**Auditing for New SQL Server Agent Jobs**

By: [K. Brian Kelley](https://www.mssqltips.com/sqlserverauthor/25/k.-brian-kelley/)

FROM: <https://www.mssqltips.com/sqlservertip/5095/auditing-for-new-sql-server-agent-jobs/>

**Problem**

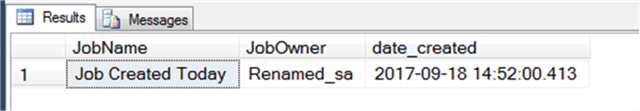
Reading through the various ways someone can run code as a sysadmin on SQL Server, I realized that someone could create a [SQL Server Agent Job](https://www.mssqltips.com/sql-server-tip-category/27/sql-server-agent/) to run and make its owner the [sa account](https://www.mssqltips.com/sqlservertip/3695/best-practices-to-secure-the-sql-server-sa-account/) or another [sysadmin](https://www.mssqltips.com/sqlservertip/1887/understanding-sql-server-fixed-server-roles/) level account. Then the code would run with those privileges. How do I audit for these newly created jobs?

**Solution**

The good news is that all of the job information we care about is in the **msdb** database. As a result, with the result of a few queries, we can detect the creation of new jobs. We’re interested in the *sysjobs* table in msdb, along with sys.server\_principals, so that we can determine what jobs are newly created. Here we’re looking for any jobs created within the last day:

SELECT j.[name] AS JobName, l.[name] AS JobOwner,  
 j.date\_created  
FROM msdb.dbo.sysjobs AS J  
  JOIN sys.server\_principals AS L  
    ON J.owner\_sid = L.[sid]  
WHERE j.date\_created > DATEADD(DAY, -1, GETDATE());

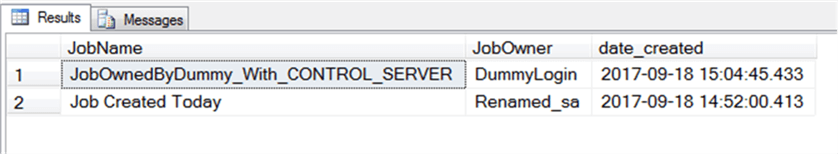
It just so happens that I have such a job, and the query returns a result:



We could tweak the query to only return jobs owned by members of sysadmin, but then we’d also need to tweak the query to return any who had the [CONTROL SERVER permissions](https://www.mssqltips.com/sqlservertip/2634/potential-security-exploit-using-control-server-permissions-in-sql-server/), too. We can do so like so, but my preference is to return any created job, unless I know that jobs are being created automatically, such as with SQL Server Reporting Services and subscriptions.

SELECT j.[name] AS JobName, AllLogins.[name] AS JobOwner,  
 j.date\_created  
FROM msdb.dbo.sysjobs AS J  
 JOIN (  
 -- Get all members of sysadmin  
 SELECT L.[name], L.[sid]  
 FROM sys.server\_principals AS L  
 JOIN sys.server\_role\_members AS RM  
   ON L.principal\_id = RM.member\_principal\_id  
 JOIN sys.server\_principals AS R  
   ON RM.role\_principal\_id = R.principal\_id  
 WHERE R.[name] = 'sysadmin'  
  
 UNION  
  
 -- Control Server  
 SELECT L.[name], L.[sid]  
 FROM sys.server\_principals AS L  
 JOIN sys.server\_permissions AS Perm  
   ON L.principal\_id = Perm.grantee\_principal\_id  
 WHERE Perm.[type] = 'CL' AND Perm.[state] = 'G'  
 ) AS AllLogins  
 ON J.[owner\_sid] = AllLogins.[sid]  
WHERE j.date\_created > DATEADD(DAY, -1, GETDATE());

I intentionally created one job that’s owned by a login with [CONTROL SERVER](https://www.mssqltips.com/sqlservertip/2634/potential-security-exploit-using-control-server-permissions-in-sql-server/), but which isn’t a member of the sysadmin fixed server role:



Modified Jobs

Another attack vector is to simply modify an existing job. Since looking at the modified date can tell us when someone has modified a job, we can modify the queries from before. For all jobs that have been changed (this will also pick up new jobs):

SELECT j.[name] AS JobName, l.[name] AS JobOwner,  
 j.date\_modified  
FROM msdb.dbo.sysjobs AS J  
  JOIN sys.server\_principals AS L  
    ON J.owner\_sid = L.[sid]  
WHERE j.date\_modified > DATEADD(DAY, -1, GETDATE());

And for cases when we’re looking for the specific privileged logins as owners:

SELECT j.[name] AS JobName, AllLogins.[name] AS JobOwner,  
 j.date\_modified  
FROM msdb.dbo.sysjobs AS J  
 JOIN (  
 -- Get all members of sysadmin  
 SELECT L.[name], L.[sid]  
 FROM sys.server\_principals AS L  
 JOIN sys.server\_role\_members AS RM  
   ON L.principal\_id = RM.member\_principal\_id  
 JOIN sys.server\_principals AS R  
   ON RM.role\_principal\_id = R.principal\_id  
 WHERE R.[name] = 'sysadmin'  
  
 UNION  
  
 -- Control Server  
 SELECT L.[name], L.[sid]  
 FROM sys.server\_principals AS L  
 JOIN sys.server\_permissions AS Perm  
   ON L.principal\_id = Perm.grantee\_principal\_id  
 WHERE Perm.[type] = 'CL' AND Perm.[state] = 'G'  
 ) AS AllLogins  
 ON J.[owner\_sid] = AllLogins.[sid]  
WHERE j.date\_modified > DATEADD(DAY, -1, GETDATE());

Automating These Queries

Of course, if you’re going to audit, you’re going to want to have these run on a schedule. Preferably, you want these to run using some scheduling solution *other* than SQL Server Agent, since that’s what we’re interested in monitoring. Task Scheduler is a possibility, especially on a system with the SQL Server client tools installed that’s not one of the SQL Servers you care about. Of course, if you have a more robust job scheduling solution, you should consider that first.

Why should we avoid SQL Server Agent? If you have a SQL Server Agent job that gives some indication you’re monitoring jobs, then an attacker may cover his or her tracks. Obviously, a DBA or someone with similar rights who knows about the monitoring can work around the solution, and you’ll be looking for other ways to detect changes, but this solution is designed to catch someone who doesn’t know that this type of monitoring is in place, such as someone from the outside. After all, too many organizations don’t do any monitoring, so this is typically a safe assumption to make.

What This Doesn’t Catch

We’re merely looking at if a job has been created or changed. However, if a job step makes a call to a [stored procedure](https://www.mssqltips.com/sqlservertutorial/160/sql-server-stored-procedure-tutorial/) or calls a script at the command-line, we won’t pick up those changes. You’ll need other monitoring solutions to detect changes downstream.

**Next Steps**

* Learn how to [query job history data for your SQL Server Agent job](https://www.mssqltips.com/sqlservertip/2850/querying-sql-server-agent-job-history-data/)s.
* Find out other [information you can query with respect to Agent jobs](https://www.mssqltips.com/sqlservertip/4921/queries-to-inventory-your-sql-server-agent-jobs/).

Last Update: 2017-09-22