Déclencheurs de type LOGON

par Etienne ZINZINDOHOUE () (Blog)

Date de publication : 14 août 2010

Dernière mise à jour :

I - Introduction

Depuis la version 2005 de SQL SERVER, Microsoft a introduit le trigger (déclencheur) de connexion (LOGON). Chaque fois qu'un utilisateur ou une application se connecte à une instance SQL SERVER, l'évènement LOGON est levé et provoque ainsi l'activation du déclencheur de connexion.
Ce couple "évènement-déclencheur" de connexion peut être utilisé pour diverses raisons. En voici quelques unes :

- mettre en place l'historique des accès à une instance SQL SERVER
- appliquer des règles particulières à un compte particulier.
- imposer un nombre maximal de sessions pour un compte (au-delà de ce nombre, toute tentative de connexion à SQL SERVER par ce compte sera rejetée)

Dans cet article, nous allons voir différentes manières d'utiliser le trigger LOGON :

- auditer par exemple le compte "sa" SQL SERVER
- limiter le nombre de session que peut ouvrir un compte
- examiner enfin les problèmes courants à l'utilisation du trigger LOGON

II - Audit du compte sa

Nous envisageons ici d'avoir l'historique de connexion du compte "sa" à une instance SQL SERVER. Pour ce faire, nous allons créer dans la base **master** une table **audit_loginsa** qui va collecter l'historique des connexions du compte concerné.

II-A - Création de la table de collecte

La DDL de la table de collecte est la suivante :

```
USE master
GO
CREATE TABLE [dbo].[audit_loginsa](
 [loginName] [varchar](50) NULL,
 [loginType] [varchar](50) NULL,
 [loginTime] [datetime] NULL,
 [hostUser] [varchar](50) NULL
) ON [PRIMARY]

GO
```

Le descriptif des colonnes de la table audit_loginsa est la suivante :

Colonne	Description	Commentaire
loginName	Nom du login	Dans notre exemple ce champ a pour valeur "sa"
loginType	Type d'authentification	2 types d'authentification : SQL ou Windows
loginTime	Date et heure de connexion	
hostUser	Nom ou adresse IP du poste utilisateur	Nom ou l'IP de la machine à partir duquel la connexion est établie

II-B - Création du trigger de connexion

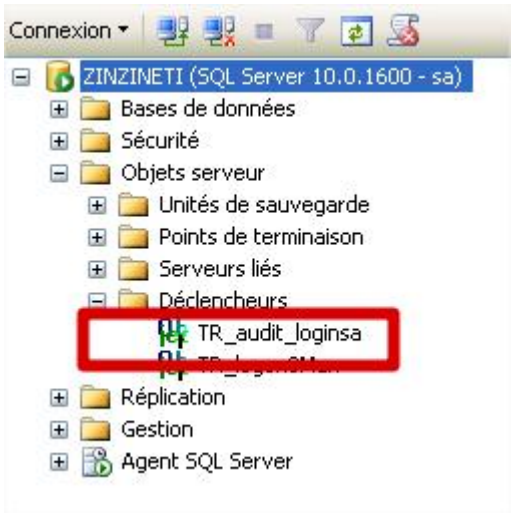
```
-- Creation du trigger : l'idée ici c'est de tracer le compte 'sa'
CREATE TRIGGER TR_audit_loginsa
ON ALL SERVER
FOR LOGON
AS
BEGIN
	DECLARE @DataTrigger XML
	SET @DataTrigger = EVENTDATA() ;

	IF ORIGINAL_LOGIN()= 'sa'
	INSERT INTO master..audit_loginsa
	SELECT @DataTrigger.value('(/EVENT_INSTANCE/LoginName)[1]', 'varchar(50)'),
	 @DataTrigger.value('(/EVENT_INSTANCE/LoginType)[1]', 'varchar(50)'),
		@DataTrigger.value('(/EVENT_INSTANCE/PostTime)[1]', 'datetime'),
		@DataTrigger.value('(/EVENT_INSTANCE/ClientHost)[1]', 'varchar(50)')
END
```

- TR_audit_loginsa : nom du trigger
- ON ALL SERVER : la portée(l'étendue) du déclencheur
- FOR LOGON : déclenchement à la connexion
- DECLARE @DataTrigger XML :pour capturer les données XML renvoyées par EVENTDATA()
- L'événement LOGON retourne le schéma des données XML suivant :

```
<EVENT_INSTANCE>
    <EventType>event_type</EventType>
    <PostTime>post_time</PostTime>
    <SPID>spid</SPID>
    <ServerName>server_name</ServerName>
    <LoginName>login_name</LoginName>
    <LoginType>login_type</LoginType>
    <SID>sid</SID>
    <ClientHost>client_host</ClientHost>
    <IsPooled>is_pooled</IsPooled>
</EVENT_INSTANCE>
```

II-C - Comment visualiser le trigger créé ?

Comment peut-on visualiser ce trigger qui a une portée (étendue) SERVER (ON ALL SERVER) ? Le premier réflexe consiste à aller voir dans la base master, dans le dossier Déclencheur : non ce n'est pas là qu'il faut le chercher ! Ce trigger est visible dans le dossier Déclencheurs sous Objets serveur :



La requête suivante permet d'afficher les caractéristiques du trigger créé :

```
SELECT name,
parent_class_desc 'Class',
tr.Type,
tr_ev.Type_desc + '_' + tr.Type_desc 'Trigger_Type_Desc',
is_ms_shipped,is_disabled
FROM master.sys.server_triggers tr
Inner Join master.sys.server_trigger_events tr_ev
on tr.object_id = tr_ev.object_id
```

Le résultat est le suivant :

Résultats | Messages

	name	class	Type	trigger_type_desc	is_ms_shipped	is_disabled
1	TR_audit_loginsa	SERVER	TR	LOGON_SQL_TRIGGER	0	0

II-D - Test de connexion du compte "sa"

Essayons de nous connecter avec le compte "sa" avec SQL Server Management Studio :

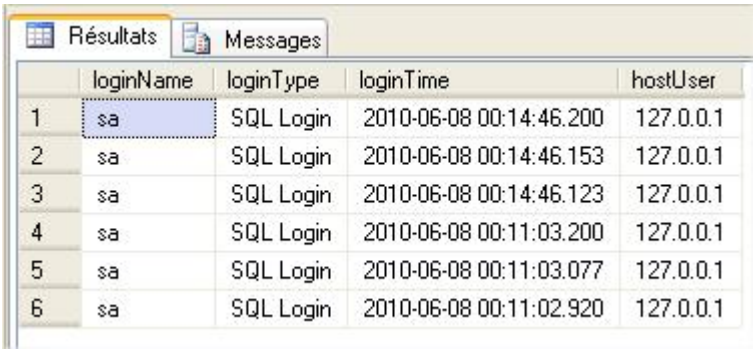
La connexion s'est correctement effectuée. Déconnectons-nous et essayons de se connecter à nouveau à la même instance SQL SERVER, mais cette fois si avec un autre compte (authentification Windows ou un autre autre compte SQL). Dans mon cas j'utilise un autre compte SQL :

II-E - Audit des connexions

Les connexions avec le compte "sa" peuvent être auditées dès à présent à l'aide de la requête suivante :

```
-- Audit du login 'sa'
SELECT * FROM master.dbo.audit_loginsa
ORDER BY loginTime DESC
```

Le résultat est le suivant :



Résultats | Messages

	loginName	loginType	loginTime	hostUser
1	sa	SQL Login	2010-06-08 00:14:46.200	127.0.0.1
2	sa	SQL Login	2010-06-08 00:14:46.153	127.0.0.1
3	sa	SQL Login	2010-06-08 00:14:46.123	127.0.0.1
4	sa	SQL Login	2010-06-08 00:11:03.200	127.0.0.1
5	sa	SQL Login	2010-06-08 00:11:03.077	127.0.0.1
6	sa	SQL Login	2010-06-08 00:11:02.920	127.0.0.1

On observe ici l'historique de connexion du compte "sa". On constate bien que seul ce compte est audité.

III - Limiter le nombre de sessions ouvertes à l'aide du trigger LOGON

Le trigger LOGON permet également de limiter le nombre de sessions ouvertes avec un compte déterminé. Pour mettre ceci en évidence, créons d'abord un compte qu'on va nommer "logon3Max" :

```
-- Créer le compte 'logon3Max'
USE master;
GO
CREATE LOGIN logon3Max WITH PASSWORD = 'pwdlogon3Max'
GRANT VIEW SERVER STATE TO logon3Max;
GO
```

III-A - Création du trigger

```
-- Limiter le nombre de connexion au server à 3 pour le compte 'logon3Max'
CREATE TRIGGER TR_logon3Max
ON ALL SERVER
FOR LOGON
AS
BEGIN
    IF ORIGINAL_LOGIN()= 'logon3Max' AND
    (SELECT COUNT(*)
     FROM sys.dm_exec_sessions
     WHERE is_user_process = 1 AND original_login_name = 'logon3Max') > 3
     ROLLBACK;
END
```

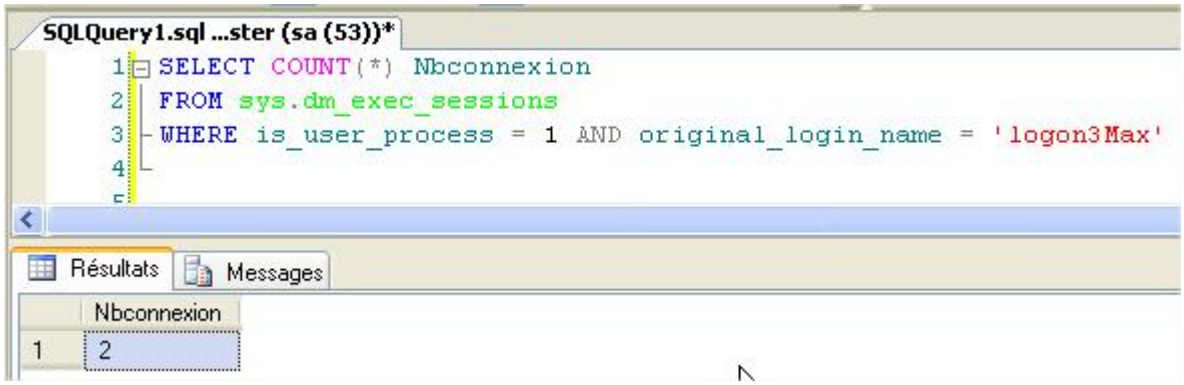
III-B - Test du trigger

Première session avec le compte "logon3Max" :

La requête suivante indique qu'il n'existe qu'une seule session associée au compte "logon3max" pour le moment :

```
SELECT COUNT(*) Nbconnexion
FROM sys.dm_exec_sessions
WHERE is_user_process = 1 AND original_login_name = 'logon3Max'
```

Deuxième session avec le compte "logon3Max". La requête précédente indique qu'il existe 2 sessions pour le compte "logon3Max" :



Répétons les mêmes étapes jusqu'à la 4ème session. Pour cette dernière un message d'erreur apparaît :

Comment peut-on débloquer cette situation ?
Cette question nous amène à examiner quelques problèmes que l'on peut rencontrer lorsqu'on utilise ce type de déclencheur.

Je me souviens de la première fois où j'ai eu un problème avec le trigger LOGON, c'était sur PC portable et j'avais créé un déclencheur LOGIN avec une erreur de frappe. J'ai passé des heures à m'arracher les cheveux ;-) . Le principal problème avec le trigger LOGON (ON ALL SERVER) est le blocage de la session SQL SERVER. Ceci peut arriver dans les cas suivants :

- Vous faites une erreur dans la requête de création du trigger LOGON ayant une portée SERVER (ON ALL SERVER)
- Vous faites une erreur lors de la mise à jour du trigger

Prenons un cas concret :

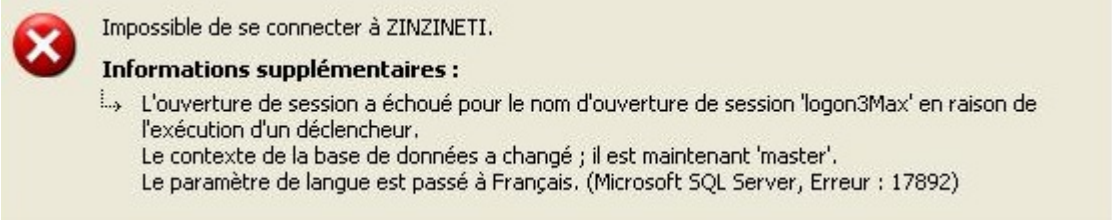
```
CREATE TRIGGER TR_logon3Max
ON ALL SERVER
FOR LOGON
AS
BEGIN
    IF ORIGINAL_LOGIN()= 'logon3Max' AND
    (SELECT COUNT(*)
     FROM sys.dm_exec_sessions
     WHERE is_user_process = 1 AND original_login_name = 'logon3Max') > 3
     ROLLBACK;
END
```

Supposons maintenant qu'il existe une erreur de frappe sur de la vue *sys.dm_exec_sessions* comme ci-dessous :

```
ALTER TRIGGER TR_logon3Max
ON ALL SERVER
FOR LOGON
AS
BEGIN
    IF ORIGINAL_LOGIN()= 'logon3Max' AND
    (SELECT COUNT(*)
     FROM sys.dm_exec_session
     WHERE is_user_process = 1 AND original_login_name = 'logon3Max') > 3
     ROLLBACK;
END
```

A l'exécution de cette commande SQL Server Management Studio affiche "commande réussie " ! et c'est à partir de là que tous les problèmes commencent !

Essayons maintenant de nous connecter à l'instance avec les différents types de compte en notre possession (sa, logon3Max et l'authentification Windows par défaut). Un message d'erreur apparaît pour les 3 connexions :



Impossible donc de se connecter à l'instance ! Si vous avez ce problème sur une base en production, les conséquences seront désastreuses : aucune application, aucun utilisateur ne pourront accéder à la base !!! Vous devez donc savoir très rapidement comment remédier à ce problème d'accès à SQL SERVER.

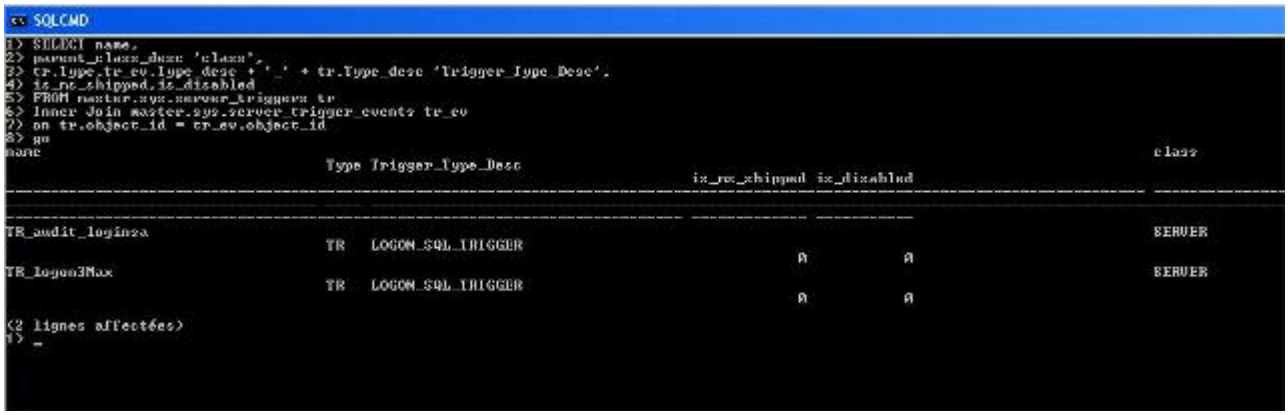
Comment régler ce problème ?

Il faut d'abord se connecter à l'instance SQL Server avec une connexion administrateur dédiée (DAC). Pour cela on peut lancer une commande DOS comme-ci dessous :

Il faut ensuite afficher la liste des triggers LOGON de l'instance à l'aide les vues systèmes **sys.server_triggers** et **sys.server_events**.

```
SELECT name,
parent_class_desc 'class',
tr.Type,tr_ev.Type_desc + '_' + tr.Type_desc 'Trigger_Type_Desc',
is_ms_shipped,is_disabled
FROM master.sys.server_triggers tr
Inner Join master.sys.server_trigger_events tr_ev
on tr.object_id = tr_ev.object_id
```

Le résultat est le suivant :



On visualise bien les deux triggers que l'on avait créés : *TR_audit_loginsa* et *TR_logon3Max*. Une solution rapide consiste à désactiver l'ensemble des triggers de type LOGON si on ne connaît pas le trigger à l'origine du problème. La commande suivante permet de désactiver tous les triggers de type LOGON de l'instance SQL Server :

```
Disable Trigger All ON ALL Server;
```

Cependant dans notre cas, on sait que c'est le trigger **TR_logon3Max** qui est la source du problème. On va donc juste le désactiver à l'aide de la commande suivante :

```
Disable Trigger TR_logon3Max ON ALL Server;
```

Une fois le trigger désactivé on peut éditer afin de voir où se situe exactement l'erreur et le modifier en conséquence

```
SELECT definition
FROM master.sys.server_sql_modules sq
Inner Join master.sys.server_triggers tr on sq.object_id = tr.object_id
```

Une méthode violente existe également et consiste à supprimer le trigger à l'origine du problème. Attention cette méthode n'est à utiliser qu'en dernier recours !!

```
DROP TRIGGER TR_logon3Max ON ALL SERVER;
```