Making the Hospital Integrity Agency look corrupted

Introduction

My database models an agency which investigates hospitals, doctors and events such as briberies and malpractices.

The agency is organized hierarchically in regional and local agencies. It also stores integrity agents, patients, doctors, hospitals and insurance providers.

The idea behind this attack is that such an organization, in order to be trusted, should have a clean image and seem honest. Someone who wants to do it harm could try to make it look like the agency itself is corrupted, because a corrupted integrity agency cannot be trusted to investigate cases of corruption.

Description of the attack

We have an application for managing pair of 1:n tables in the agency's database. The names of the tables, and the primary/foreign keys binding them, are specified in a configuration file.

Firstly, the configuration file has many keys, one of them being "childTable". A proper value for it would be "LOCAL_AGENCIES."

Secondly, after the application's form is loaded, the following query is executed:
`"SELECT * FROM " + childTable`
where "childTable" is the value of the key with that name. This is vulnerable because of the string addition.

In the configuration file, if we put a semicolon at the beginning of the value of the "childTable" key, the text which follows the semicolon will be executed as a query when the application is started. Thus, we can take full control of the database.

Once we take control of the database, we will do the following subtle modification: change the budget of all local agencies by 2%. It is a change which has high chances not to be observed immediately, but it could have long-term consequences if done periodically. For example, a financial audit might drastically sanction the agency for this evasion.

If it is possible to upload the configuration file to the machines of the agents which use the application, it would be a good idea to only do the update with a probability of 2% or 5%, such that the changes are not immediately visible. We could define the key in the following way:

```
<add key="childTable" value="LOCAL_AGENCIES;
                    IF RAND() &lt; 0.5
                    BEGIN
                          UPDATE LOCAL_AGENCIES
                          SET lBudget=lBudget*102/100
```

```
                    END
" />
```

Possible major improvement: create a trigger, on update or insert, which does the discussed change to the budgets. This will not necessitate gaining control of any agent's machine, as the trigger can be created on our machines. This way, each time an agent does changes to the database, there will be a small probability that some numbers in it are slightly altered. This could be achieved in a way similar to the one presented below, which unfortunately is not yet functional:

```
<add key="childTable" value="LOCAL_AGENCIES; GO
        CREATE TRIGGER update_logs
              ON LOCAL_AGENCIES
              AFTER UPDATE
              AS
                    IF RAND() &lt; 0.5
                    BEGIN
                          UPDATE LOCAL_AGENCIES
                          SET lBudget=lBudget*102/100
                    END
" />
```

Runtime error encountered:

```
System.Data.SqlClient.SqlException: 'Incorrect syntax near 'GO'.
'CREATE TRIGGER' must be the first statement in a query batch.'
```