MySQL Logon trigger

Oli Sennhauser

Rebenweg 6
CH – 8610 Uster
Switzerland
oli.sennhauser@bluewin.ch

Introduction

With MySQL 5.0 the database provides trigger functionality on INSERT, REPLACE, UPDATE and DELETE.
Those from you who know some other RDBMS know, that there are also some system events where one would like to have triggers.
For example the following triggers:

Event or DDL statement	When allowed or applicable
STARTUP	AFTER
SHUTDOWN	BEFORE
SERVERERROR	AFTER
LOGON	AFTER
LOGOFF	BEFORE
CREATE	BEFORE and AFTER
DROP	BEFORE and AFTER
ALTER	BEFORE and AFTER

Unfortunately MySQL does not (yet) provide such functionality. This is sad because as database administrator this would be sometimes very helpful.
But you can build your own LOGON and STARTUP trigger.
MySQL provides some hooks for these three events:

- init_connect
- init_file
- init_slave

In the my.cnf file you can add a SQL script file which should be executed on database startup (init_file) and during a connect or a reconnect of a slave a command, listed in init_connect and init_slave is executed.
These hooks can be used to build the LOGON and the STARTUP trigger.
Unfortunately there are no hooks for LOGOFF and SHUTDOWN.

Implementation of the LOGON trigger

First we create a table where the information of the LOGON should be stored:

```
CREATE TABLE test.logon_log (
    id            INT UNSIGNED NOT NULL AUTO_INCREMENT
  , connection_id INT UNSIGNED NOT NULL
  , login_ts      TIMESTAMP    NOT NULL
  , `schema`      VARCHAR(64)  NULL
  , user          VARCHAR(77)  NOT NULL
  , PRIMARY KEY (id)
);
```

Then we create a procedure which should be executed when the event happens:

```
# DROP PROCEDURE test.logon_trigger;

DELIMITER //

CREATE PROCEDURE test.logon_trigger()
       SQL SECURITY DEFINER
BEGIN
  INSERT INTO test.logon_log (connection_id, login_ts
                            , `schema`, user)
  VALUES (CONNECTION_ID(), CURRENT_TIMESTAMP()
        , SCHEMA(), SESSION_USER());
END;
//

DELIMITER ;
```

Now we can test, if the whole works as expected:

```
SELECT * FROM test.logon_log;
SHOW GLOBAL VARIABLES LIKE 'init%';
CALL test.logon_trigger;
SELECT * FROM test.logon_log;
```

When everything works fine, the hook must be connected to the procedure to build the trigger:

```
SET GLOBAL init_connect='CALL test.logon_trigger()';
```

A big disadvantage of this solution is, that you have to grant the EXECUTE privileges to each user you want to monitor in this way.

```
GRANT EXECUTE ON PROCEDURE test.logon_trigger
   TO 'testuser'@'localhost';
GRANT EXECUTE ON PROCEDURE test.logon_trigger
   TO 'testuser'@'%';
```

If you do not do that you will find some error messages in the MySQL error log.
Finally you should see something like this in you logon table:

```
+----+---------------+---------------------+--------+---------------------+
| id | connection_id | login_ts            | schema | user                |
+----+---------------+---------------------+--------+---------------------+
|  1 |             1 | 2007-05-25 11:04:36 | test   | root@localhost      |
|  2 |            37 | 2007-05-25 11:05:01 | test   | testuser@localhost  |
|  3 |            39 | 2007-05-25 11:05:47 | test   | testuser@localhost  |
|  4 |            41 | 2007-05-25 11:06:36 | test   | testuser@master     |
+----+---------------+---------------------+--------+---------------------+
```

Do NOT forget to fix the hook in the my.cnf file. Otherwise it is not activated after the next system restart.

Limitation
This logon trigger does not work with users who have SUPER privileges.

Implementation of the STARTUP trigger

Maybe a little bit better known is the startup functionality it is based on the init_file hook.
First, also here, we create a table where the information should be stored:

```
CREATE TABLE test.startup_log (
    id            INT UNSIGNED NOT NULL AUTO_INCREMENT
  , login_ts      TIMESTAMP    NOT NULL
  , PRIMARY KEY (id)
);
```

Then we create a procedure which should be executed when the event happens:

```
# DROP PROCEDURE test.startup_trigger;

DELIMITER //

CREATE PROCEDURE test.startup_trigger()
      SQL SECURITY DEFINER
BEGIN
  INSERT INTO test.startup_log (login_ts)
  VALUES (CURRENT_TIMESTAMP());
END;
//

DELIMITER ;
```

Also here we do a little test to see if it works:

```
SELECT * FROM test.startup_log;
CALL test.startup_trigger;
SELECT * FROM test.startup_log;
```

Then we create a script startup.sql which should be executed during startup and hook it into MySQL to build the STARTUP trigger:

```
init_file = '.../startup.sql';

# startup.sql
CALL test.startup_trigger();
```

After this we have to restart the database. And it look like it works. :-)
Even if the mysqld process dies and the angle process (safe_mysqld) restarts the mysql process it works.

Outlook

With the LOGON trigger we are able to monitor user connections: Who connected when.
With the STARTUP trigger we can already collect some information about database uptime for example to prove some availability data for SLA.
It would be really nice if there would be some hooks for LOGOFF and SHUTDOWN implemented. Then some real session monitoring and resource accounting could be implemented.