

# Connectivity - Using a TCP proxy to connect to SQL Database over VPN

Last updated by | Keith Elmore | Aug 5, 2020 at 1:39 PM PDT

## Contents

- [Scenario](#)
- [Challenge](#)
- [Workaround](#)
  - [Public Doc Reference](#)

## Scenario

Connect to SQL Database from on-premises by allowing outbound traffic to well-known IP addresses for Azure SQL Database gateway for your Azure region that are documented [here](#).

## Challenge

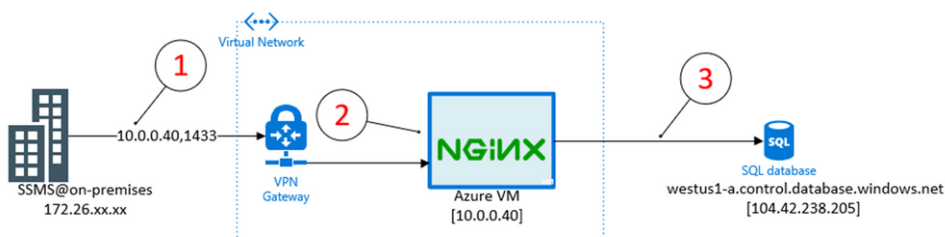
However, this traffic goes over the Internet which introduces with it some inherent security and reliability challenges. A common example is having to manage firewall rules for mobile clients whose SNAT addresses change frequently.

## Workaround

An alternative is to setup a private connection to Azure – via P2S VPN, S2S VPN or Express Route – and then use a TCP proxy server to forward traffic to public IP address for SQL Database. In this blog, we provide a proof-of-concept of how this can be achieved using P2S VPN and NGINX server.

We used P2S VPN as easy way to get traffic to flow from on-premises to Azure. In real-world scenarios customers should rely on more robust peering solutions like S2S VPN or Express Route for their production workloads.

Here is the high-level architecture diagram of how this solution works in practice



Step1 - User connects from on-premises (over VPN) by specifying Private IP address for Azure VM & port 1433. *Alternately, hostname can be used with custom DNS that then maps it to Private IP address.*

Step 2 - NGINX is running on Azure VM and listening for traffic on port 1433

Step 3 - Traffic is forwarded by NGINX to the Sql Database Gateway (for the region hosting your SQL Database server) as part of normal login flow

Here is a step by step guide to implementing this architecture

1. Start by adding Azure VM vnet&subnet to virtual network rule on SQL Database as described [here](#)

This will ensure that the subnet hosting Azure VM can communicate to Sql Database. The result should be a firewall rule that looks like this

Virtual networks		+ Add existing virtual network	+ Create new virtual network		
RULE NAME	VIRTUAL NETWORK	SUBNET	ADDRESS RANGE	ENDPOINT STATUS	
<...> rnVmSubnetRule	rmdatavnet	vmsubnet	10.0.0.32/27	Enabled	

2. Next, assign a Private IP address to your via VM Blade à Networking à IP Configurations. For this demo I typed in 10.0.0.40

ipconfig1  
rmdatasql1795

Save Discard

Public IP address settings

Public IP address

Disabled Enabled

\* IP address

rmdatasql17-ip (Unassigned)

Private IP address settings

Virtual network/subnet

rmdatavnet/vmsubnet

Assignment

Dynamic Static

\* IP address

10.0.0.40

3. (Optional) Configure and install P2S VPN client per instructions given [here](#) and connect to it.

4. Download Nginx and change only the body the nginx.conf file **as follows** without modifying headers

```
#user nobody;

worker_processes 1;

#error_log logs/error.log;

#error_log logs/error.log notice;

#error_log logs/error.log info;

#pid logs/nginx.pid;

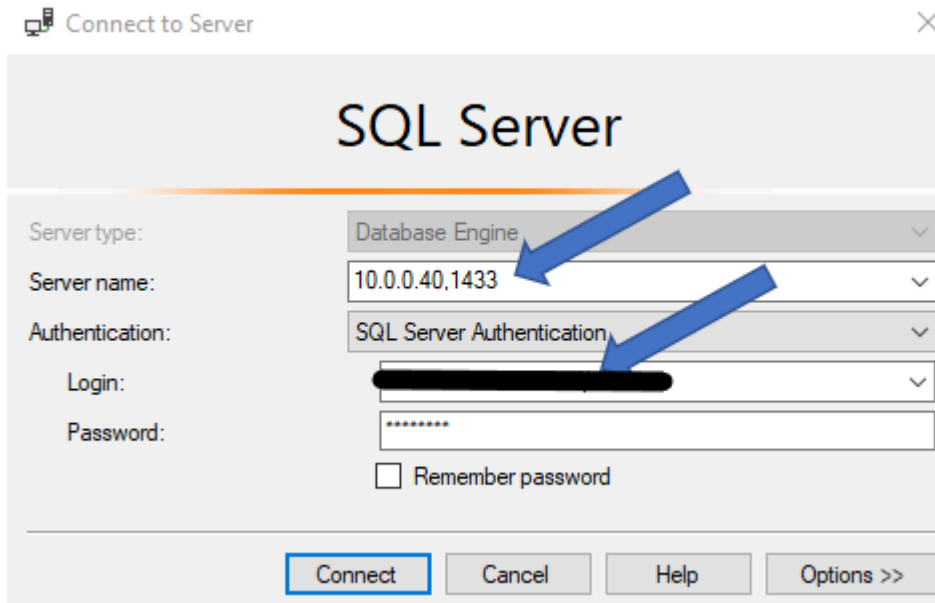
events {
    worker_connections 1024;
}

stream {
    upstream sqlvm {
        server rnsqldbsrv.database.windows.net:1433;
    }
    server {
        listen 1433;
        proxy_pass sqlvm;
    }
}
```

This tells NGINX to forward any traffic received on port 1433 to my SQL Database server rnsqldbsrv.database.windows.net:1433 which is located in West US datacenter ( as confirmed by a simple ping command shown below)

```
>ping rnsqldbsrv.database.windows.net
Pinging westus1-a.control.database.windows.net [104.42.238.205] with 32 bytes of data:
```

5. Now connect via SSMS specifying the Private IP address of VM and port 1433( which Nginx is listening on)



We used **username@servername** for the traffic to flow correctly between on-premises and SQL Database. Connection policy of SQL Database needs to be explicitly set to Proxy mode (as outlined [here](#)) for this to work without specifying the servername.

Additionally, this method only works for SQL Authentication. Custom DNS is needed for this to work with AAD Auth.

To recap, we showed you a proof-of-concept for how you can connect to Sql Database from on-premises by using a TCP proxy server to forward traffic.

## Public Doc Reference

<https://techcommunity.microsoft.com/t5/Azure-SQL-Database/Using-a-TCP-proxy-to-connect-to-SQL-Database-over-VPN/ba-p/390962>

How good have you found this content?

