General MSDTC connectivity investigation

Last updated by | Vitor Tomaz | Feb 24, 2023 at 3:30 AM PST

Contents

In order to run distributed transactions between various participants, there needs to be network connectivity between them.

All MSDTC communication runs through Remote Procedure Call (RPC). Because of this, MSDTC connectivity requires both port 135 (for the endpoint mapper) and all the ports between 1024 - 65535 for Transaction Monitor/Manager (TM) RPC server to be open. (We are working on limiting the range for the ports needed for the TM RPC server.)

Further, MSDTC works only with 15-character NetBIOS names. Because of this, we had to add a new hostname for the container in which sqlservr.exe and msdtc.exe are running.

This name is 15 characters long, always starts with **chn** and ends with 12 numbers and letters encoded from managed server id to which MSDTC belongs.

It will look like this, for example:

```
chnzv8n9dhfc0cx
```

Azure, on the other hand, works with fully qualified domain names.

To meet these two requirements, we are registering the FQDN of the MSDTC in Azure DNS. This name will look like this:

```
chnzv8n9dhfc0cx.aa2xj7eorkc1.database.windows.net
```

For a particular Managed Instance, this name can be found in CMS:

```
select container_hostname, azure_dns_record_name
from managed_server_msdtcs
where managed_server_id = '<managed server id>'
```

This name will point to the IP of the container in which msdtc.exe is running and it will be refreshed on every failover.

For MSDTC partners to resolve MI's MSDTC, they need to add the DNS suffix to their suffix search lists. For MIs in the same subscription, we do this automatically and there is no need to add anything.

Non-MI partners will have to add the suffix to their search list:

```
aa2xj7eorkc1.database.windows.net
```

In short, when investigating MSDTC connectivity, make sure that the partners can resolve each other's short DNS names and that nothing is blocking the traffic between them. Run the commands below on both sides. Make sure that the commands are trying to resolve name of the VM from MI and MI name from the VM.

Resolving DNS names

To resolve the short DNS names, have customer run below:

```
On customer's VM:
    Resolve-DnsName chnzv8n9dhfc0cx

On MI:
    Resolve-DnsName <customer vm short name>
```

If the name can't be resolved, check if the FQDNs can be resolved. Have customer run these:

```
On customer's VM:
    Resolve-DnsName chnzv8n9dhfc0cx.aa2xj7eorkc1.database.windows.net

On MI:
    Resolve-DnsName <customer vm FQDN>
```

If the FQDN can be resolved, suffixes are not correctly set up. If this is on MI side, PG will have to add the customer's suffix to MI's search list. Raise an ICM with *SQL Managed Instance: Distributed Transactions* team with relevant details collected.

If this is on customer's side, ask them to add the MI's DNS suffix to the VM they are trying to use to communicate with MSDTC. Example:

```
$list = (Get-DnsClientGlobalSetting).SuffixSearchList += "aa2xj7eorkc1.database.windows.net"
Set-DnsClientGlobalSetting -SuffixSearchList $list
```

If both the FQDN and the short name can't be resolved, check customer's DNS setup on both sides. Customer needs to have visibility to Azure DNS for MSDTC to work.

If the short name can be resolved, check if the name of the VM resolves to a public or a private IP address. For example, 20.x.x.x are are public IP addresses and MSDTC won't work with them. The address should look like 10.x.x.x.

If the short name can be resolved, VMs short name resolves to private IP and transactions still won't go through, check the network connectivity on both sides. First, check if endpoint mapper port is open:

```
On customer's VM:
    Test-NetConnection chnzv8n9dhfc0cx -port 135

On MI:
    Test-NetConnection <customer's vm shortname> -port 135
```

If port 135 is open, figure out on which port MSDTC is listening:

```
<msdtc process id> = tasklist /svc | findstr msdtc
<msdtc port> = netstat -abno | findstr <msdtc process id>

On customer's VM:
    Test-NetConnection chnzv8n9dhfc0cx -port <MI msdtc port>

On MI:
    Test-NetConnection <customer's vm short name> -port <msdtc port>
```

Note: The commands on customer's VM can be run through normal PowerShell window. The commands on MI can be executed through SqlAgent job.

If none of this helps, raise an ICM with *SQL Managed Instance: Distributed Transactions* team.

How good have you found this content?