Redmine - Defect #19040

Potential DB deadlocks on concurrent issue creation

2015-02-05 15:55 - Serghei Zagorinyak

Status:	Closed	**Start date:**	
Priority:	Normal	**Due date:**	
Assignee:	Jean-Philippe Lang	**% Done:**	0%
Category:	Database	**Estimated time:**	0.00 hour
Target version:	3.0.0		
Resolution:	Fixed	**Affected version:**	

Description

We're running Redmine 2.5.0 with approx 120 users backed by an MS SQL database. The problem we have encountered is that from time to time issue creation fails because of DB deadlocks.
To reproduce this we created a python script that would start 7 threads and fire POST requests to create new issues in Redmine. Pretty soon 6 out of 7 threads were dead because of deadlocks on issue validation.

TinyTds::Error: Transaction (Process ID 227) was deadlocked on lock resources with another process and has been chosen as the deadlock victim. Rerun the transaction.: EXEC sp_executesql N'SELECT [issues].* FROM [issues] WHERE [issues].[parent_id] = 22244 ORDER BY [lft] ASC'
Issue Load (5027.7ms) EXEC sp_executesql N'SELECT [issues].* FROM [issues] WHERE [issues].[parent_id] = 22244 ORDER BY [lft] ASC'

After that we turned on READ_COMMITTED_SNAPSHOT option in DB that allows to read transaction data even if it hasn't been committed yet, and this helped us move one step forward: deadlocks began to happen on INSERTs when new issues were to be created. Still it didn't help to solve the problem.

Is it possible to soften lock conditions or reduce transaction sizes as obviously concurrency suffers from that?

Related issues:

Related to Redmine - Feature #18860: Replace awesome_nested_set gem with a cu...	**Closed**	
Related to Redmine - Defect #6579: Tree hierachy being currupted on multiple ...	**Closed**	**2010-10-05**

History

#1 - 2015-02-06 02:52 - Toshi MARUYAMA

- Related to Feature #18860: Replace awesome_nested_set gem with a custom implementation of nested sets added

#2 - 2015-02-06 02:52 - Toshi MARUYAMA

- Related to Defect #6579: Tree hierachy being currupted on multiple submissions of an issue added

#3 - 2015-02-06 02:56 - Toshi MARUYAMA

- Subject changed from DB deadlocks on concurrent user requests to MS SQL Server deadlocks on concurrent user requests

#4 - 2015-02-06 02:58 - Toshi MARUYAMA

It seems awesome_nested_set does not catch MS SQL server dead lock.
https://github.com/collectiveidea/awesome_nested_set/blob/v2.1.5/lib/awesome_nested_set/awesome_nested_set.rb#L553

#5 - 2015-02-06 16:25 - Serghei Zagorinyak

As I'm not really familiar with RoR, I started a clone of Redmine, went to /lib/plugins/awesome_nested_set/lib/awesome_nested_set/model/transactable.rb and edited the def in_tenacious_transaction(&block) method like this (added logging and rescue Exception instead of what was there):

```
    def in_tenacious_transaction(&block)

            logger = Logger.new(File.open(Rails.root.join('log/deadlock_msg.log'), 'a', sync: true))
            logger.formatter = Logger::Formatter.new
            logger.info "in transaction"

            retry_count = 0
            begin
              transaction(&block)
```

```
            rescue Exception => error
              logger.error "exception happened"
              logger.error "error: #{error.message}"
              logger.error error.backtrace
              raise unless connection.open_transactions.zero?
              raise unless error.message =~
/Deadlock found when trying to get lock|Lock wait timeout exceeded|has been chosen as the deadlock victim/
              raise unless retry_count < 10
              retry_count += 1
              logger.info "Deadlock detected on retry #{retry_count}, restarting transaction"
              sleep(rand(retry_count)*0.1) # Aloha protocol
              retry
            end
          end
```

What confuses me is that the only records I see in this log file are the "in transaction" ones, but no error messages which I expected to see. Still I see deadlock errors in production.log.

#6 - 2015-02-08 16:03 - Jean-Philippe Lang

- Subject changed from MS SQL Server deadlocks on concurrent user requests to Potential server deadlocks on concurrent issue creation

- Status changed from New to Closed

- Assignee set to Jean-Philippe Lang

- Target version set to 3.0.0

- Resolution set to Fixed

This problem is not SQLServer specific. With PostgreSQL set to french locale, dead locks also occur.

This is fixed in 3.0 where awesome_nested_set is replaced with a custom implementation of nested sets that should properly set locks to prevent dead locks. A test with concurrent issue creation/deletion was added to demonstrate and was failing with awesome_nested_set:
source:/trunk/test/unit/issue_nested_set_concurrency_test.rb

#7 - 2015-02-08 16:03 - Jean-Philippe Lang

- Subject changed from Potential server deadlocks on concurrent issue creation to Potential DB deadlocks on concurrent issue creation

#8 - 2015-02-20 05:30 - Toshi MARUYAMA

I created issue.
https://github.com/collectiveidea/awesome_nested_set/issues/298