SQL Server Replication Setup

Please outline the steps for setting up Microsoft SQL Server Transaction Replication between a Azure Managed SQL Server as a subscriber and a Microsoft SQL Server installation on an Azure Virtual Machine in the same Azure tenant. Assume both servers are on the same domain.

Below is a high-level overview of the steps required to set up Transactional Replication between a SQL Server instance on an Azure Virtual Machine (acting as Publisher/Distributor) and an Azure SQL Managed Instance (acting as Subscriber). These instructions assume:

- The Azure VM and the Azure SQL Managed Instance (MI) are in the same domain (or at least can communicate via internal networking).
- You have network connectivity and permissions in place to configure replication.
- You have administrative privileges on both SQL Server instances.

> **Important Note on Terminology:**
> - **Publisher**: The source instance that owns the database to be replicated.
> - **Distributor**: A server that hosts the distribution database and stores metadata/history. It can be on the same server as the Publisher or on a separate instance.
> - **Subscriber**: The target instance where replicated data is received.

---

1. Prerequisites and Connectivity

1. **Ensure Network Connectivity**
   - Both SQL Server on the Azure VM and the Azure SQL Managed Instance must be able to “see” each other.
   - Verify that all necessary ports (typically TCP 1433 for SQL Server, plus any ports for replication agents if configured differently) are open in the Network Security Groups (NSGs), Azure Firewall, and local server firewalls.
2. **DNS / Hostname Resolution**

- On the VM, ensure that the Managed Instance's FQDN can be resolved.
- If necessary, add host entries on the VM or configure DNS so that the MI's hostname resolves internally.

3. **Domain / Authentication**
   - In many Transaction Replication setups, SQL Authentication is used between Publisher/Distributor and Subscriber. If you plan to use Windows Authentication (and both are truly in the same domain), ensure that the domain user(s) running the replication agents have the appropriate permissions.
   - Alternatively, set up SQL logins and provide them in the replication security configuration.
4. **Version Compatibility**
   - Azure SQL Managed Instance supports being a Subscriber to Transaction Replication from an on-premises or Azure VM SQL instance.
   - Confirm that your SQL Server VM version is supported for replication with Managed Instance. Generally, you'll want SQL Server 2016 SP2 or later if possible (though older versions can often work with certain constraints).

---

2. Configure the Distributor (on the Azure VM)

1. **Open SQL Server Management Studio (SSMS)** on the Azure VM that will host the distribution database.
2. **Right-click on 'Replication'** in Object Explorer → **'Configure Distribution...'**.
3. **Specify the Distributor**:
   - If you want the Publisher to act as its own Distributor (most common in simpler setups), select **'Use the following server as its Distributor'** and choose the local server.
   - Otherwise, if you have a dedicated Distributor on another server, select that server.
4. **Specify Snapshot Folder**:
   - Enter a UNC path that both the Publisher and Subscriber can access, such as `\\YourVMName\ReplicationSnapshots`.

- Ensure that the Windows account or SQL login used by replication has read/write access to that folder.

5. **Complete the Wizard:**
   - The wizard will create the distribution database on the specified path and configure the necessary jobs.
   - Make note of the distribution database name (defaults to `distribution`).

---

3. Configure the Publication (on the Azure VM)

1. **Right-click on 'Replication'** → **'New Publication...'** in SSMS.
2. **Select the Database** you want to publish from (on the Azure VM).
3. **Choose Publication Type:**
   - Select **'Transactional Publication'**.
4. **Select Articles**:
   - Pick the tables, stored procedures, or other objects you wish to replicate.
   - Ensure primary keys exist on the tables you plan to replicate (a requirement for transactional replication).
5. **Article Properties (Optional)**:
   - If needed, specify filters or change how certain columns are replicated.
6. **Snapshot Agent Settings**:
   - If you want the initial snapshot to be applied immediately, ensure the schedule is set accordingly.
   - By default, it's set to run on demand the first time the subscription is set up.
7. **Security Settings**:
   - Specify the account under which the Snapshot Agent and Log Reader Agent run. Typically, a domain account or a SQL login with db_owner on the publication database and appropriate role memberships in the distribution database.
8. **Name the Publication** and complete the wizard.

4. Prepare Azure SQL Managed Instance for Subscription

1. **Connect to the Azure SQL Managed Instance** in SSMS.
2. **Confirm Database Exists**:
   - You can replicate to an existing database or let replication create it. Typically, you create an empty database with the appropriate settings (collation, max size, etc.) that matches or is compatible with the source.
3. **Set Database to Proper Recovery Model**:
   - Transaction replication doesn't strictly require a particular recovery model on the subscriber, but Full or Simple are both common.
   - Make sure the Managed Instance database is ready for data to be applied.
4. **Confirm Replication Support**:
   - Azure SQL Managed Instance supports being a subscriber to transactional replication. Just ensure you're on a recent-enough version of SQL Managed Instance and that you have the required permissions.
   - Typically, you need `db_owner` on the subscription database or the ability to create a new database.

---

5. Create the Subscription (on the Azure VM pointing to MI)

1. **Back on the Azure VM's SSMS**:
   - Right-click on **'Local Publications'** → **'New Subscriptions...'**.
2. **Select the Publication** you created.
3. **Choose Subscriber**:
   - In the 'Subscriber' drop-down, select **'Add SQL Server Subscriber...'**.
   - Enter the **Managed Instance name** (e.g., `yourmi.public.123456.database.windows.net`) or the domain-style name if appropriate.

- If SSMS doesn't detect the MI automatically, you may need to type in the server name manually.

4. **Subscription Database**:
   - Select an existing database on the MI (the one created in step 4.2) or specify a new one.
5. **Distribution Agent Security**:
   - Specify **how the Distribution Agent will connect** to the Subscriber.
   - Typically, you'll use SQL Authentication on Azure SQL Managed Instance with a login that has the correct privileges.
   - Alternatively, if domain authentication is configured (and the MI is domain-joined in a special scenario), you could use Windows Authentication, but this is less common in typical Azure setups.
6. **Agent Schedule**:
   - Set the Distribution Agent schedule. By default, it might run continuously to push transactions to the Subscriber in near real-time.
7. **Review and Finish** the wizard. This creates the subscription, starts the Snapshot Agent (if not already run), and then applies the initial snapshot to the subscriber.

---

6. Validate Replication

1. **Check Replication Monitor**:
   - In SSMS on the VM, right-click **'Replication'** → **'Launch Replication Monitor'**.
   - Verify that the snapshot is created and applied successfully.
   - Look for any errors under the Publisher, Distributor, or Subscriber nodes.
2. **Verify Data**:
   - On the Managed Instance database, query one of the replicated tables. You should see data that matches the Publisher's table.
3. **Confirm Transaction Flow**:
   - Make an update/insert in the published table on the Publisher.

- Verify that the same change appears at the Subscriber within the expected timeframe.

---

7. Ongoing Maintenance & Monitoring

1. **Monitor Agent Jobs**:
   - On the VM's SQL Server Agent, you'll see jobs for the Snapshot Agent, Log Reader Agent, and Distribution Agent.
   - Monitor and confirm they're running without errors.
2. **Retention Settings**:
   - On the Distributor, check the distribution database retention settings for transaction history.
   - Ensure cleanup jobs (Distribution Cleanup) are running properly.
3. **Schema Changes**:
   - For schema changes, either replicate them automatically (by setting publication article properties) or apply them manually to the Subscriber as needed.
4. **Scaling & Performance**:
   - Keep an eye on performance counters (log reader throughput, distribution agent throughput).
   - Consider adjusting the subscription schedules, the number of parallel agents, or snapshot frequency as usage grows.
5. **Disaster Recovery**:
   - Have a plan for DR at both the Publisher/Distributor side and the Subscriber side.
   - Document steps to rebuild or reinitialize replication if needed.

Common Pitfalls & Tips

- **Firewall and Port Issues**: Often the biggest obstacle. Verify connectivity and open ports in Azure NSGs, the server's local firewall, and any Azure Firewall or VPN configurations.
- **Security Context**: Ensure the correct permissions in the publication database, distribution database, and on the Azure SQL Managed Instance. Using a dedicated SQL login for replication on the MI side is common.
- **Snapshot Folder Permissions**: The account running the Snapshot Agent must have read/write access to the snapshot folder.
- **Publisher/Subscriber Versions**: Always confirm that your VM's SQL Server version is fully supported for replicating to Azure SQL Managed Instance. Consult Microsoft's official replication compatibility matrix for details.
- **Agent Job Ownership**: Make sure the jobs on the VM's SQL Server Agent have a valid owner (often `sa`) and that `sa` or the job owner has the required permissions.

Conclusion

By carefully following these steps—configuring the Distributor on the Azure VM, creating the Publication, preparing the Azure SQL Managed Instance as a valid Subscriber, and then creating the Subscription—you can successfully set up transactional replication in Azure. The key challenges typically involve properly configuring security and network connectivity, but once those are addressed, the replication engine works much as it does in an on-premises scenario.