

# Other error not listed

Last updated by | Vitor Tomaz | Nov 24, 2021 at 3:06 AM PST

## Contents

- [See Other error not listed under SQL DB as well for more T...](#)
- [Self-help content presented in Azure Portal](#)
  - [Run Azure SQL Connectivity Checker](#)
  - [Login failed due to client TLS version being less than mini...](#)
    - [Target Principal Name is incorrect](#)
    - [Connection timeout expired](#)
  - [TCP error code 10060](#)
  - [via public endpoint](#)
    - [Resources](#)

See [Other error not listed](#) under SQL DB as well for more TSGs

## Self-help content presented in Azure Portal

(This content was shown to the customer during case submission. It's also visible on 'Diagnose and solve problems' blade.)


Connection issues can be caused by various reasons, such as reconfiguration, firewall settings, connection timeouts, login failures, resource limits, and network issues. Another potential cause is not applying practices and guidelines during client application design.

To resolve these issues, use insights from the following diagnostics, run Azure SQL Connectivity Checker, and review the issues in the following section headings to find the solution that meets your needs.

## Run Azure SQL Connectivity Checker

The Azure SQL Connectivity Checker is a PowerShell script that automates a series of checks for the most common configuration issues. Most issues detected include recommendations on how to resolve the issue.

Run the PowerShell script from Windows client computers where the error is occurring.

*To run the tests from Linux, from machines without internet access, or from a containerized environment, see [SQL Connectivity Checker on GitHub](#) .*

1. Open Windows PowerShell ISE (in **Administrator mode** if possible).

**Note:** To collect a network trace along with the tests ( `CollectNetworkTrace` parameter), you must run PowerShell as an administrator.

2. Open a **New Script** window.

3. Paste the following in the script window:

```

$parameters = @{
    # Supports Single, Elastic Pools and Managed Instance (please provide FQDN, MI public endpoint is supp
    # Supports Azure Synapse / Azure SQL Data Warehouse (*.sql.azuresynapse.net / *.database.windows.net)
    # Supports Public Cloud (*.database.windows.net), Azure China (*.database.chinacloudapi.cn), Azure Germ
    Server = '*.database.windows.net' # or any other supported FQDN
    Database = '' # Set the name of the database you wish to test, 'master' will be used by default if not
    User = '' # Set the login username you wish to use, 'AzSQLConnCheckerUser' will be used by default if
    Password = '' # Set the login password you wish to use, 'AzSQLConnCheckerPassword' will be used by def

    ## Optional parameters (default values will be used if omitted)
    SendAnonymousUsageData = $true # Set as $true (default) or $false
    RunAdvancedConnectivityPolicyTests = $true # Set as $true (default) or $false, this will load the libr
    ConnectionAttempts = 1 # Number of connection attempts while running advanced connectivity tests
    DelayBetweenConnections = 1 # Number of seconds to wait between connection attempts while running advan
    CollectNetworkTrace = $true # Set as $true (default) or $false
    #EncryptionProtocol = '' # Supported values: 'Tls 1.0', 'Tls 1.1', 'Tls 1.2'; Without this parameter op
}

$ProgressPreference = "SilentlyContinue";
if ("AzureKudu" -eq $env:DOTNET_CLI_TELEMETRY_PROFILE) {
    $scriptFile = '/ReducedSQLConnectivityChecker.ps1'
} else {
    $scriptFile = '/AzureSQLConnectivityChecker.ps1'
}
$scriptUrlBase = 'http://raw.githubusercontent.com/Azure/SQL-Connectivity-Checker/master'
cls
Write-Host 'Trying to download the script file from GitHub (https://github.com/Azure/SQL-Connectivity-Check
try {
    [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12 -bor [Net.SecurityProtoc
    Invoke-Command -ScriptBlock ([Scriptblock]::Create((Invoke-WebRequest ($scriptUrlBase + $scriptFile) -U
    })
} catch {
    Write-Host 'ERROR: The script file could not be downloaded:' -ForegroundColor Red
    $_.Exception
    Write-Host 'Confirm this machine can access https://github.com/Azure/SQL-Connectivity-Checker/' -Foregr
    Write-Host 'or use a machine with Internet access to see how to run this from machines without Internet
}
#end

```

4. Set the parameters on the script. You must set the server name and database name. User and password are optional, but are best practices.
5. Run the script.

Results are displayed in the output window. If the user has permissions to create folders, a folder with the resulting log file will be created, along with a ZIP file ( AllFiles.zip ). When running on Windows, the folder opens automatically after the script completes.

6. Examine the output for any issues detected, and recommended steps to resolve the issue.

If the issue can't be resolved, send AllFiles.zip using the **File upload** option in the **Details** step of creating your support case.

## Login failed due to client TLS version being less than minimal TLS version allowed by the server

The minimal TLS version is enforced by the managed instance for inbound connections. See [configure minimal TLS version in Azure SQL Managed Instance](#) for commands to set the TLS version, or if you have further questions or concerns. Check the client application setting for TLS requirement. If the client application is not set to use any encryption and if the TLS setting on SQL Managed Instance is set to minimum, you might encounter TLS Login errors.

## Target Principal Name is incorrect

The error, "Target Principal Name is incorrect," can occur in the following scenarios:

### Using local host file:

The Azure SQL Managed Instance host name is resolved from the host's file and connection is established successfully.

However, if the **Encrypt connection** option is selected and **Trust server certificate** is not selected, the client tries to validate the server certificate and the validation will fail because the record in the host's file doesn't match the principal name in server certificate.


To connect successfully using the host's file and the **Encrypt connection** option, you need to select **Trust server certificate** in order to avoid certificate validation.

### Using [custom DNS](#) zone instead of the default, for example, .contoso.com:

- Use `cliConfig` to define an alias. The tool is just a registry settings wrapper, so it can be done using group policy or a script, as well.
- Use `CNAME` with the `TrustServerCertificate=true` option. This is set on the connection string. However, we don't recommend this option. With `TrustServerCertificate` enabled, the server certificate is not validated by the client application. This exposes you to cybersecurity attacks.
- The alias must be created with the same name as the Azure SQL Managed Instance `<instance name>`. If the alias doesn't match with instance name, you may get a Target Principal error.

### Trying to connect using the IP address instead of the FQDN of your server:

Connecting to a managed instance using an IP address is not supported. A Managed Instance's host name maps to the load balancer in front of the Managed Instance's virtual cluster. Because one virtual cluster can host multiple Managed Instances, a connection can't be routed to the proper Managed Instance without specifying its name.

For more information on SQL Managed Instance virtual cluster architecture, see [Virtual cluster connectivity architecture](#) .

## Connection timeout expired

The connection timeout can expire when connecting from an Azure VM to Azure SQL Managed Instance when there's no connectivity established between the virtual networks. Connecting an application when it resides within a different virtual network from SQL Managed Instance is more complex, because SQL Managed Instance has private IP addresses in its own virtual network. To connect, an application needs access to the virtual network where SQL Managed Instance is deployed. So you need to make a connection between the application and the SQL Managed Instance virtual network. The virtual networks don't have to be in the same subscription in order for this scenario to work.

There are two options for connecting virtual networks:

- [Azure VNet peering](#) 
- VNet-to-VNet VPN gateway using ([Azure portal](#) , [PowerShell](#) , or [Azure CLI](#) )

For more information, see [Connect application instances](#) .

## TCP error code 10060


Error code 10060 typically means that the client can't connect to the server. This usually indicates a client-side networking issue that you'll need to pursue with your local network administrator.

### Connecting via VPN (private endpoint)

SQL Managed Instance is placed inside the Azure virtual network and a subnet that is dedicated to managed instances. This deployment provides you with a secure private IP address. You can connect to SQL Managed Instance via private endpoint if you are connecting from one of the following:


- Machine inside the same virtual network
- Machine in a peered virtual network
- Machine that is network-connected by VPN or Azure ExpressRoute

If you're trying to connect via private endpoint, review the following:




- Your machine has virtual network access to managed instance
- The host name is valid, and the format is <mi\_name>.<dns\_zone>.database.windows.net
- The port used for the connection is 1433 and you have firewalls and Network Security Groups (NSG) open to allow access on ports 1433
- If the [connection type](#)  is Redirect, and firewalls and Network Security Groups (NSG) are open to allow access on ports 11000-11999

Learn more about how to [connect your application to Azure SQL Managed Instance](#) .

### via public endpoint



If the machine doesn't have virtual network access to managed instance, you can use [public endpoint](#)  for data access to your managed instance from outside the virtual network. For example, this allows access from multitenant Azure services like Power BI, Azure App Service, or an on-premises network, via a public endpoint.

If you are trying to connect via public endpoint, review the following:

- You have [public endpoint enabled](#) 
- The [host name is valid](#) , and the format is <mi\_name>.public.<dns\_zone>.database.windows.net,3342
- The port used for the connection is 3342, as mentioned earlier: [Inbound security rule was added in the NSG for port 3342](#) 

To learn more, see [how to configure a public endpoint in Azure SQL Managed Instance](#) .

### Resources

- [Troubleshoot connectivity issues and other errors](#) 
- [Troubleshoot transient connection errors](#) 

### How good have you found this content?



-