High Availability Plan For Hybrid Infrastructure

Shaunak Patel

Applied Network Infrastructure and System Administration, Conestoga Collage

Table of Contents

INTRODUCTION

In response to the new directive to extend high-availability functionality to the remaining core components, this document outlines a comprehensive deployment plan aimed at developing and implementing a robust high-availability (HA) cluster. The primary objective is to achieve and maintain an uptime as close to 99.99% as possible. This initiative involves addressing critical questions that guide the strategic decisions and technical considerations throughout the deployment process.

The initial inquiry focuses on determining the optimal hardware configuration for the HA cluster. This includes identifying the types of servers, storage systems, and networking equipment that are best suited for achieving redundancy and fault tolerance. The selection of hardware components will play a pivotal role in ensuring the reliability and performance of the entire infrastructure.

A crucial decision in the deployment plan involves the choice between an on-premises, offsite, or cloud-based HA cluster. Each option comes with its own set of advantages and considerations. Evaluating factors such as scalability, flexibility, cost implications, and control will help determine the most suitable environment for housing the high-availability cluster.

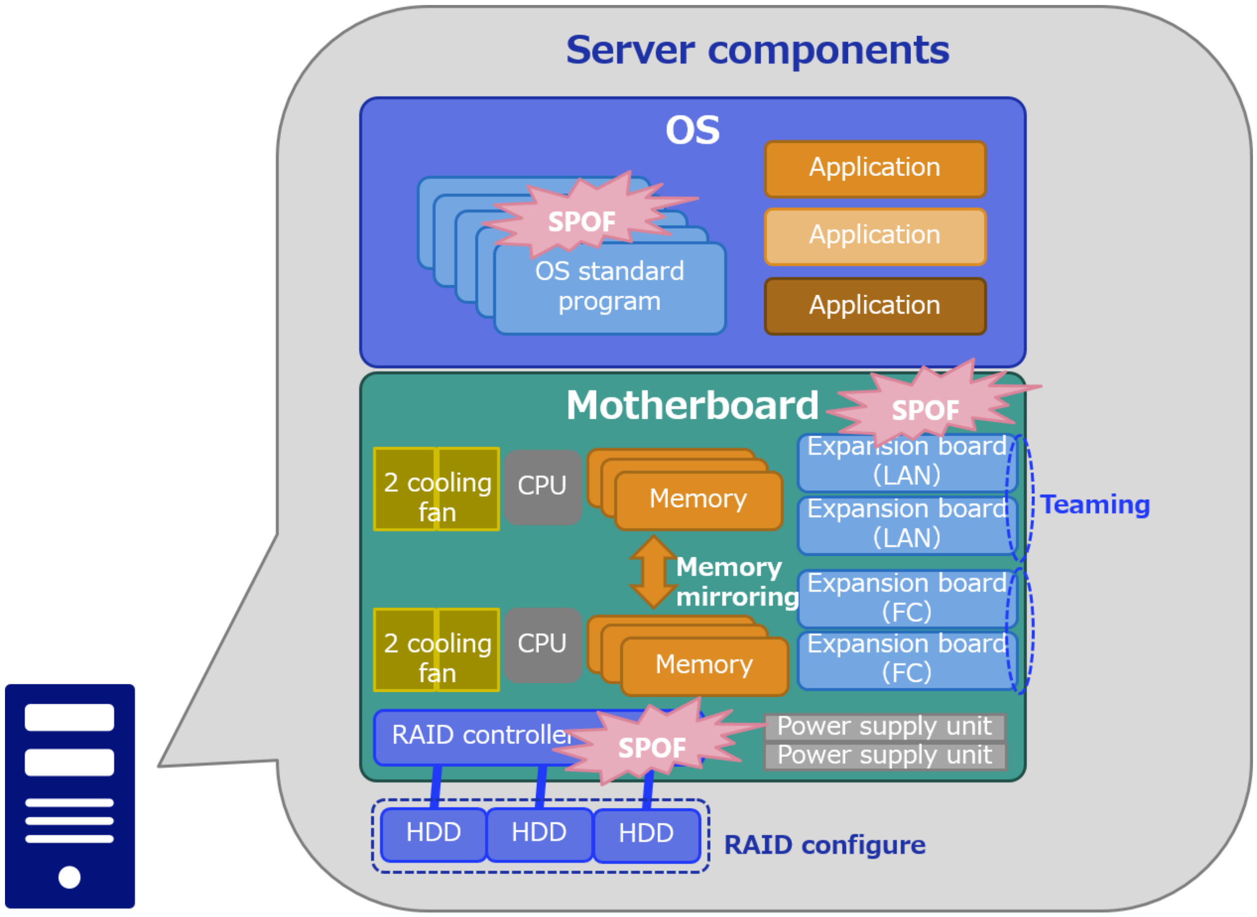
The document also addresses the necessity of implementing test procedures to validate the success of the deployment plan. Rigorous testing is crucial to identifying potential issues, verifying failover mechanisms, and ensuring the overall resilience of the HA cluster. Test procedures will be designed to simulate various scenarios, including failovers, load handling, and disaster recovery, contributing to a more robust and reliable deployment.

Lastly, the deployment plan will consider any constraints that may impact the implementation of the high-availability cluster. This could include budget limitations, resource constraints, or compatibility issues with existing infrastructure. Identifying and addressing these constraints in the planning phase is essential for a successful deployment that aligns with organizational goals and limitations.

By addressing these key questions and considerations, the document aims to provide a comprehensive and detailed plan for extending high-availability functionality to the core components while maintaining a focus on achieving close to 99.99% uptime. The outcomes of this research will contribute to a resilient and reliable IT infrastructure capable of meeting the demands of a high-performance environment.

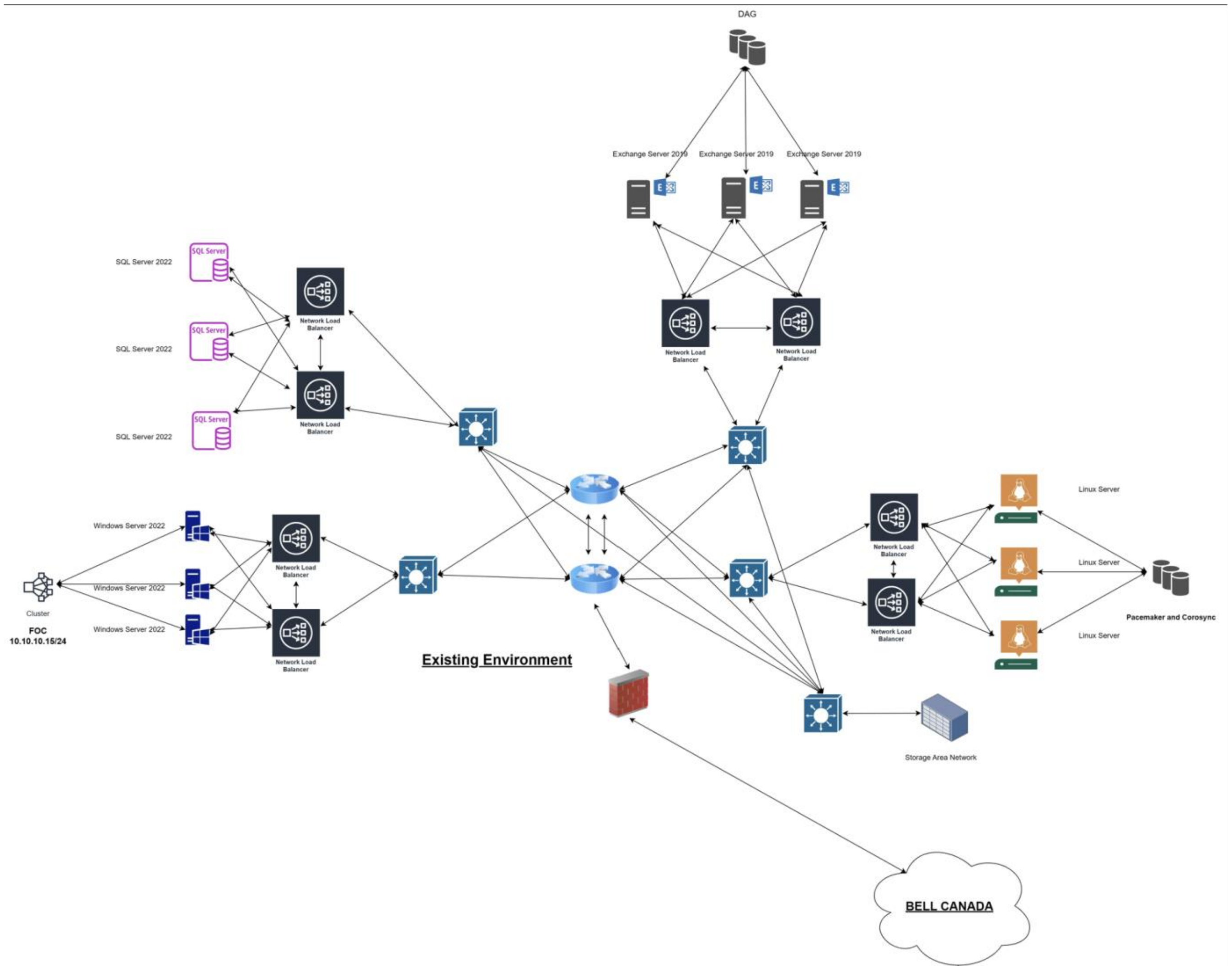
<u>Core Component Identification:</u>

Server with redundancy:



(*HA Cluster Introduction Part 2: Redundancy of Various Components*, n.d.)

Diagram of HA Cluster within Existing Environment:



Cluster IP addressing:

	Cluster Name	IP Address
Failover Cluster for Windows Server (On-Premises)	FOC	10.10.10.15/24
Load Balancer for Windows Servers (On-Premises)	LB_WS_1	10.10.10.21/24

Load Balancer for Windows Servers (On-Premises)	LB_WS_2	10.10.10.22/24
Load Balancer for SQL Servers (On-Premises)	LB_SS_1	10.10.10.23/24
Load Balancer for SQL Servers (On-Premises)	LB_SS_2	10.10.10.24/24
DAG Cluster for Exchange Server (On-Premises)	ES_DAG	10.10.10.16/24
Load Balancer for Exchange Servers (On-Premises)	LB_ES_1	10.10.10.25/24
Load Balancer for Exchange Servers (On-Premises)	LB_ES_2	10.10.10.26/24
High Availability Cluster for Linux Server (On-Premises)	Linux_Cluster_On	10.10.10.17/24
Load Balancer for Linux Servers (On-Premises)	LB_LS_1	10.10.10.27/24
Load Balancer for Linux Servers (On-Premises)	LB_LS_2	10.10.10.28/24
Failover Cluster for Windows Server (Cloud)	FOC_Cloud	10.10.10.18/24
Load Balancer for Windows Servers (Cloud)	LB_WS_1_C	10.10.10.29/24
Load Balancer for Windows Servers (Cloud)	LB_WS_2_C	10.10.10.30/24
Load Balancer for SQL Servers (Cloud)	LB_SS_1_C	10.10.10.31/24
Load Balancer for SQL Servers (Cloud)	LB_SS_2_C	10.10.10.32/24
DAG Cluster for Exchange Server (Cloud)	ES_DAG_C	10.10.10.19/24
Load Balancer for Exchange Servers (Cloud)	LB_ES_1_C	10.10.10.33/24
Load Balancer for Exchange Servers (Cloud)	LB_ES_2_C	10.10.10.34/24
High Availability Cluster for Linux Server (Cloud)	Linux_Cluster_C	10.10.10.20/24
Load Balancer for Linux Servers (Cloud)	LB_LS_1_C	10.10.10.35/24
Load Balancer for Linux Servers (Cloud)	LB_LS_2_C	10.10.10.36/24

Proposed New Hardware:

Several components of a server contribute to increasing availability and minimizing downtime. These components are designed to provide redundancy, fault tolerance, and the ability to recover from failures. Here are key server components that can enhance availability:

Redundant Power Supply:

- Power supply failures can lead to server outages. Redundant power supplies ensure continuous power, reducing the risk of downtime.
- Dell PowerEdge R750xd
  - Two hot-plug redundant power supplies.
  - 1600W Power supply

Redundant Fans:

- Efficient cooling is crucial for preventing hardware overheating. Redundant fans contribute to the server's reliability by maintaining optimal temperatures.
- HPE ProLiant DL380 Gen10
  - Redundant dual-rotor hot-plug fans.
  - Enhanced fan efficiency

Redundant Network Interfaces:

- Network connectivity is essential. Redundant network interfaces provide resilience against network-related failures and help maintain continuous communication.
- Cisco Catalyst 9300 Series Switches
  - Multiple Gigabit and 10-Gigabit Ethernet ports.
  - Optional redundant power supplies.

Out-of-Band Management:

- Remote management capabilities ensure administrators can monitor and manage servers even if the primary network is down, facilitating prompt response to issues.
- iDRAC (Integrated Dell Remote Access Controller)
  - Remote console access.
  - Power monitoring and management.

Redundant NICs (Network Interface Cards):

- Network redundancy is crucial for maintaining connectivity. Redundant NICs enhance network resilience and contribute to uninterrupted operations.
- Intel X710 Dual Port 10GbE SFP+ Adapter
  - Dual 10GbE SFP+ ports.
  - NIC teaming support for redundancy and load balancing.

Load Balancer:

A hardware load balancer (HLB) is a dedicated physical device designed to distribute incoming network traffic across multiple servers or computing resources. It typically operates at Layer 4 (Transport Layer) or Layer 7 (Application Layer) of the OSI model, providing load balancing based on factors such as IP addresses, ports, or application content.

Here we are using **F5 BIG-IP i7800** model as a hardware load balancing device.

Specifications:

- Throughput: Up to 60 Gbps at layer 4, Up to 20 Gbps at layer 7
- Comprehensive ADC features, including traffic management, load balancing, and application acceleration.
- Security Features: Provides protection against Distributed Denial of Service (DDoS) attacks. Protects web applications from various security threats using Web Application Firewall.
- Offers health monitoring and load balancing algorithms to ensure efficient distribution of traffic.
- Supports a wide range of protocols, including HTTP, HTTPS, TCP, UDP, and more.
- Intelligent Traffic Processing:
  - L7 requests per second: 3M
  - L4 connections per second: 1.1M
  - L4 HTTP requests per second: 14M
  - Maximum L4 concurrent connections: 80M
- Hardware Compression: 20 Gbps
- Memory: 96 GB DDR4
- Software Architecture: 64-bit TMOS
- Processor: One 6-Core Intel Xeon processor (total 12 hyperthreaded logical
- processor cores)

- Power Supply: 2x 650W Platinum AC PSU (2x 650W DC PSU Option)

Router Redundancy:

Implementing router redundancy typically involves using two or more routers in a redundant configuration to ensure high availability and network continuity. The specific hardware requirements for router redundancy depend on the chosen redundancy protocol, the network topology, and the level of redundancy needed.

Here we are using **Cisco Catalyst 9600 Integrated Services Routers** (ISR 9000).

Features:

- Scalability: They provide modular slots for various line cards, allowing organizations to expand their capabilities as network requirements evolve.
- High Performance: These routers offer high-performance routing capabilities, making them suitable for handling large volumes of network traffic efficiently. They are designed to provide low-latency and high-throughput performance, critical for demanding applications.
- Redundancy: The Cisco Catalyst 9600 Series routers support redundancy protocols like HSRP and VRRP, enabling high availability and automatic failover in case of router or link failure. This ensures continuous network operation and minimizes downtime.
- Advanced Security Feature: The routers include integrated security features such as firewall capabilities, VPN support, and threat defense mechanisms. This helps organizations protect their networks from cyber threats and vulnerabilities.
- Quality of Service (QoS): The routers support Quality of Service (QoS) features, allowing organization to prioritize and manage network traffic based on business-critical applications, ensuring optimal performance for essential services.

Proposed Cluster Design:

In our pursuit of achieving exceptional uptime close to 99.99%, we have implemented a robust hybrid solution that seamlessly integrates on-premises infrastructure with Microsoft Azure cloud services. This strategic approach leverages advanced clustering techniques to ensure high availability and fault tolerance across both physical servers within our local data center and virtual machines (VMs) hosted in the Azure cloud environment.

Windows Server:
Achieving high availability in a Windows Server 2022 environment involves creating a cluster design that ensures continuous operation, fault tolerance, and minimal downtime. Microsoft provides the Failover Clustering feature to create high-availability clusters.

Prerequisite:

- All servers are running on same version.
- While adding cluster storage ensure that all servers has access to storage area network.
- Make sure that all servers which we want to add to cluster are joined to same domain.
- Ensure that the user account that we are going to use for creating cluster has administrator privileges. Also user need to have permission for ‘create computer objects’.

Steps for Installing Failover Cluster:

- Open Server Manager.
- Select ‘Add Roles and Feature’ option from tools.
- On the select install type page, select ‘Role based installation’ and select next.
- On the select server role select next. On the ‘select feature’ select ‘Failover Clustering’ check box.
- Then add features and select next.
- On confirm installation selection page click install.

Perform all steps in all three windows server 2022 which we have in existing environments and four windows server 2022 of new environment which we just acquired.

Validate Configuration:
Validating a failover cluster in Windows Server 2022 is a crucial step to ensure that the hardware, software, and network configurations meet the requirements for clustering. The Failover Cluster Validation Wizard helps identify potential issues before creating the cluster. Here's how you can validate a failover cluster:

- Open Failover Cluster Manager.
- In the left navigation pane, select "Failover Cluster Manager."
- Under the "Management" section, click on "Validate a Configuration."
- In the Validation Wizard, enter the names of the servers that will be part of the failover cluster.
- Click "Next" to proceed.
- Choose the tests you want to run. For a comprehensive validation, select "Run all tests" or select specific tests based on your requirements.
- Click "Next" to proceed.
- Review the selected options and tests.
- Click "Next" to start the validation process.
- The wizard will run the selected tests to validate the failover cluster configuration.
- Monitor the progress, and the wizard will display the results.
- After the validation is complete, review the results. Address any warnings or errors identified by the wizard.
- Click "Finish" to close the wizard.

We need to perform same steps in another environment and have to add all four servers for cluster validation.

Create Failover Cluster:
Creating a failover cluster in Windows Server 2022 involves several steps. Below is a step-by-step guide using the Failover Cluster Manager. Ensure that we have already validated the cluster configuration using the Failover Cluster Validation Wizard before proceeding with the creation.

- Open Failover Cluster Manager.
- In the Failover Cluster Manager, under the "Management" section, click on "Create Cluster."
- The Create Cluster Wizard will open. Click "Next" to proceed. Enter the names of the servers or else we can use IP address of servers that will be part of the failover cluster.

- Click "Next" to proceed.
- The wizard will run a validation to ensure that the selected nodes and configuration meet the requirements for clustering.
- Review the summary, and if the validation completes successfully, click "Next."
- Enter a name for the cluster as “FOC”.
- Assign an IP address as ’10.10.10.15/24’ and network name for administering the cluster. Click "Next" to proceed.
- Review the configuration summary. Click "Next" to create the cluster.
- The wizard will create the failover cluster. Monitor the progress, and once completed, review the summary.
- Click "Finish" to close the wizard.

In the same way we have to create cluster but here in cluster name we have to give “FOC2” and IP address as ’10.10.10.16/24’.

Add Storage to Cluster:

- In Failover Cluster Manager, right-click on the cluster name ‘FOC’ in the navigation pane and select "More Actions" > "Configure Cluster Quorum Settings."
- In the Quorum Configuration Wizard, select disk witness on the SAN.
- Follow the wizard to complete the quorum configuration.
- Then in Failover Cluster Manager, right-click on "Services and Applications" and select "Configure a Service or Application."
- Follow the wizard to configure CSV.
- In Failover Cluster Manager, navigate to "Storage" in the navigation pane.
- Right-click on "Disks" and select "Add Disk."
- Select the LUNs from the SAN that want to add to the failover cluster.
- In Failover Cluster Manager, right-click on "Roles" and select "Configure Role."
- Follow the wizard to configure the role, specifying storage from the SAN.
- Verify that the storage is visible and accessible from all nodes in the cluster.
- Ensure that the disk resources are online and available.
- Perform a test failover of the cluster roles to ensure that they can move between cluster nodes seamlessly without issues.

SQL Server:

Database Mirroring is a deprecated high-availability feature in SQL Server that provides real-time data replication between two SQL Server instances. While it has been superseded by Always On Availability Groups in terms of features and capabilities, Database Mirroring may still be relevant in certain scenarios, and it's essential to understand its characteristics and limitations.

Key Characteristics of Database Mirroring:

Synchronous and Asynchronous Modes:

Database Mirroring supports both synchronous and asynchronous modes. In synchronous mode, transactions are committed on the principal and mirrored databases simultaneously, ensuring data integrity but potentially impacting performance. In asynchronous mode, the mirroring server may lag behind the principal, providing better performance but with the risk of potential data loss in case of a failover.

Automatic and Manual Failover:

Database Mirroring allows for both automatic and manual failover. Automatic failover occurs when the principal server becomes unavailable. Manual failover can be initiated for planned maintenance or in scenarios where administrators want control over the failover process.

Witness Server:

A witness server can be configured to vote in case of a dispute between the principal and mirror servers. This helps achieve automatic failover even in the absence of direct communication between the principal and mirror.

Reasons Database Mirroring Might be Considered:

Ease of Setup and Management:

Database Mirroring is known for its simplicity in setup and management compared to some other high-availability solutions. This can be beneficial for organizations with limited resources.

Lightweight on Resources:

In environments where minimal impact on system resources is a priority, Database Mirroring in asynchronous mode might be a suitable option as it can provide high availability with less impact on transactional performance compared to synchronous solutions.

Compatibility with Legacy Systems:

In scenarios where legacy systems are still in use and migration to newer SQL Server versions or features is not immediately feasible, Database Mirroring can be a transitional solution.

Considerations and Limitations:

Deprecated Feature:

Database Mirroring is marked as deprecated starting from SQL Server 2012, and it's not available in future versions. Organizations are encouraged to consider alternative high-availability solutions.

Limited Scale-Out Capabilities:

While Database Mirroring provides high availability at the database level, it does not offer the same scale-out capabilities as Always On Availability Groups, which can span multiple databases and instances.

No Load Balancing:

Database Mirroring does not provide load balancing across multiple servers. Always On Availability Groups, in contrast, allows for read-only secondary replicas to offload read workloads.

Reduced Features Compared to Always On Availability Groups:

Always On Availability Groups, introduced in SQL Server 2012, offers advanced features such as support for multiple databases, read-only replicas, and integration with Windows Server Failover Clustering.

Prerequisites:

Edition Compatibility:

Ensure that the SQL Server editions on both the principal and mirror servers are compatible for Database Mirroring.

Connectivity:

Ensure that there is proper network connectivity between the principal, mirror, and witness servers.

Service Accounts:

Ensure that the SQL Server services on all servers (principal, mirror, and witness) are running under domain user accounts with the necessary permissions.

Firewall Configuration:

Adjust the firewall settings on all servers to allow communication on the specified Database Mirroring port (default is TCP 7022). Open the port for both inbound and outbound traffic.

Here, for both environments we have two SQL servers so before starting mirroring process we have to deploy one more SQL server 2022 in each environment.

Steps to Configure Database Mirroring:
Backup Database

- Connect to the Principal Server. Open SQL Server Management Studio (SSMS)
- In the Object Explorer, navigate to the "Databases" node.
- Right-click on the database want to back up. Choose "Tasks" -> "Backup."
- In the Backup Database dialog, configure the backup options, including the destination as a ‘SMB share’ for the backup file.
- Click "OK" to perform the backup.

Restore Database:

- Access SMB Share which has database backup files.
- Open SSMS and connect to the mirror server.
- In the Object Explorer, navigate to the "Databases" node.
- Right-click on the "Databases" node and choose "Restore Database."
- In the "General" page of the Restore Database dialog, select the source of the database from SMB share to restore it.
- Here we need to select ‘Backup with no recovery option’.
- Click "OK" to perform the restore.

Configure Database Mirroring:

- Connect to principal server instance.
- Select database to be mirrored.
- By right clicking database select ‘Tasks’ and then click Mirror option.
- Click ‘Configure Security’ button to open configure database mirroring security.

- This wizard automatically creates the database mirror endpoints. Fill up filed of principle, mirror and witness based on role of server instance.
- Set operating mode as high safety with automatic failover(synchronous).

Exchange Server:
Implementing high availability in Exchange Server 2019 involves configuring redundancy and failover mechanisms to ensure continuous availability of mailbox services.

A Database Availability Group (DAG) in Exchange Server is a high-availability and disaster recovery solution that provides automatic database-level recovery from failures. It offers several advantages over alternative high-availability options, such as clustering or standalone mailbox servers.

Reasons Why DAG:
Automated Database-Level Failover:

DAG provides automated database-level failover, allowing for quick and automatic recovery in the event of a failure. It doesn't rely on manual intervention, reducing downtime and improving overall system availability.

Incremental Deployment:

DAG allows for incremental deployment, meaning we can add additional mailbox servers to the DAG as our organization grows. This provides scalability and flexibility without the need for a complete overhaul of the infrastructure.

Active-Active Configuration:

DAG supports an active-active configuration where multiple mailbox servers can host copies of the same databases simultaneously. This enables load balancing of client connections, optimizing resource utilization across the environment.

Database Copies:

DAG allows for multiple copies of mailbox databases to be maintained across different servers. These copies provide redundancy, and in the event of a failure, clients can automatically connect to an available copy of the database.

Site Resilience:

DAG supports site resilience by allowing you to deploy mailbox servers in different physical locations. This ensures that if an entire site becomes unavailable, mailbox services can continue from the surviving site.

No Shared Storage Requirement:

Unlike traditional clustering solutions, DAG doesn't require shared storage. Each mailbox server in the DAG maintains its own copy of the databases, eliminating the need for complex and expensive shared storage configurations.

Flexibility in Witness Server Placement:

DAG provides flexibility in choosing the witness server location. The witness server can be placed in a third datacenter, a different geographical location, or even in a cloud environment, enhancing the overall resilience of the solution.

Built-In Monitoring and Alerting:

Exchange Server includes built-in monitoring and alerting features for DAG. Administrators can easily monitor the health and status of the DAG through tools like Exchange Management Shell (EMS) or Exchange Admin Center (EAC).

Incremental Backup and Restore:

DAG supports incremental backup and restore processes, making it easier to back up and restore specific databases without affecting the entire environment.

Integration with Exchange Features:

DAG seamlessly integrates with other features of Exchange Server, such as the Client Access Server (CAS) array and load balancing, providing a comprehensive high-availability solution.

Prerequisites:

- Ensure that the servers have adequate hardware resources and are running a supported operating system.
- Each server in the DAG should have at least two network interfaces—one for MAPI and another for replication traffic.
- Assign static IP addresses to the network interfaces.

Here, before creating and configuring DAG we need to deploy one more Microsoft Server 2019 in each environment.

Steps to Create DAG:

- Open the Exchange Management Shell (EMS) on one of the servers.
- Run the following command to create a DAG:

“New-DatabaseAvailabilityGroup -Name DAG_Name -WitnessServer WitnessServer_Name -WitnessDirectory Path_WitnessDirectory”

- Add Servers to DAG

“Add-DatabaseAvailabilityGroupServer -Identity DAG_Name -MailboxServer ServerName”

- Configure DAG Networks:

“Set-DatabaseAvailabilityGroupNetwork -Identity DAG_Name -ReplicationEnabled $true -IgnoreNetwork $false”

- Run the following command to create a copy of each mailbox database on another DAG member:

“Add-MailboxDatabaseCopy -Identity "DatabaseName" -MailboxServer "MailboxServerName" ”

- Monitor the status using “Get-MailboxDatabaseCopyStatus” command.
- Ensure that each DAG member has a static IP address for the DAG.
- Run the following command to configure the DAG IP address:

“Set-DatabaseAvailabilityGroup -Identity DAG_Name -DatabaseAvailabilityGroupIPAddresses IPAddress”

- Run the following command to configure the witness server and directory:

“Set-DatabaseAvailabilityGroup -Identity DAG_Name -WitnessServer WitnessServerName -WitnessDirectory Path_WitnessDirectory”

- Run the following command to set the AutoDatabaseMountDial to "BestAvailability":

“Set-MailboxServer -Identity "MailboxServerName" -AutoDatabaseMountDial BestAvailability”

- Test the DAG failover by simulating failures and ensuring that databases switch to other DAG members.

In summary, DAG is preferred for its automated failover, scalability, flexibility, and built-in monitoring capabilities, making it a robust solution for ensuring high availability and resilience in Exchange Server environments. To configure it using above steps.

Linux Server:

Creating a high-availability (HA) cluster in Linux involves several technologies and mechanisms. One popular approach is to use clustering software that provides tools for managing and monitoring the cluster, facilitating failover, and ensuring high availability of services. One of the commonly used clustering solutions for Linux is the Pacemaker and Corosync stack.

Pacemaker and Corosync are commonly used as a combination to create high-availability clusters in Linux environments. There are several reasons why Pacemaker and Corosync are preferred over other mechanisms for implementing high-availability which are described below.

- Open Source and Well-Established
- Active Community and Development
- Flexible and Extensible
- Resource Orchestration
- Quorum Mechanism
- Support for Multiple Services
- Active/Passive and Active/Active Configurations
- Integration with Clustered File Systems
- Compatibility with Different Linux Distributions

Steps to Configure Pacemaker and Corosync:

- Install Pacemaker and Corosync using ‘sudo yum install pacemaker corosync’ command.
- Edit the Corosync configuration file (/etc/corosync/corosync.conf) to define the cluster nodes and communication settings.
- Start and enable the Corosync service on each node using ‘sudo systemctl start corosync; sudo systemctl enable corosync’ command.
- Install the Pacemaker package on each node using ‘sudo yum install pacemaker’ command.

- tart and enable the Pacemaker service on each node using ‘sudo systemctl start pacemaker; sudo systemctl enable pacemaker’ command.
- Use the crm command-line tool to configure cluster resources such as virtual IP addresses, services, and resource groups.
- Set constraints and policies to define how resources should be managed, and under what conditions failover should occur.
- Use commands like crm status or crm_mon to verify the status of the cluster and its resources.

On-Premise Load Balancer:

Load balancing algorithms are techniques used by load balancers to distribute incoming network traffic across multiple servers or resources. Each algorithm has its own characteristics, advantages, and use cases.

Round Robin Load Balancing Algorithm:

The Round Robin (RR) load balancing algorithm is one of the simplest and widely used methods for distributing incoming network traffic across a group of servers. In a Round Robin scheme, each incoming request is sequentially assigned to the next server in the list, forming a circular order. The algorithm cycles through the servers in a loop, ensuring an equal distribution of requests among all available servers.

How Round Robin Works:

- Sequential Assignment: Requests are distributed in a sequential order to each server in the list.
- Equal Distribution: Each server gets an approximately equal share of the incoming traffic, promoting a balanced workload.
- No Consideration of Server Load: Round Robin does not take into account the current load or performance metrics of the servers. It assumes that all servers are equally capable of handling requests.
- Stateless Operation: Round Robin is stateless and does not retain information about the state of servers or previous requests. Each request is treated independently.
- Simple Implementation: The simplicity of the Round Robin algorithm makes it easy to implement and manage. It does not require complex configurations or frequent adjustments.

Advantages of Round Robin:
Simplicity: Round Robin is easy to understand and implement, making it a straightforward choice for basic load balancing requirements.

Uniform Distribution: In environments where servers have similar capacities and workloads, Round Robin ensures a fairly uniform distribution of requests.

No Server Preference: Every server is treated equally, which can be advantageous in scenarios where all servers are expected to handle similar workloads like us.

Considerations and Limitations:
Unequal Capacities: If servers have different capacities or performance characteristics, Round Robin may lead to uneven distribution, as it does not consider server load or responsiveness.

No Session Persistence: Round Robin does not guarantee that requests from the same client are directed to the same server. This lack of session persistence may impact applications that rely on consistent session states.

Limited Intelligence: The algorithm lacks intelligence about server health or current utilization, making it less suitable for dynamic or heterogeneous server environments.

Steps to Configure F5 BIG-IP i7800 Load Balancer:
Assigning an IP address to an F5 BIG-IP load balancer involves configuring a management IP address for accessing the system and, assigning virtual IP addresses for load balancing services.

Assign Management IP Address:

- Open a web browser and enter the management IP address of the F5 BIG-IP load balancer in the address bar. Then login with appropriate credentials.
- In the Configuration utility, navigate to "Network" -> "Self IPs."
- Click on the "Create" button to add a new Self IP.
- Enter a name for the Self IP.
- Specify the IP address and subnet mask.
- Choose the VLAN 10 and set other relevant options.
- Click "Finished" to save the Self IP configuration.

Deploy another load balancer for redundancy and assign IP address to it. For that we have to perform same as above.

Assign Virtual IP Address for Load Balancing:

This configuration needs to be done in both load balancers and steps are same for both.

- In the Configuration utility, go to "Local Traffic" -> "Virtual Servers."
- Click on the "Create" button to define a new virtual server.
- Enter a name as 'LB_1' in first load balancer and 'LB_2' in second load balancer for the virtual server.
- Assign an IP addresses of Linux servers for the load balancing service.
- Choose the appropriate protocol (HTTP, HTTPS, etc.).
- Configure additional settings like the default pool, profiles, and iRules.
- Click "Finished" to save the virtual server configuration.
- Confirm that the new virtual server is listed in the "Virtual Servers" section.

Configure Synchronization:

- Log in to the F5 BIG-IP Configuration Utility.
- Navigate to System > High Availability > Device Management.
- In the Overview tab, click on Setup under Device Trust.
- Enable Device Trust and set a passphrase for secure communication between devices.
- Click Finished to save the settings.
- In the Device Management menu, click on Overview.
- Click on Sync-Failover in the menu.
- Under Configuration, click Change Device.
- Enter the IP address of the second device, specify the passphrase, and click Finished to save.
- Click Sync to synchronize the configuration to the peer device.

Implement Health Monitoring:

- Navigate to Local Traffic > Monitors.
- Click Create to add a new health monitor.
- Enter a name, select the type of monitor (e.g., HTTP, TCP), and configure parameters such as the interval, timeout, and expected response.
- Click Finished to save the monitor.
- Navigate to Local Traffic > Pools.
- Edit the pool we created for our backend servers.
- Under Health Monitors, add the previously configured health monitor.
- Click Finished to save the pool configuration.

Define Failover Mechanism:

- Navigate to System > High Availability > Failover.
- Under Failover Trigger, configure conditions that will trigger failover, such as Network Failures, System Failures, or Manual Failover.
- Click Update to save the failover trigger settings.
- In the Failover menu, click on Failover under Failover Policies.
- Configure the actions as Switch to Standby.
- Set priorities for each device in the device group.
- Click Finished to save the failover policy.

The same way we have to deploy load balancers for SQL Servers, Exchange Servers and Linux Servers.

Microsoft Azure Cloud:

To achieve high availability for our infrastructure we are using hybrid solution in which we decided use Microsoft Azure Cloud. The reason behind this is given below.

Hybrid Cloud Capabilities:

Azure is known for its strong support for hybrid cloud deployments, allowing organizations to seamlessly integrate on-premises data centers with cloud services. This is particularly beneficial for businesses with existing infrastructure investments.

Enterprise Integration:

Microsoft Azure is often preferred by enterprises that use Microsoft technologies such as Windows Server, Active Directory, and .NET. It provides seamless integration with Microsoft's suite of productivity tools like Office 365 and Dynamics 365.

Diverse Service Offerings:

Azure offers a comprehensive set of services, including computing, storage, databases, AI, machine learning, IoT, and more. This extensive range of services allows organizations to build, deploy, and manage applications across various domains.

Global Presence:

Microsoft has an extensive network of data centers globally, allowing users to deploy applications and services in multiple regions. This helps in reducing latency and ensuring high availability.

Security and Compliance:

Azure places a strong emphasis on security and compliance. It provides a range of security features, including identity management, threat detection, and encryption, to help organizations protect their data and meet regulatory requirements.

Enterprise Agreements:

Microsoft offers flexible licensing options and enterprise agreements, making it easier for large organizations to manage costs and scale their infrastructure based on their needs.

Developer Tools:

Azure provides a set of robust developer tools and supports various programming languages and frameworks. Integration with Visual Studio and a rich set of development tools can make it easier for developers to build and deploy applications.

AI and Cognitive Services:

Azure has a strong focus on artificial intelligence (AI) and cognitive services, offering a wide range of tools and APIs for machine learning, natural language processing, computer vision, and more.

Ecosystem and Partnerships:

Microsoft Azure has a vast ecosystem of partners and a marketplace where users can find and deploy third-party solutions. This can simplify the process of integrating additional services and applications into a cloud environment.

Windows Server in Cloud:

Deploying a failover cluster for Windows Server 2022 VMs in Azure involves several steps. Here we are creating 3 virtual machines. Below is a high-level guide to help us through the implementation process.

- Deploy three Windows Server 2022 VMs in Azure. Make sure these VMs are part of the same Azure Virtual Network.

- Assign static IP addresses 10.10.10.31/24, 10.10.10.32/24, 10.10.10.33/24 to the VMs.
- Ensure proper DNS configuration for name resolution.
- Join each Windows Server VM to the same Active Directory domain.
- Now for installing failover cluster feature, validate cluster configuration and creating cluster we have to follow same steps that we followed for on-premise server.
- Here we are creating separate cluster from cluster that we created for on-premise windows server.
- Here we need to give cluster name as 'FOC_Cloud_WS' and cluster ip address as '10.10.10.21/24'.

SQL Server in Cloud:
Implementing a high-availability (HA) cluster for SQL Server in the Azure cloud involves using Always On Availability Groups, which is a feature of SQL Server designed to provide high availability and disaster recovery.

- Deploy three SQL Server 2022 VMs in Azure. Make sure these VMs are part of the same Azure Virtual Network.
- Connect to each Azure VM and install SQL Server 2022.
- Create failover cluster for these 3 VMs. For this we need to implement same steps that we used in windows server. Here configure cluster name as 'FOC_Cloud_SQL' and cluster ip address as '10.10.10.22/24'.
- configure a shared disk that can be accessed by all cluster nodes.
- Connect to each SQL Server instance and enable the Always On Availability Groups feature.
- Use SQL Server Management Studio (SSMS) or T-SQL to create an Availability Group and add the desired databases to it.
- Set up an Availability Group listener to provide a consistent endpoint for client connections.
- Add the Azure VMs as replicas to the Availability Group.
- configure the Availability Group for automatic failover.

Exchange Server in Cloud:
Creating a high-availability (HA) cluster for Microsoft Exchange Server in the Azure cloud typically involves deploying multiple Exchange Server instances in a database availability group (DAG). The DAG provides automatic database-level recovery from failures that affect individual servers or databases.

- Deploy three Exchange Server 2019 VMs in Azure. Make sure these VMs are part of the same Azure Virtual Network.
- To create DAG we are following same steps that we did in on-premises exchange servers.

Linux Server in Cloud:
Creating a high-availability (HA) cluster for Linux servers in Azure involves using clustering software to ensure that your applications and services remain available in the event of a failure. Here, I'll outline a general approach using Pacemaker and Corosync, two commonly used tools for creating HA clusters in Linux.

- Deploy three Linux Server VMs in Azure. Make sure these VMs are part of the same Azure Virtual Network.
- To Install and configuration of Pacemaker and Corosync HA cluster we have to follow same procedure that we did in on-premises linux servers.

Azure Load Balancer:
Implementing load balancing with round-robin algorithms in Azure involves using Azure Load Balancer, a Layer 4 (TCP, UDP) load balancer that distributes incoming network traffic across multiple servers to ensure no single server is overwhelmed with too much traffic.

Steps to Configure Load Balancer:
- In the Azure portal, go to "Create a resource" > "Networking" > "Load Balancer."
- Configure the basic settings, such as the subscription, resource group, and region.
- Define a frontend IP configuration as ’10.10.10.12/24’ and link it to a backend pool containing the VMs want to load balance.
- Set up health probes to monitor the health of your VMs. This ensures that traffic is only directed to healthy VMs.
- Define load balancing rules to specify how incoming traffic is distributed. Choose the round-robin algorithm to evenly distribute traffic among the VMs.
- Review your configuration and create the load balancer.
- Ensure that your VMs' Network Security Groups allow traffic from the load balancer. Inbound rules should permit traffic on the required ports (e.g., HTTP, HTTPS).

Cluster Connectivity:

Connecting an on-premise high-availability (HA) cluster with an HA cluster in Azure, using Azure ExpressRoute, involves establishing a private, dedicated network connection between our on-premises infrastructure and the Azure cloud. This ensures secure and reliable communication between your local data center and resources deployed in Azure.

Why Azure ExpressRoute?

Azure ExpressRoute is a dedicated, private connection between your on-premises infrastructure and Azure. Here are reasons why it might be preferred over other mechanisms.

Private Connection:

ExpressRoute provides a private, dedicated connection that doesn't traverse the public internet, ensuring enhanced security and reliability.

Predictable Performance:

With ExpressRoute, we can expect more consistent and predictable network performance compared to public internet connections.

Reduced Latency:

ExpressRoute often offers lower latency compared to public internet connections, which can be crucial for applications with low-latency requirements.

Enhanced Security:

The private connection provided by ExpressRoute enhances security, making it suitable for scenarios where data privacy and compliance are critical.

Increased Bandwidth Options:

ExpressRoute provides flexible bandwidth options, allowing us to choose the right level of connectivity based on our requirements.

Integration with Azure Services:

ExpressRoute integrates seamlessly with various Azure services, providing a dedicated path for services like Azure Virtual Networks.

Global Reach:

ExpressRoute offers global reach, allowing us to connect to Azure data centers worldwide.

Why Bell Canada?

The choice of a third-party service provider, such as Bell Canada, for networking services including Azure ExpressRoute, can depend on various factors including regional considerations, service offerings, and customer requirements. Here are some potential advantages of choosing Bell Canada as a third-party service provider for networking services in Toronto:

- Regional Presence: Bell Canada has a strong regional presence, providing better connectivity and lower latencies in Toronto.
- Extensive Network Infrastructure: Bell Canada operates a reliable network infrastructure, ensuring stable and robust connectivity.
- Local Expertise and Support: Local teams with expertise in the Toronto market offer on-site support and quick response times.
- Diverse Connectivity Options: Bell Canada provides various bandwidth plans, redundancy options, and flexible network configurations.
- Compliance and Security: Adherence to industry standards and compliance regulations is crucial for businesses with specific requirements.
- Integrated Solutions: Beyond connectivity, Bell Canada may offer integrated solutions, simplifying network management.
- Scalability: Bell Canada's services are scalable, accommodating growth and changes in network demands.
- Service Level Agreements (SLAs): Robust SLAs provide assurance of service reliability.
- Partnership with Microsoft: Collaboration with Microsoft ensures seamless integration and support for Azure services.
- Customer Reviews and Reputation: Researching customer reviews and Bell Canada's reputation helps gauge service quality and support.

Implementing Azure ExpressRoute with a third-party connectivity provider, such as Bell Canada, involves coordination between our organization, the connectivity provider, and Microsoft Azure.

For this we have to reach out to Bell Canada to initiate the process of establishing an ExpressRoute connection. Discuss our connectivity requirements, bandwidth needs, and any additional services offered by Bell Canada.

Steps to Configure Azure ExpressRoute:

Configuring Azure ExpressRoute involves several steps, including setting up the ExpressRoute circuit in the Azure portal, associating the Azure Virtual Network, and establishing the on-premises connectivity.

- In the Azure portal, go to "Create a resource" > "Networking" > "ExpressRoute."
- Follow the wizard to provide details for the ExpressRoute circuit, such as circuit name, SKU, and bandwidth.
- Specify the settings for the ExpressRoute circuit, including the location, peering location, and SKU.
- Define the routing preference for the circuit.
- Review the configuration settings and click "Create" to provision the ExpressRoute circuit.
- After creating the ExpressRoute circuit, navigate to the "Virtual network connections" section within the ExpressRoute circuit.
- Click on "Add virtual network" and select the Azure Virtual Network that want to associate with the ExpressRoute circuit.
- Configure the settings for the virtual network connection.
- Here we need to define route filters to specify how traffic is routed between on-premises networks and Azure. Add the necessary routes for the Azure Load Balancer's IP address space are included.
- Review the settings and save the virtual network connection.

On-Premises Configuration:

- First we need to share the ExpressRoute circuit details obtained from Azure with Bell Canada. This will include circuit ID, peering locations, and other specific configurations required.
- Coordinate with Bell Canada to set up the physical connection between our on-premises network and the ExpressRoute circuit.
- Configure the routing on our on-premises network to direct traffic destined for Azure over the ExpressRoute circuit.
- Establish BGP (Border Gateway Protocol) peering between our on-premises router and the Microsoft edge router.
- Exchange Autonomous System Numbers (ASN) information with Microsoft to facilitate BGP peering.
- Define that traffic intended for the Azure Load Balancer's IP address is directed through the ExpressRoute connection.

- After completing the configuration, check the connectivity between your on-premises network and Azure resources.

Test Procedure to Validate HA:

Testing a high-availability (HA) cluster between on-premises and Azure involves validating the failover and load balancing capabilities of the cluster. Below is a detailed test procedure to validate an HA cluster for Windows Server 2022, SQL Server 2022, Exchange Server 2019, and Linux Server deployed on VMs in Azure environment and On-premise physical server.

Validate On-Premises HA Cluster:

Windows Server 2022

- Test 1: Failover of Windows Server Roles
  - Move Windows Server roles (e.g., DHCP, DNS) between nodes in the on-premises cluster. Verify that services remain available during the failover.
- Test 2: File Share Witness Testing
  - Simulate a node failure and ensure that the File Share Witness correctly determines the availability of nodes.

SQL Server 2022

- Test 1: Failover
  - Intentionally simulate a failure on the principal server (e.g., stop SQL Server service).
  - Monitor the mirroring state to ensure that failover to the mirror server occurs automatically.
  - Validate that the mirror server has taken over the role of the principal server.
  - Confirm that applications can connect to the new principal server without manual intervention.
- Test 2: Data Integrity
  - Simulate a database update on the principal server.
  - Monitor the mirroring state and confirm that the update is replicated to the mirror server.
  - Intentionally failover to the mirror server.
  - Validate that the data on the mirror server remains consistent after the failover.
- Test 3: Network Disruption

    - Temporarily disrupt the network connection between the principal and mirror servers.
    - Observe how the database mirroring handles the network interruption.
    - Restore the network connection and validate the synchronization between the principal and mirror servers.

Exchange Server 2022

- Test 1: Database Failover
    - Trigger a failover in Exchange Server 2019 Database Availability Group (DAG). Confirm that mailbox databases failover seamlessly.
- Test 2: Database Switchover
    - Perform a planned switchover of the active database to another DAG member. Confirm that the operation completes successfully.

Linux Server

- Test 1: Application Failover
    - Deploy a sample application on the Linux server cluster.
    - Simulate a failure of one of the cluster nodes.
    - Verify that the application fails over to another node in the cluster.
- Test 2: Service Failover
    - Deploy a service on the Linux server cluster (e.g., a web server).
    - Simulate a network failure or node failure.
    - Verify that the service fails over to another node, and clients can still access the service.

Validate Azure HA Cluster:

Windows Server 2022

- Test 1: Azure VM Failover
    - Simulate a failure of an Azure VM hosting a Windows Server 2022 node. Verify that Azure automatically restarts the VM and rejoins it to the cluster.
- Test 2: Azure Site Recovery (ASR)
    - Test failover using Azure Site Recovery to ensure a seamless transition of on-premises workloads to Azure in case of a disaster.

SQL Server 2022

- Test 1: Azure VM Failover
    - Simulate a failure of an Azure VM hosting a SQL Server 2022 node. Verify that SQL Server failover occurs and data remains available.

- Test 2: Database Instance Failure
    - Simulate database instance failover from one VM. Verify that data accessible or not.

Exchange Server 2022

- Test 1: Azure VM Failover
    - Simulate a failure of an Azure VM hosting an Exchange Server 2019 node. Confirm that mailbox databases failover to another Azure VM.
- Test 2: Database Switchover to Azure
    - Perform a planned switchover of the active database to an Azure-based DAG member. Confirm that the operation completes successfully.

Linux Server

- Test 1: Azure VM Failover
    - Simulate a failure of an Azure VM hosting a Linux Server cluster node. Verify that the Linux Server application fails over to another Azure VM.
- Test 2: Autoscaling
    - Test autoscaling capabilities in Azure for the Linux Server cluster to handle increased workloads.

Validate Cross-Environment Tests:

Hybrid Failover Testing:

- Test 1: On-Premises to Azure Failover
    - Trigger a planned failover from on-premises to Azure for one or more workloads. Verify that the failover occurs seamlessly.
- Test 2: Azure to On-Premises Failover
    - Reverse the direction and simulate a planned failover from Azure back to on-premises. Confirm that workloads successfully fail back.

Load Balancing Across Environments:

- Test 1: Cross-Environment Load Balancing
    - Distribute traffic across on-premises and Azure nodes using a load balancer. Verify that the load balancing algorithm is effective and evenly distributes traffic.
- Test 2: Autoscaling Across Environments
    - Test autoscaling capabilities that span on-premises and Azure environments.

Conclusion

In our pursuit of achieving an impressive uptime of close to 99.99%, we meticulously designed and implemented a resilient on-premises infrastructure. To fortify the reliability of our physical servers, we incorporated redundant power supplies, fans, and network interfaces (NICs). The F5 BIG-IP i7800 load balancer became a cornerstone in optimizing network traffic distribution, ensuring efficient load balancing across our server resources. Recognizing the critical role of network infrastructure, we bolstered our setup with router redundancy, utilizing the Cisco Catalyst 9000 Series Integrated Services Routers to enhance network availability. In a strategic move to further elevate availability, we embraced a hybrid infrastructure model, seamlessly integrating Azure cloud as our cloud solution. For our on-premise Windows servers, we implemented a failover cluster to ensure continuous availability. The SQL Server environment leveraged database mirroring for data redundancy and failover capabilities. Exchange Server, in turn, adopted a Database Availability Group (DAG) architecture for high availability and reliability. On the Linux side, we configured a robust High Availability (HA) cluster using Pacemaker and Corosync, ensuring that Linux-based services maintain uptime and reliability. In our Azure cloud environment, we deployed three virtual machines for each of the essential components - Windows Server 2022, SQL Server 2022, Exchange Server 2019, and Linux Server. Load balancers were intricately configured for each server instance to efficiently distribute incoming network traffic and optimize resource utilization. To seamlessly connect and manage both on-premises and Azure environments, Azure ExpressRoute was employed, offering a secure and high-performance connection. For external network support, we opted for Bell Canada as our third-party service provider, leveraging their expertise to enhance our connectivity and ensure robust networking capabilities. Testing the resilience of our HA clusters became a priority, and we meticulously crafted comprehensive test procedures. These procedures encompassed the validation of on-premises server clusters, testing the failover mechanisms in Azure cloud-based HA clusters, and conducting thorough cross-environment tests. Our commitment to meticulous testing ensures that our hybrid infrastructure can withstand real-world scenarios, providing the confidence that our systems will continue to operate seamlessly even in the face of unexpected challenges. This detailed approach to infrastructure design, implementation, and testing underscores our dedication to achieving and maintaining exceptional levels of system availability and reliability.

References

HA Cluster Introduction Part 2: Redundancy of various components. (n.d.). https://www.nec.com/en/global/prod/expresscluster/en/blog/20211004/ha-cluster-introduction-part2.html

Modular Infrastructure Solutions – Enterprise Servers. (n.d.). Dell USA. https://www.dell.com/en-us/dt/servers/modular-infrastructure/index.htm?gacd=9650523-1141-5763017-266683695-0&dgc=ST&SA360CID=71700000099052792&gclid=ad45c65d779518150b78a32e128aa759&gclsrc=3p.ds&msclkid=ad45c65d779518150b78a32e128aa759#tab0=0

HPE ProLiant DL380 Gen10 drivers - Bing. (n.d.). Bing. https://www.bing.com/search?q=HPE+ProLiant+DL380+Gen10+drivers&qs=n&form=QBRE&sp=-1&ghc=2&lq=0&pq=hpe+proliant+dl380+gen10+drivers&sc=7-32&sk=&cvid=9135BC603CE5444EA5DFC7E6483CEC2A&ghsh=0&ghacc=0&ghpl=

Cisco Catalyst 9300 Series Switches product video. (2023, November 17). Cisco. https://www.cisco.com/site/us/en/products/networking/switches/catalyst-9300-series-switches/index.html

Integrated Dell Remote Access Controller (iDRAC) | Dell Canada. (n.d.). Dell. https://www.dell.com/en-ca/lp/dt/open-manage-idrac

Intel X710 Dual Port 10GbE SFP+ Adapter, PCIe Full Height, V2 | Dell USA. (n.d.). Dell. https://www.dell.com/en-us/shop/intel-x710-dual-port-10gbe-sfp-adapter-pcie-full-height-v2/apd/540-bdrd/networking

F5 BIG-IP iSeries Platform. (n.d.). F5, Inc. https://www.f5.com/products/big-ip-services/iseries-appliance

J. (2022, February 11). Failover Clustering. Microsoft Learn. https://learn.microsoft.com/en-us/windows-server/failover-clustering/failover-clustering-overview

J. (2023, March 16). Create a failover cluster. Microsoft Learn. https://learn.microsoft.com/en-us/windows-server/failover-clustering/create-failover-cluster

M. (2022, November 18). Database Mirroring and Replication (SQL Server) - SQL Server Database Mirroring. Microsoft Learn. https://learn.microsoft.com/en-us/sql/database-engine/database-mirroring/database-mirroring-and-replication-sql-server?view=sql-server-ver16

M. (2022, November 18). Setting Up Database Mirroring (SQL Server) - SQL Server Database Mirroring. Microsoft Learn. https://learn.microsoft.com/en-us/sql/database-engine/database-mirroring/setting-up-database-mirroring-sql-server?view=sql-server-ver16

Tajran, A. (2023, May 9). Create DAG Exchange Server step by step. ALI TAJRAN. https://www.alitajran.com/create-dag-exchange-server/#:~:text=Configure%20Database%20Availability%20Group%201%20Configure%20File%20Share,. . .%204%20Verify%20File%20Share%20Witness%20folder%20

M. (2023, March 14). Configure a distributed availability group - SQL Server Always On. Microsoft Learn. https://learn.microsoft.com/en-us/sql/database-engine/availability-groups/windows/configure-distributed-availability-groups?view=sql-server-ver16&tabs=automatic

A. (2023, February 22). Exchange 2019 preferred architecture. Microsoft Learn. https://learn.microsoft.com/en-us/exchange/plan-and-deploy/deployment-ref/preferred-architecture-2019?view=exchserver-2019

A. (2023, February 22). Configure database availability group network properties. Microsoft Learn. https://learn.microsoft.com/en-us/exchange/high-availability/manage-ha/configure-dag-network-properties?view=exchserver-2019

A. (2023, February 22). Create a database availability group network. Microsoft Learn. https://learn.microsoft.com/en-us/exchange/high-availability/manage-ha/create-dag-networks?view=exchserver-2019

Anicas, M. (2015, October 20). How To Create a High Availability Setup with Corosync, Pacemaker, and Reserved IPs on Ubuntu 14.04. DigitalOcean. https://www.digitalocean.com/community/tutorials/how-to-create-a-high-availability-setup-with-corosync-pacemaker-and-reserved-ips-on-ubuntu-14-04

Installing and Configuring Pacemaker and Corosync. (2022, October 14). Oracle Help Center. https://docs.oracle.com/en/operating-systems/oracle-linux/8/availability/availability-InstallingandConfiguringPacemakerandCorosync.html#install-pacemaker

C. (2023, October 16). Quickstart - Create a Windows VM in the Azure portal - Azure Virtual Machines. Microsoft Learn. https://learn.microsoft.com/en-us/azure/virtual-machines/windows/quick-create-portal

Takor, H. (2023, March 25). Creating a Virtual Machine in Azure: A Step-by-Step Guide for Beginners. DEV Community. https://dev.to/henriettatkr/creating-a-virtual-machine-in-azure-a-step-by-step-guide-for-beginners-387f

Khalid, T. (2023, January 2). What is Azure ExpressRoute and Why You Should Use It. Geekflare. https://geekflare.com/azure-expressroute/#:~:text=Azure%20ExpressRoute%20is%20a%20service%20offered%20within%20Azure,towards%20non-public%20connections%2C%20ExpressRoute%20primarily%20promises%20greater%20security.

D. (2023, September 20). Azure ExpressRoute Overview: Connect over a private connection. Microsoft Learn. https://learn.microsoft.com/en-us/azure/expressroute/expressroute-introduction

D. (n.d.). ExpressRoute documentation. Microsoft Learn. https://learn.microsoft.com/en-us/azure/expressroute/

D. (2023, November 6). Connectivity providers and locations for Azure ExpressRoute. Microsoft Learn. https://learn.microsoft.com/en-us/azure/expressroute/expressroute-locations

M. M. (2023, September 15). What is Azure Load Balancer? - Azure Load Balancer. Microsoft Learn. https://learn.microsoft.com/en-us/azure/load-balancer/load-balancer-overview

M. (n.d.). Load-balancing options - Azure Architecture Center. Microsoft Learn. https://learn.microsoft.com/en-us/azure/architecture/guide/technology-choices/load-balancing-overview