence each transaction passing the above mentioned framework built-in checks without raising any errors, until one transaction has been committed successfully but then is followed by another transaction's INSERT which in turn makes the database throw a Duplicate Key Fatal Error, crashing the application.

To solve this problem, there are several options I can think of:
1) Explicitly set database table lock before the transaction starts by customizing _cm_getDatabaseLock() for every table with non auto-increment primary key or other unique keys.
2) Use a hidden field token to guard against multiple submissions of the same form, for example as shown in http://docstore.mik.ua/orelly/webprog/pcook/ch09_06.htm and http://phpsense.com/2006/prevent-duplicate-form-submission/.
3) Since we are relying on the framework for validations and unique key constraint checking anyway, we might as well use a different SQL insert command that does not throw Unique Key Fatal Errors on insert. For example in MySQL use INSERT IGNORE instead of INSERT.
4) Temporary lower the Isolation Level for Transactions in the database before the transaction begins. For example, in MySQL, lower from default level 3 'REPEATABLE READ' to level 1 'READ UNCOMMITTED'.

In my case I am using option 3 as that is least amount of work. But for the long run and improving robustness of the framework I think the 2nd option could be worth some effort.