Inside clustering, mysterious hangs, the SA account and more

Edited by Nancy Michell

THANKS TO THE FOLLOWING MICROSOFT IT PROS FOR THEIR TECHNICAL EXPERTISE:
Ken Adamson, Sunil Agarwal, Siggi Bjarnason, Shaun Cox, Laurentiu Cristofor, Ernie DeVore, Michael Epprecht, Lucien Kleijkers, Raymond Mak, Chat Mishra (MSLI), Niraj Nagrani, Rick Salkind, Jacco Schalkwijk, Vijay Sirohi, Vijay Tandra Sistla, Matthew Stephen, and Buck Woody.

Q I need to gain a better understanding of how clustering works. Our environment will consist of 64-bit Windows Server 2003 running SQL Server 2005, a SQL Server Reporting Services (SSRS) Web farm (a report server scale-out deployment), an SSRS TempDB Catalog server and SQL Server that pumps data from a third-party database via a linked server and will store the data for SSRS.

We want a 3-node active/active/passive cluster. Node 1 would be active and would store the data pumped from the third-party database. Node 2 would be active and would store the SSRS Catalog. Node 3 would be passive and be a failover for either Node 1 or Node 2. Can you help?

A Too many people are misled by the terms active/active and active/passive in reference to SQL Server clustering. They think SQL clustering may support 'scale out' of one database or SQL instance across multiple servers. That's not the case. On SQL Server, there is no such thing as an active/active database or instance. An 'instance' is an installation of SQL Server with corresponding databases. Our clustering per SQL Server instance is active (1) to passive (*n*) always (note that the value of *n* is anywhere from 1 to 7 depending on your version of SQL Server). That's why it's called failover clustering.

Once that is understood, people can start to consider installing multiple instances of failover clustering on a set of nodes. For example three physical servers all using shared disks could have one instance that is active by default on Node 1 and a second instance that is active by default on Node 2, and both can fail to Node 3. The instances are completely separate; they do not share data and are not active/active. They are both active/passive and both share the same failover instance. If both instances fail to Node 3, then the challenge over time is knowing whether it will buckle under the load. By design, failover is intended to rely on equal processing power for the failover. If peak load requires two nodes for processing under normal operation, it is unlikely that Node 3 would survive under the peak load normally assigned to two nodes.

That said, given the relative cost of hardware capable of running a cluster, we understand how people would weigh the likelihood of both principal nodes failing simultaneously and forcing the entire load to one node. With that consideration in mind, they may make the business decision to assume the risk rather than purchase 100 percent failover capacity.

Fortunately, there's some good news: SQL Server 2005 offers many more options for high availability (HA), including alternatives that can fail over more quickly than a cluster and can even mean duplicate copies of data (clustering relies on a single SAN). The options include mirroring, peer-to-peer replication, and more. With those new alternatives, we have a lot more options to satisfy all sorts of needs, including some that may combine a number of HA features.

The Microsoft Cluster Configuration Validation Wizard (ClusPrep), now available at: **www.microsoft.com/uk/sqlqanda**, replaces what used to be the Hardware Certification List (HCL) testing, which could take months to validate a complete configuration to deem it "supportable" under clustering. This puts the hardware validation tool into the hands of the DBA, further driving down the cost (in money and time) of getting certified hardware in place. It may even make it possible to validate and deploy heterogeneous hardware within one cluster node set.

Q A delete procedure on one of my machines appears to be hanging after 12 hours. It's not blocked. Looking at the slowest query plan reveals a trigger running for 87,327 seconds, so I assume the procedure is hanging in that trigger. How can I see exactly which statement is hanging?

A It is quite possible that a loop within the trigger is not exiting. If you're hanging for a long time and you want to see which statement is executing, run the code in **Figure 1**. It will tell you which statement is currently running, and this should be the one that is causing your machine to hang.

Q I need to support transactional replication through a firewall. The publisher and distributor are on the outside of the firewall and the subscriber is on the inside. The

subscriber is set to listen on 1433 and these are my machine names: Publisher: PUBMACHINE, Distributor: DISTMACHINE, Subscriber: SUBMACHINE. What ports do I need to open to allow the initial snapshot and the publication push to succeed?

A If you are using a push subscription, opening the SQL Server port (1433 in your case) should be sufficient as the distribution agent will be running on the distributor machine (outside the firewall) and will likely have local access to the snapshot files generated by the snapshot agent. But if you are using a pull subscription, the distribution agent running on the subscriber machine will need to access the snapshot files through the firewall somehow. Here are the options that you can consider.

Assuming that the snapshot files are already accessible from a file share outside of the firewall, you can open up the Windows file sharing port(s) through the firewall so the distribution agent running inside the firewall can access the snapshot files located outside (though beware of the security implications that this may have for other parts of your infrastructure). Note that if you have configured a local path as the default snapshot location (SSMS default), you may need to use the /AltSnapshotFolder option of the distribution agent to override the snapshot file pickup location.

You can also configure replication to use FTP for transferring snapshot files (and you would need to open up port 21 for that).

Q I'd like to know if there are any drawbacks to disabling the SA account in SQL Server 2005 and if disabling SA adds true security value. Is there a whitepaper on this question?

A On a new SQL Server 2005 installation when Mixed Mode is not enabled, the SA account is disabled by default and a random password is generated for it. You can also disable it yourself. There isn't a whitepaper on it, but disabling and renaming logins is discussed in a best practices paper at: www.microsoft.com/uk/sqlqanda2.

If you want to protect yourself from attempts to break the SA account, you can rename it as well. Just remember, if you enable a disabled account, you should set a new password for it.

To answer your question about whether this offers true security, remember that the added security of disabling the account comes from the fact that password guessing is futile while the account is disabled. No matter how much time a hacker or virus has, a brute force attack against a locked account will not succeed. Renaming or disabling SA will break applications that are dependent on using the SA account for their connectivity. Finding and then fixing or eliminating these applications should be considered as a priority anyway. As mentioned, the account cannot be used to connect to the database until it is reenabled. Plus, because the authentication process fails earlier, a failed attempt will take a lesser toll on the attacked system.

Q One of my larger online transaction processing (OLTP) databases has a log file that is twice the size of the data file. I've tried using the following commands to get the log file down to a reasonable size, but I need to reduce it further:

```
backup database syslogs to backupfile
DBCC SHRINKFILE (syslogs_log)
```

A You should change your backup database to a backup log statement. Alternatively you could put your database into simple recovery mode and issue the shrinkfile statement. After the shrink of the log is finished, set the database to its previous recovery model and back up the database. If it's still not shrinking, check to make sure you have no open transactions (use dbcc opentran). See support.microsoft.com/kb/907511 for details.

Q If failover occurs during a scheduled SQL Server Agent job, what happens to that job after failover? Do I have to restart it manually?

A Yes, you will need to start it manually if you don't have some other process in place. If you don't want to restart jobs manually, you could write a script that updates a table upon job completion. If the value = 1, the job has been run; any other value indicates that the job was not completed and a second job will come in later and issue the start command. So while the job needs to be rerun if a failover occurs during its pass, by writing a script you can alleviate some of the worry for those middle-of-the-night jobs that must be completed before the next business day. ■

Figure 1 **Find currently executing statement**

```
-- Look at the current statement being run:
-- Put results to text (Ctrl + T)
DECLARE @Handle binary(20),
        @start int,
        @end int,
        @SPID int

SET    @SPID = spid

SELECT @Handle = sql_handle,
        @start = stmt_start,
        @end = stmt_end
FROM Master..sysProcesses(NOLOCK)
WHERE SPID = @SPID

IF NOT EXISTS (SELECT * FROM ::fn_get_sql(@Handle))PRINT 'Handle not found in cache'
ELSE
   SELECT 'Current Statement'= substring(text, (@start + 2)/2, CASE @end WHEN -1
THEN (datalength(text))
       ELSE (@end -@start + 2)/2 END)
       FROM ::fn_get_sql(@Handle)
```