


Azure SQL Managed Instance Link

Last updated by | Vitor Tomaz | Nov 16, 2022 at 5:43 AM PST

Contents

- [Intro to Azure SQL Managed Instance Link](#)
- [Customer should confirm network connectivity between S...](#)
- [Verify that trust is established between SQL Server and Ma...](#)
- [Verify operation parameters](#)

Intro to Azure SQL Managed Instance Link

Here is the link to operational guide provided to customers with instructions to establish MI link: [Operational Guide: Managed Instance Link](#) 

During the private preview, almost all customers had issues with initial setup of networking, namely Express Route /VPN and opening ports. Most of the incidents were related to bad configurations during link setup and therefore the database remains stuck in Copying state. If the customer reports that the link creation is stuck, we should first check the network configuration, then check if they have imported the certificates properly and finally whether the availability group and database names on the MI side matches the names on the box side.

On MI side, we are starting investigation with the following two queries:

```
MonUcsConnections
| where AppName == "d29e9b9f1d10"
| where PreciseTimeStamp >= ago(1d)
| project PreciseTimeStamp, AppName, LogicalServerName, event, endpoint_type, error_code, error_message, stream
```

Error messages in result and DISCONNECTED stream status indicates that there is either: • networking error between SqlServer and MI • Issue with certificates - they are not exchanged properly • Parameters to create link on MI side didn't match parameters on BOX

```
MonManagementOperations
| where operation_type == "DistributedAvailabilityGroupsLinkCreate"
| project PreciseTimeStamp, request_id, operation_parameters
| where PreciseTimeStamp >= datetime(2021-10-26 19:15:40.5928730)
```

Operation_parameters column will contain parameters used on MI to create link.

These 4 bold parameters should match parameters on Sql Servers side used to create AG and DAG.

```
<TargetDatabase>SanjaM_Test</TargetDatabase>
<SourceEndpoint>TCP://10.0.1.32:5022</SourceEndpoint>
<DistributedAvailabilityGroupName>SanjaM_DAG</DistributedAvailabilityGroupName>
<PrimaryAvailabilityGroupName>SanjaM_AG</PrimaryAvailabilityGroupName>
```

```
<InputParameters xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-inst
<ManagedServerName>chimera-prod-gp-05eu</ManagedServerName>
<TargetDatabase>SanjaM_Test</TargetDatabase>
<SourceEndpoint>TCP://10.0.1.32:5022</SourceEndpoint>
<DistributedAvailabilityGroupName>SanjaM_DAG</DistributedAvailabilityGroupName>
<PrimaryAvailabilityGroupName>SanjaM_AG</PrimaryAvailabilityGroupName>
<SecondaryAvailabilityGroupName>chimera-prod-gp-05eu</SecondaryAvailabilityGroupName>
<DistributedAvailabilityGroupId>a8549a9a-d114-8cc9-e2c1-fa5ce1efadd6</DistributedAvailabilityGroupId>
<SourceReplicaId>98b57dab-5c68-6e29-a288-0173d3b851f5</SourceReplicaId>
<TargetReplicaId>80c69706-10b4-2472-8c25-e98e6d2e69d5</TargetReplicaId>
</InputParameters>
```

First off all, when customer report the issue they to confirm that link is in DISCONNECTED state using query below and send us the results:

```
declare @dagName nvarchar(max)= 'SanjaM_DAG'
SELECT
    ag.[name] AS [DAG Name],
    ag.is_distributed,
    ar.replica_server_name AS [Underlying AG],
    ars.role_desc AS [Role],
    ars.connected_state_desc AS [Connected Status],
    ars.synchronization_health_desc AS [Sync Status],
    ar.endpoint_url as [Endpoint URL],
    ar.availability_mode_desc AS [Sync mode],
    ar.failover_mode_desc AS [Failover mode],
    ar.seeding_mode_desc AS [Seeding mode],
    ar.primary_role_allow_connections_desc AS [Primary allow connections],
    ar.secondary_role_allow_connections_desc AS [Secondary allow connections]
FROM sys.availability_groups AS ag
INNER JOIN sys.availability_replicas AS ar
    ON ag.group_id = ar.group_id
INNER JOIN sys.dm_hadr_availability_replica_states AS ars
    ON ar.replica_id = ars.replica_id
WHERE ag.is_distributed = 1 and ag.name = @dagName
GO
```

	DAG Name	is_distributed	Underlying AG	Role	Connected Status	Sync Status	Endpoint URL	Sync mode	Failover mode	Seeding mode	Primary allow connections	Secondary allow connections
1	SanjaM_DAG	1	SanjaM_AG	PRIMARY	CONNECTED	HEALTHY	TCP://10.0.1.32:5022	ASYNCHRONOUS_COMMIT	MANUAL	AUTOMATIC	ALL	ALL
2	SanjaM_DAG	1	chimera-prod-gp-05eu	SECONDARY	DISCONNECTED	NOT_HEALTHY	tcp://chimera-prod-gp-05eu.7a059cce123c.database....	ASYNCHRONOUS_COMMIT	MANUAL	AUTOMATIC	ALL	ALL

Customer should confirm network connectivity between SQL Server and Managed Instance:

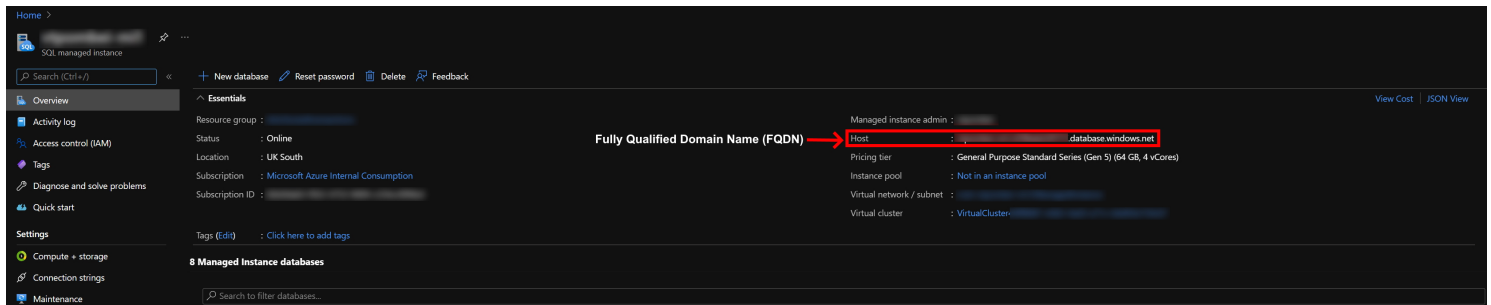
What to do on SQL Server environment

- Open outbound port 5022 on the firewall of the SQL Server environment to the entire subnet where Managed Instance is deployed
- Open inbound port 5022 to the firewall of the SQL Server environment to the entire subnet where Managed Instance is deployed It is important that port is open to the entire subnet, because if port is open only for one IP address, after MI failover this link will crash.

SQL Managed Instance environment

- Open inbound port 5022 to the IP address of the SQL Server
- Open outbound port 5022 to the IP address of the SQL Server

1. Validate that SQL Server can reach Managed Instance Validate that SQL Server can reach Managed Instance using TNC (Test Network-Connection) command. First, we need to find out the fully qualified domain name (FQDN) of your Managed Instance from the instance overview blade in Azure portal:



If there is connectivity, expected output will contain **"TcpTestSucceeded : True"**

```
PS C:\Users\cloudSA> tnc -ComputerName .database.windows.net -Port 5022

ComputerName      : .database.windows.net
RemoteAddress     : 10.0.0.254
RemotePort        : 5022
InterfaceAlias    : Ethernet 3
SourceAddress     : 10.0.1.32
TcpTestSucceeded : True
```

2. Validate that MI can reach SQL Server

To check connectivity from Managed Instance to SQL Server, on SQL Server host PowerShell run ipconfig/all to get the IP address. Example:

```
PS C:\Users\cloudSA> ipconfig /all

Windows IP Configuration
Host Name . . . . . : sql2019-prod-03
...
DNS Suffix Search List. . . . . : *.microsoft.com

Ethernet adapter Ethernet 3:
...
Link-local IPv6 Address . . . . . : fe80::****:****:de7:df2d%4(Preferred)
IPv4 Address. . . . . : 10.0.1.30(Preferred)
Subnet Mask . . . . . : 255.255.255.0
Lease Obtained. . . . . : Monday, March 29, 2021 1:52:51 PM
Lease Expires . . . . . : Saturday, May 7, 2157 1:32:55 AM
Default Gateway . . . . . : 10.0.1.1
...
NetBIOS over Tcpip. . . . . : Enabled
```

!Important! IPv4 Address. : 10.0.1.30(Preferred)

- This IP address should match IP address from MonManagementOperations <SourceEndpoint>TCP://10.0.1.30:5022</SourceEndpoint>
- Customer had issues setting up the link in the situation where he had 2 IPs on the machine, internal and external. Customer should use external address – accessible to Azure.

Once the address and the port are confirmed, on MI create SQL Agent PowerShell job with one PowerShell step:

```
tnc -ComputerName 10.0.1.30 -Port 1433.
```

In the Job History user will be able to see the outcome. If there are no issues output will say, as shown on the screen shot below, **TcpTestSucceeded : True**.

The screenshot shows the SQL Server Enterprise Manager Job History window. The left pane shows the 'Select logs' section with 'Job History' selected. The main pane displays a table of job history entries. The selected row details are shown below the table.

Date	Step ID	Server	Job Name	Step Name	Notifications	Message
3/30/2021 5:05:27 PM	1	CHIMERA-PROD...	Test network connectivity between MI and SQL Server	tnc sql se...		Executed as u...
3/30/2021 4:53:55 PM		CHIMERA-PROD...	Test network connectivity between MI and SQL Server			The job succe...
3/30/2021 4:52:21 PM		CHIMERA-PROD...	Test network connectivity between MI and SQL Server			The job succe...
3/30/2021 4:48:27 PM		CHIMERA-PROD...	Test network connectivity between MI and SQL Server			The job succe...

Selected row details:

Date: 3/30/2021 5:05:27 PM
 Log: Job History (Test network connectivity between MI and SQL Server)

Step ID: 1
 Server: CHIMERA-PROD-GP-03EU.7A059CCE123C.DATABASE.WINDOWS.NET
 Job Name: Test network connectivity between MI and SQL Server
 Step Name: tnc sql server
 Duration: 00:00:23
 Sql Severity: 0
 Sql Message ID: 0
 Operator Emailed:
 Operator Net sent:
 Operator Paged:
 Retries Attempted: 0

Message: Executed as user: DB4C1\WFbzYEdQGKcE6JG. ComputerName : 10.0.1.30 RemoteAddress : 10.0.1.30 RemotePort : 1433 InterfaceAlias : SourceAddress :
TcpTestSucceeded: True Process Exit Code 0. The step succeeded.

In case that TCP didn't succeed customer should follow [onboarding documentation](#) to enable connectivity properly.

Verify that trust is established between SQL Server and Managed Instance

This is the process that needs to be completed first to secure database mirroring endpoints for both entities and establish trust between them:

- Generate certificate on SQL Server, obtain the certificate public key,
- Obtain certificate public key on Managed Instance,
- Exchange the public key of the SQL Server certificate with Managed Instance
- Exchange the public key of the Managed Instance certificate with SQL Server

If you find this error in error log on either side of the link there is an issue with certificates. **"Logon Database Mirroring login attempt failed with error: 'Connection handshake failed. The certificate used by the peer is invalid due to the following reason: Certificate not found. State 89'"**

```
MonSQLSystemHealth
| where AppName == "d29e9b9f1d10"
| where PreciseTimeStamp >= ago(1d)
| where message contains "Connection handshake failed"
| project PreciseTimeStamp, message
```

Look for this message in the errorlog on the Managed Instance side:

```

MonSQLSystemHealth
| where AppName == "d29e9b9f1d10"
| where PreciseTimeStamp >= ago(1d)
| where message contains "SetupTenantCertificates Hybrid certificate is not imported. Operation finished with
| project PreciseTimeStamp, message

```

If you see this error that means that during the startup of the instance, hybrid certificates have not been imported. Mitigation:

- 1st possibility is that certificates have been dropped by the customer from the MS in the meantime. If this is the case, the customer should import the certificate back to the MS and it will automatically be propagated to the engine.
- 2nd scenario, if certificate still exist in the MS the mitigation would be to manually import the desired certificate to the PHYSMASTER of the instance. In this case create incident and assign to: **Azure SQL DB SQL Managed Instance Chimera**

If this error is not present, we should retrieve input parameters from MonManagementOperations table:

```

MonManagementOperations
| where PreciseTimeStamp >= ago(1d)
| where operation_type == "UploadManagedInstanceServerTrustCertificate"
| project PreciseTimeStamp, request_id, operation_parameters, operation_result

```

In case that operation exists, column operation_results will contain PublicBlob and thumbprint of BOX certificate:

```

<OutputParameters xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-ins
<PublicBlob>308202D2308201BAA00302010202101B6F8A6DF6F029A84EA98478533A5A21300D06092A864886F70D01010B0500302531
<Thumbprint>8DE51A30C01C8356F001BEDA78AC129A2A11F928</Thumbprint>
<IsExistingEntity>false</IsExistingEntity></OutputParameters>

```

Customer should confirm that thumbprint and PublicBlob matches certificate he used to create DATABASE MIRRORING endpoint:

```

CREATE ENDPOINT database_mirroring_endpoint
STATE=STARTED
AS TCP (LISTENER_PORT=5022, LISTENER_IP = ALL)
FOR DATABASE_MIRRORING (
ROLE=ALL,
AUTHENTICATION = CERTIFICATE BOX_CERT,
ENCRYPTION = REQUIRED ALGORITHM AES)
GO

```

This will return public blob of BOX_CERT:

```

DECLARE @PUBLICKEYENC VARBINARY(MAX);
SELECT @PUBLICKEYENC = CERTENCODED(CERT_ID('BOX_CERT'));
SELECT @PUBLICKEYENC AS PublicKeyEncoded;

```

PublicKeyEncoded

0x308202D2308201BAA00302010202101B6F8A6DF6F029A84...

```
select name, thumbprint from sys.certificates where name = 'BOX_CERT'
```

name

thumbprint

BOX_CERT

0xA59FA84C9ADE5DD0A245BCB5F88AEE96F695268D

If operation to import BOX certificate was successful and customer confirm that PublicBlob and thumbprint contains good value, we should check MonUcsConnections: **An error occurred while receiving data: '10054(An existing connection was forcibly closed by the remote host.)** This error indicates that MI certificate is not present on SqlServer side.

In this case, customer should check errorlog on SqlServer side and if this error exists: **"Logon Database Mirroring login attempt failed with error: 'Connection handshake failed. The certificate used by the peer is invalid due to the following reason: Certificate not found. State 89"**

They should obtain the certificate public key on Managed Instance. For SQL Managed Instance, an integrated stored procedure can be used to obtain its certificate public key. Log in to Managed Instance through a client such is SSMS , and execute the following stored procedure on Managed Instance:

```
exec sp_get_endpoint_certificate 4
```



Use this value to create certificate on SqlServer side:

```
CREATE CERTIFICATE MI_PUBLIC_CERT
FROM BINARY = 0x308203AE30820296A00302010202101F6DAA87BF04138E406C
GO
```

Verify operation parameters

```
<TargetDatabase>SanjaM_Test</TargetDatabase> <SourceEndpoint>TCP://10.0.1.32:5022</SourceEndpoint>
<DistributedAvailabilityGroupName>SanjaM_DAG</DistributedAvailabilityGroupName>
<PrimaryAvailabilityGroupName>SanjaM_AG</PrimaryAvailabilityGroupName>
```

BOX side:

These 4 parameters which are used on MI side in powershell command should match to the relevant one on SQL side:

```

CREATE DATABASE SanjaM_Test
GO
ALTER DATABASE SanjaM_Test SET RECOVERY FULL
GO
BACKUP DATABASE SanjaM_Test TO DISK = N'G:\log\backup.bak'
GO

CREATE AVAILABILITY GROUP SanjaM_AG --<PrimaryAvailabilityGroupName>
WITH (CLUSTER_TYPE = NONE)
FOR DATABASE SanjaM_Test
REPLICA ON
'sql2019-prod-05' WITH
(
    ENDPOINT_URL = 'TCP://10.0.1.32:5022',    --<SourceEndpoint>
    AVAILABILITY_MODE = SYNCHRONOUS_COMMIT,
    FAILOVER_MODE = MANUAL,
    SEEDING_MODE = AUTOMATIC
);
GO

CREATE AVAILABILITY GROUP SanjaM_DAG
WITH (DISTRIBUTED)
AVAILABILITY GROUP ON
'SanjaM_AG' WITH    --<PrimaryAvailabilityGroupName>
(
    LISTENER_URL = 'TCP://10.0.1.32:5022',    --<SourceEndpoint>
    AVAILABILITY_MODE = ASYNCHRONOUS_COMMIT,
    FAILOVER_MODE = MANUAL,
    SEEDING_MODE = AUTOMATIC,
    SESSION_TIMEOUT = 20
),
'chimera-prod-gp-05eu' WITH    -- This should be same as <SecondaryAvailabilityGroupName>chimera-prod-g
(
    LISTENER_URL = 'tcp://chimera-prod-gp-05eu.7a059cce123c.database.windows.net:5022;Server=[chimera-pro
    AVAILABILITY_MODE = ASYNCHRONOUS_COMMIT,
    FAILOVER_MODE = MANUAL,
    SEEDING_MODE = AUTOMATIC
);
GO

```

chimera-prod-gp-05eu.7a059cce123c.database.windows.net - FQDN of provided Managed Instance

In case that customer provided bad IP address, this error will be present in log on MI:

```

MonSQLSystemHealth
| where AppName == "d29e9b9f1d10"
| where PreciseTimeStamp >= datetime(2021-10-26 19:38:02.6302051) and PreciseTimeStamp <= datetime(2021-10-26
| where message contains "networking or firewall issue exists"
| project PreciseTimeStamp, message

```

A connection timeout has occurred while attempting to establish a connection to availability replica '98B57DAB-5C68-6E29-A288-0173D3B851F5' with id [98B57DAB-5C68-6E29-A288-0173D3B851F5]. Either a networking or firewall issue exists, or the endpoint address provided for the replica is not the database mirroring endpoint of the host server instance.

How good have you found this content?

