

Connect via Public Endpoint

Last updated by | Vitor Tomaz | Jun 8, 2022 at 5:36 AM PDT

Contents

- Self-help content presented in Azure Portal
 - Learn how to configure a public endpoint on your managed...
 - Configuring a public endpoint
 - Connect from Power BI via public endpoint
 - Connect from Azure Data Factory via public endpoint
 - Connect from Azure App Services via public endpoint
 - Test connectivity using Azure SQL Connectivity Checker
 - Resources

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.)

Learn how to configure a public endpoint on your managed instance

A public endpoint for a managed instance enables data access to your managed instance from outside the virtual network. You're able to access your managed instance from multi-tenant Azure services like Power BI, Azure Data Factory, Azure DevOps, Azure App Service, or an on-premises network. By using the public endpoint on a managed instance, you don't need to use a VPN.

Use the following guidance to configure a public endpoint; connect via a public endpoint from Power BI, Azure Data Factory, or Azure App Services; test connectivity using the Azure SQL Connectivity Checker; and more.

Configuring a public endpoint

Configuring a public endpoint is a multi-step process. You'll need to:

- Enable the public endpoint for your managed instance. You can use the [Azure portal](#) or [PowerShell](#).
- [Allow public endpoint traffic](#) on the network security group.
- Use the managed instance [public endpoint connection string](#).

See detailed information at [Configure public endpoint in Azure SQL Managed Instance](#).



Connect from Power BI via public endpoint

In addition to using a public endpoint to connect from Power BI, you can use Azure service tags with Power BI to enable an Azure SQL Managed Instance (MI) to allow incoming connections from the Power BI service.


In Azure, a service tag is a defined group of IP addresses you can configure to be automatically managed, as a group, to minimize the complexity of updates or changes to network security rules. By using service tags with Power BI, you can enable a SQL Managed Instance to allow incoming connections from the Power BI service.

See [Using service tags with Power BI](#) .

Connect from Azure Data Factory via public endpoint


To access the SQL Managed Instance [public endpoint](#) , you can use an Azure Data Factory–managed Azure [integration runtime](#) .

Make sure that you enable the public endpoint and you allow public endpoint traffic on the network security group so Azure Data Factory can connect to your database.

Note that the *DataFactory* [service tag](#)  is not currently supported by Data flows and Azure Integration Runtime, which enable Managed Virtual Network. You can still use the *DataFactory* service tag for data movement, pipelines, and external activity executions.


Connect from Azure App Services via public endpoint


You can limit access to the managed instance public endpoint by limiting access to a list of IP addresses.

Any outbound connection from the [App Service](#)  app, such as to back-end databases, uses one of the outbound IP addresses as the origin IP address.

Regardless of the number of scaled-out instances, each app has a set number of outbound IP addresses at any given time.

The IP address to use is selected randomly at runtime, so your back-end service must open its firewall to all the outbound IP addresses for your app.

To find *all* possible [outbound IP addresses](#)  for your app, regardless of pricing tiers, select **Properties** in your app's left-hand navigation. They are listed in the **Additional Outbound IP Addresses** field.


[App Service Environments](#)  use dedicated network infrastructures, so apps running in an App Service environment get static, dedicated IP addresses for both inbound and outbound connections.

Use the list of IP addresses in the network security group to limit access to the managed instance public endpoint on port 3342.

Test connectivity using Azure SQL Connectivity Checker

The Azure SQL Connectivity Checker tool is a PowerShell script, run from the client machine, that automates a series of checks for the most common configuration problems. Most issues it detects will come with recommendations for how to resolve them.

Run the following PowerShell script from the 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 (provide FQDN, MI public endpoint is supported)
    # Supports Azure Synapse / Azure SQL Data Warehouse (*.sql.azure-synapse.net / *.database.windows.net)
    # Supports Public Cloud (*.database.windows.net), Azure China (*.database.chinacloudapi.cn), Azure Ge
    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 n
    User = '' # Set the login username you wish to use, 'AzSQLConnCheckerUser' will be used by default i
    Password = '' # Set the login password you wish to use, 'AzSQLConnCheckerPassword' will be used by d

    ## 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 li
    ConnectionAttempts = 1 # Number of connection attempts while running advanced connectivity tests
    DelayBetweenConnections = 1 # Number of seconds to wait between connection attempts while running adv
    CollectNetworkTrace = $true # Set as $true (default) or $false
    #EncryptionProtocol = '' # Supported values: 'Tls 1.0', 'Tls 1.1', 'Tls 1.2'; Without this parameter
}

$ProgressPreference = "SilentlyContinue";
if ("AzureKudu" -eq $env:DOTNET_CLI_TELEMETRY_PROFILE) {
    $scriptFile = '/ReducedSQLConnectivityChecker.ps1'
} else {
    $scriptFile = '/AzureSQLConnectivityChecker.ps1'
}
$scriptUrlBase = 'http://raw.githubusercontent.com/Azure/SQL-Connectivity-Checker/master'
cls
Write-Host 'Trying to download the script file from GitHub (https://github.com/Azure/SQL-Connectivity-Che
try {
    [Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12 -bor [Net.SecurityPro
    Invoke-Command -ScriptBlock ([Scriptblock]::Create((Invoke-WebRequest ($scriptUrlBase + $scriptFile)
    })
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/' -Fore
    Write-Host 'or use a machine with Internet access to see how to run this from machines without Intern
}
#end
```

4. Set the parameters on the script. You must set the server name and database name. Setting the user and password is best practice, but optional.
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 is 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 any recommended steps to resolve the issue.
7. If the issue can't be resolved, send `AllFiles.zip` using the **File upload** option in the **Details** step of creating your support case.

Resources

- [Connectivity architecture for Azure SQL Managed Instance](#) 
- [Connect your application to Azure SQL Managed Instance](#) 
- [Configure connection types](#) 

How good have you found this content?

