AD Domain Migration and Infrastructure Realignment

Project Report

by Rob Das

DTEK Consulting Services Ltd.
Alberta, Canada

January 2025 – July 2025

This project report documents the Active Directory domain migration and infrastructure realignment project. The target audience is: sys admins.

1. Executive Summary

This project was initiated in response to an Active Directory domain-level disruption that triggered a formal review of the underlying infrastructure. The disruption exposed the potential single point of failure posed by the original primary domain controller, a Windows Server 2012 domain controller for the company's Active Directory domain. An attempted failover to the secondary domain controller revealed that it was non-functional, likely due to extensive legacy data corruption and aging replication metadata.

Given the instability and age of the domain environment, a decision was made to transition away from the original domain entirely, which led to the creation of a new, cleanly deployed Active Directory domain built using best practices and updated Windows Server systems. The project entailed building new domain controllers, rearchitecting infrastructure roles, migrating workstations and services, centralizing windows updates, and establishing a repeatable, standards-based configuration for long-term stability.

The result is a more reliable, secure, and maintainable domain environment that removes legacy dependencies and lays a foundation for future scalability.

Note: for security reasons since this document is published online, some of the internal domain and computer names have been changed.

TABLE OF CONTENTS

2. Project Overview

The domain migration project began after a Windows Update maintenance cycle caused the original primary domain controller for the company's internal Active Directory domain, *dtek.internal*, to temporarily go offline. Despite the presence of a secondary domain controller, domain services did not fail over properly, resulting in a temporary domain-wide outage. Extensive troubleshooting revealed that replication between the controllers had long been broken, and the stale replication metadata within the domain's internal databases made recovery infeasible. the original secondary domain controller was demoted and removed from the domain, exposing the original primary domain controller as a potential single point of failure.

A temporary remediation effort was undertaken to stabilize the original primary domain controller so that core authentication services could continue functioning while a new environment was planned. During this period, attempts to promote a new secondary domain controller into the existing domain failed due to persistent data integrity issues.

At that point, the decision was made to build a brand-new domain, *d1.internal*, from the ground up, with clean Active Directory configuration, hardened DNS, and improved operational practices. A pair of domain controllers (*dc-01* and *dc-02*) were deployed on updated Windows Server systems hosted on dedicated Hyper-V hypervisors. Existing systems were assessed for migration, and a staged migration plan was developed to move production systems either into the new domain or into standalone WORKGROUP configurations, depending on their role.

The project's scope of work was scaled up to include a significant infrastructure realignment:

- Replaced multi-purpose physical servers with role-dedicated hypervisors.
- Created a new virtualized file server.
- Migrated user profiles and application services.
- Established operational standards and checklists for long-term maintainability and disaster recovery.
- Introduced centralized Windows Update management to streamline Windows patching and reduce admin costs.

With the completion of the migration and decommissioning of the original primary domain controller, the company, DTEK Consulting Services Ltd., now operates on a more resilient and standards-compliant foundation.

3. Infrastructure Summary

This section provides a summary of the core infrastructure elements that were implemented, replaced, or restructured as part of the domain migration project. It reflects the current state of the company's IT environment following the completion of the transition from dtek.internal to d1.internal and supporting infrastructure realignment.

3.1 Active Directory and Domain Controllers

The new d1.internal domain is served by two Windows Server 2019 virtual machine domain controllers hosted on separate Hyper-V hypervisors. These domain controllers were configured from scratch using Microsoft-recommended defaults, with replication configured and verified

- dc-01: Primary domain controller hosted on Hyper-V server sirius.
- dc-02: Secondary domain controller hosted on Hyper-V server titan.

3.2 Hyper-V Hosts

The company's virtualization infrastructure was reorganized to move away from multipurpose physical servers. All critical workloads were moved to virtual machines hosted on dedicated Hyper-V servers:

- lunar (Windows Server 2016, retained due to long-standing stability)
- europa (Windows Server 2019)
- titan (Windows Server 2019)
- sirius (Windows Server 2019)

Each hypervisor is configured as a standalone host in WORKGROUP mode and does not participate in domain services.

3.3 File Server Services

The file server infrastructure was reorganized to simplify the role of Hyper-V hosts and establish clear separation of responsibilities. Previously, europa served both as a Hyper-V host and as a file server. A new virtual machine named file-01, running Windows Server 2016, was created to assume all file server responsibilities previously handled by europa.

- file-01 is now the second file server in the organization.
- It is hosted on europa and uses a dedicated VHDX volume backed by local storage.

- File shares were migrated from europa's physical F: drive to file-01, and a new access model based on domain security groups was implemented to follow the principle of least privilege.

In parallel, significant cleanup efforts were made on an existing file server named lunar-file. While structural improvements were implemented, such as removing redundant shares and standardizing folder layout, additional work is still planned to bring its permissions model in line with the standards used on file-01.

3.4 Domain Membership

Each computer in the organization was reviewed for its function, and then reassigned to one of two network roles:

- domain members: systems requiring Active Directory services were joined to d1.internal
- workgroup systems: systems with no dependency on domain services remained or were moved to WORKGROUP

This structure ensures minimal reliance on domain services where unnecessary, reducing risk and simplifying future disaster recovery.

3.5 Legacy System Handling

Legacy systems to support dtek.internal were retired, demoted, or repurposed:

- the original primary domain controller was demoted and removed from Active Directory.
- the original secondary domain controller was decommissioned earlier in the project.
- Systems dependent on dtek.internal were reviewed and migrated case-by-case.

3.6 Database Servers

Two database servers are maintained within the updated infrastructure to support both production and development workloads:

- lunar-sql: A SQL Server 2014 system hosting databases used by multiple applications. It was migrated from the dtek.internal domain to d1.internal as part of the project.
- w16-cosmosdb: A Windows Server 2016 virtual machine that runs the Azure Cosmos DB Emulator for on-premises development and testing.

3.7 WSUS Server

A dedicated Windows Server Update Services (WSUS) server was deployed to centralize Windows update distribution within the d1.internal domain.

- WSUS2, a Windows Server 2016 powered WSUS server, serves as a local Windows update patch repository management system.
- WSUS reduces bandwidth usage and streamlines the monthly Windows update cycles by eliminating the need for systems to download Windows updates individually.

3.8 Tools and Utilities Used

The following tools were instrumental throughout the planning, execution, and validation phases of the project:

- Active Directory Domain Services (AD DS)
- DNS Manager
- Server Manager
- Hyper-V Manager
- PowerShell and Command Prompt
- Remote Desktop Connection (RDP)
- Visual Studio – for debugging web deployments and TFS integration
- IIS Management Console – for website configuration and bindings
- SQL Server Management Studio (SSMS) – for validating SQL connectivity and security
- Windows Server Update Services Console (WSUS) – for config, sync, update approval
- Local Group Policy Editor (gpedit.msc) – for configuring WSUS client behavior
- ChatGPT – for architectural planning, PowerShell scripting, troubleshooting

4. Domain Architecture

This section describes the Active Directory domain architecture.

4.1 Purpose and design of dc-01 and dc-02

The domain d1.internal is hosted by two domain controllers, both running Windows Server 2019 edition:

- dc-02 is hosted on titan
- dc-01 is hosted on sirius

All four Hyper-V hosts in the environment (lunar, europa, titan, sirius) operate independently and are treated as equal peers. Domain controller placement across different physical hosts was deliberately chosen to ensure domain availability in the event of hardware failure.

Both domain controllers provide:

- Active Directory Domain Services (AD DS)
- DNS Server
- Global Catalog

Checkpoints are explicitly avoided for both dc-01 and dc-02. On 2025-05-07, both domain controller VMs were shut down and exported to form a backup named:

```
d1-domain-controllers_2025-05-07.zip
```

4.2 Active Directory domain design and OU structure

The domain was created from scratch using the default Server Manager GUI workflow. No objects, configurations, or policies were imported from the legacy domain dtek.internal.

As of deployment, the directory contains only the default containers automatically created by Windows:

- Builtin
- Computers
- Domain Controllers
- ForeignSecurityPrincipals
- Managed Service Accounts
- Users
- *No Custom Organizational Units (OUs) have been created.*

4.3 DNS and DHCP configuration summary

DNS is AD-integrated and redundantly hosted on both domain controllers (dc-01 and dc-02).

- Primary zone: d1.internal
- Replication scope: To all domain controllers in the domain
- Reverse lookup zone for the 192.168.11.0/24 network was created as part of initial setup

DHCP is not managed by Windows Server. It continues to be provided by the network router at 192.168.11.1, which has not been modified and continues to serve leases to the entire subnet. This decision was made to avoid disrupting long-established network infrastructure.

4.4 Global catalog and FSMO role assignments

Global Catalog is a complete list of objects in the domain, and FSMO roles, or Flexible Single Master Operations roles, dictate which domain controller is responsible for specific types of updates and changes, preventing replication conflicts and ensuring the directory operates correctly. Both domain controllers (dc-01 and dc-02) are Global Catalog servers, as verified in Active Directory Sites and Services.

FSMO Role Holders are shown below (confirmed via “netdom query fsmo”):

- Schema Master: dc-01.d1.internal
- Domain Naming Master: dc-01.d1.internal
- PDC Emulator: dc-01.d1.internal
- RID Master: dc-01.d1.internal
- Infrastructure Master: dc-01.d1.internal

All FSMO roles are currently held by dc-01.

4.5 Group Policy overview

No custom Group Policies have been created or modified at this time. The following default GPOs exist:

- Default Domain Policy
- Default Domain Controllers Policy

These GPOs apply domain baseline settings. They have not been edited or extended.

5. System Configuration and Services

This section describes how the various systems are configured and the services they provide.

5.1 File Server Transition and Design Intent

Historically, europa, a voluptuous server blessed with a pair of ample hard drives, served dual roles as both a Hyper-V host and a file server. To simplify its responsibilities and improve manageability, file services were migrated to a new, dedicated file server virtual machine named file-01. File-01 was created specifically to offload user and departmental file shares from europa, allowing europa to operate solely as a Hyper-V hypervisor going forward. Europa was then chosen to host file-01, because of europa's ample hard drive capacity.

In parallel, substantial cleanup was performed on lunar-file, an older file server, to consolidate, remove redundant shares, and improve organizational clarity. Although more work remains—particularly in aligning lunar-file share permissions with the least-privilege model established on file-01—this effort marked a key step toward standardized and streamlined file service management.

Both file-01 and lunar-file are Windows Server systems joined to the d1.internal domain, managed via Server Manager and AD-integrated access controls.

5.2 Role Separation: Repurposing Multi-Role Servers into Dedicated Hypervisors

Early in the environment's lifecycle, some systems were configured with multiple roles. Over time, these systems were repurposed to isolate hypervisor responsibilities from other server roles. Specifically:

- europa was reconfigured to remove file services and act solely as a Hyper-V host.
- lunar was converted from one edition of Windows Server 2016 to another to support Hyper-V features and licensing. It remains on 2016 for stability reasons.
- sirius and titan were built as Windows Server 2019 installations from the start, dedicated to Hyper-V hosting only.

This role separation ensures better isolation, performance, and ease of maintenance.

5.3 Local Admin vs Domain Access Control

Domain-joined servers and workstations now rely on domain-based user authentication via d1.internal. Where possible, local administrator access is restricted to domain-level admin groups (D1\Administrators), rather than relying on per-machine local accounts. This approach simplifies credential management and improves auditing consistency.

Exceptions exist on standalone systems (such as those in WORKGROUP configurations) where local credentials are required. These systems are limited in scope and isolated by design.

5.4 Security Hardening and Least-Privilege Improvements

Security configurations follow a pragmatic model of high trust among internal staff. By default, all internal users are granted full access to shared file resources, consistent with company policy. File servers such as file-01 and lunar-file are configured so that most shares use “Everyone / Full Control” to streamline collaboration and minimize support overhead.

Exceptions are made for a small number of shares containing sensitive information—typically related to finance or system administration. These shares are explicitly secured using named domain accounts and group-based permissions to limit access.

Domain-based authentication is enforced across all servers and shares, ensuring only authenticated users can access internal resources. This model balances ease of access with reasonable protection of sensitive information.

6. Migration Details

This section provides a detailed overview of the migration process, including the final placement of computers, rationale for domain membership decisions, and configuration of non-standard setups.

6.1 Final Placement of Computers

The table below summarizes the current placement of all known computers across the organization. Systems were either migrated to the d1.internal domain, placed in a WORKGROUP, or in special cases left unjoined due to technical constraints.

Domain / Workgroup	Role	Number of Computers
d1.internal	App Server	3
	Batch Jobs on Task Scheduler	1
	Database Server (SQL-Server)	1
	Developer Workstation	1
	Documentation Workstation	1
	File Server	1
	General Purpose Workstation	3
	Primary DC	1
	Secondary DC	1
	Web Hosting - Dev/Test	1
	Web Hosting - Production	1

Domain / Workgroup	Role	Number of Computers
WORKGROUP	Database Server (Cosmos DB)	1
	Decommissioned DC	1
	Developer Workstation	2
	General Purpose Server	3
	General Purpose Workstation	6
	Hyper-V Host	4
mylab.local	Laboratory Computer	1
n/a	Linux Computer	1

6.2 Rationale for Domain Membership Decisions

Most systems performing core infrastructure functions (e.g., file sharing, application hosting, SQL Server) were joined to the d1.internal Active Directory domain for centralized identity management and policy enforcement.

However, several systems were intentionally not joined to the domain due to the following factors:

- Hyper-V Hosts (sirius, titan, lunar, europa) were excluded from domain membership to avoid dependency on AD availability and to improve isolation for disaster recovery purposes.
- Unsupported Operating Systems such as Windows 10/11 Home Edition and Linux do not support AD domain joining.
- Application Incompatibility: The CosmosDB emulator on w16-cosmosdb is known to behave unpredictably in domain environments, and was deliberately left unjoined.
- Unassigned / Spare Systems: Some computers have not yet been assigned for production use, so domain decisions have been deferred.

6.3 Configuration of Non-Standard Setups

Some systems deviate from standard deployment due to historical decisions or unique workloads:

- lunar-sql and lunar-tfs retain legacy TFS + SQL Server roles with domain-linked authentication still being cleaned up.
- The web servers (eu-web, webserv-test) required careful DNS and authentication coordination with SQL-based services on lunar-sql using Windows-integrated credentials.
- A few legacy computers had dual identities (e.g., previously joined to dtek.internal, then rejoined to d1.internal) which occasionally caused duplicate SPNs or SID issues during migration. These were manually resolved.

7. Migration of Interdependent Legacy Application Systems

One of the most intricate and time-consuming parts of the domain migration project was the coordinated transfer of five legacy systems that collectively supported a multi-tier, front-end web application and its supporting infrastructure. These systems had to be migrated together due to tight operational coupling, shared authentication requirements, and historical design constraints.

7.1 Systems Involved

The following systems were involved:

- devbox-1: Developer workstation for legacy Visual Studio projects
- eu-web: Production front-end web server (MVC application)
- webserv-test: Staging/test web server (mirrors production setup)
- lunar-sql: Database server hosting Webserv-dev, Webserv-test, and Webserv-prod (SQL Server 2014)
- lunar-tfs: Source control server running TFS (Team Foundation Server) with its own local SQL Server instance

7.2 Domain Authentication Dependencies

The application stack relies on Windows-integrated authentication for both web-server-to-database communication and user-level access controls. This meant the migration from dtek.internal to d1.internal could not break or disrupt:

- Application pool identities used in IIS
- SQL Server logins mapped to Active Directory domain accounts
- Security roles in individual SQL databases
- External access via leased domain name and its static IP

All interdependencies had to be preserved or recreated during the transition to d1.internal, which required:

- Adding new SQL Server logins for the domain-joined machine accounts: D1\EU-WEB$, D1\WEBSERV-TEST$, and D1\DEVBOX-1$
- Ensuring corresponding roles and permissions were reassigned to these new accounts in the Webserv-dev, Webserv-test, and Webserv-prod databases
- Reconfiguring IIS application pools and web.config files

- Carefully staged system shutdowns and reboots to coordinate AD changes

7.3 TFS Source Control Migration

The lunar-tfs server hosted a legacy TFS environment backed by its own SQL Server instance. The migration involved:

- Domain join to d1.internal without breaking TFS service identity bindings
- Verifying that TFS could still access its databases and authenticate users
- Confirming continued access to project history, check-ins, and team settings

7.4 Testing and Validation

Significant testing was conducted to confirm:

- Database connectivity and application function post-migration
- Web application online availability through leased public domain name
- Correct mapping of SQL logins to domain accounts
- TFS service availability and user access

This combined migration was one of the most delicate operations in the project due to the number of moving parts, the use of Windows authentication across multiple tiers, and the live production nature of eu-web.

8. Implementing WSUS for Improved Patch Management

Windows Server Update Services (WSUS), a software update management system, allows sys admins to manage deployment of Microsoft product updates to client computers on a network.

8.1 Purpose and Overview

To modernize monthly patching and reduce overhead, a centralized update system was deployed using Windows Server Update Services (WSUS). Updates are now downloaded once and distributed internally, cutting bandwidth use, standardizing patch levels, and improving control over deployments.

8.2 Proof-of-Concept

To validate the WSUS deployment, a proof-of-concept was conducted using a freshly installed Windows Server 2016 client joined to the domain. The test confirmed that approved updates were successfully delivered from the internal WSUS server without reaching out to Microsoft's public update infrastructure. Updates were manually approved in the WSUS console and applied to the client, verifying end-to-end functionality.

Client Configuration

To maintain decoupling from domain-level group policies, local group policy was used to configure the client's update settings:

- "Specify intranet Microsoft update service location"
- "Configure Automatic Updates"
- "Do not connect to any Windows Update Internet locations"

Troubleshooting and Resolution

Initial registration and update detection issues were traced to a memory-limited IIS application pool on the WSUS server. After adjusting the memory settings, normal operation resumed.

Verification

- Updates were approved manually in the WSUS console and assigned to the appropriate computer group, either Win2016 or All Computers.
- After successful update installation, the update log confirmed WSUS was the source.

This confirmed that server-side targeting and WSUS communication were functioning.

Outcome:

- WSUS communication via local Group Policy is functional.
- Updates can be targeted to specific OS versions using WSUS computer groups.
- Updates are properly delivered from the internal server without contacting Microsoft.

8.3 WSUS Server Infrastructure

The WSUS server wsus2.d1.internal, deployed for centralized update management, runs Windows Server 2016 Standard, is joined to d1.internal, and is hosted on the Sirius Hyper-V system. It was installed using Server Manager and is hosted under IIS. No SSL certificate has been configured at this stage, so the system currently uses HTTP for client communication.

The system was tested using manual update synchronization only, and no recurring sync schedule has been configured. Future enhancements may include scheduled syncs and expanded update categories.

Key configuration details:

- Hostname: wsus2.d1.internal
- OS: Windows Server 2016 Standard
- Domain: d1.internal
- Virtualization: Hosted on Sirius (Hyper-V)
- Database: Windows Internal Database (WID)
- Update Storage: E:\WSUS
- IIS Ports: 8530 (HTTP), 8531 (HTTPS)
- Targeting Mode: Server-Side Targeting

This WSUS deployment uses *server-side targeting*, where all clients appear in the Unassigned Computers group and are manually moved to WSUS groups (such as Win2016) via the WSUS console. Server-side targeting is the default WSUS configuration, found under:

```
WSUS Console > Options > Computers > Use the Update Services console
```

If desired, client-side targeting can be enabled instead by switching the above option and applying the "Enable client-side targeting" GPO setting on clients.

The system was designed with recovery and maintenance in mind. Because it uses WID and stores all updates locally, the virtual machine can be exported and restored easily using Hyper-V Manager. No external database or file share dependencies exist, making the configuration highly portable and well-suited for disaster recovery planning.

8.4 Non-Default Configuration

To optimize WSUS for its virtualized environment and ensure scalability, several settings were adjusted from their defaults.

Hyper-V VM Settings:

- Dynamic Memory: 2048–8192 MB

- Processor: 1 virtual CPU

This balances performance with the resource constraints of the Sirius host.

IIS Configuration:

- WsusPool Private Memory Limit: 0 (unlimited)
- Connection Timeout: 600 seconds

These prevent console hangs and allow longer-running operations during sync and approval.

WSUS Targeting and Grouping:

- Server-side targeting is enabled (clients assigned manually via WSUS console).
- Groups are organized by OS version, beginning with Windows Server 2016.

This supports phased rollouts and OS-specific update approvals.

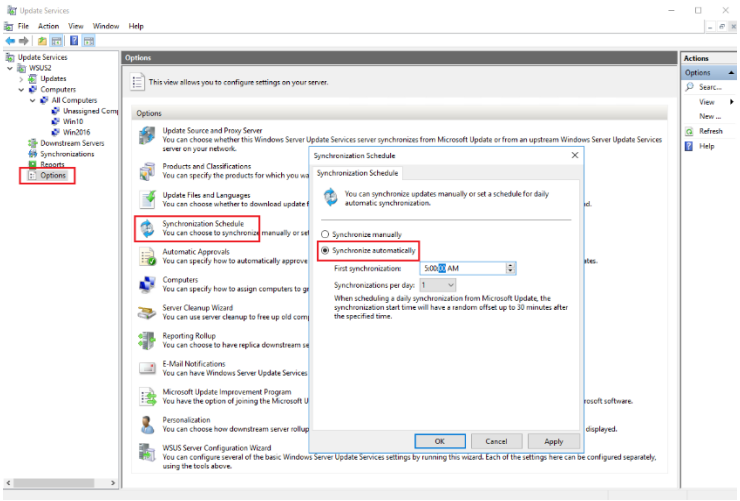
Enable WSUS Synchronization Schedule

Open

```
WSUS Console → Options → Synchronization Schedule
```

Select

- Synchronize automatically
- Set to run daily at a fixed time



WSUS will automatically fetch new updates from Microsoft every morning.

Special Case – Webserver Restart Behavior:

Two WSUS clients, eu-web and webserv-test, serve websites and exhibits near-continuous IIS activity, even during off-hours. Because Windows Server 2016 requires extended idle time to trigger automatic restarts for updates that require them, eu-web and webserv-test do not meet

this condition. Logs and active w3wp.exe processes confirm consistent traffic. This limitation only affects updates that require a system restart, which are infrequent. As such, occasional manual reboots during maintenance windows are sufficient and represent a minimal administrative overhead—especially when compared to the previous manual update process without WSUS.

8.5 Steps for Adding Additional Clients

To onboard new clients into the WSUS infrastructure, follow these steps to ensure proper registration and update delivery. This process ensures correct WSUS registration and fully automated update delivery using a dual mechanism: Local Group Policy + Scheduled scan script. This assumes server-side targeting is used and WSUS-related Group Policy is applied locally on each client.

Step 1 – Join the Domain

Ensure the client system is joined to the d1.internal Active Directory domain.

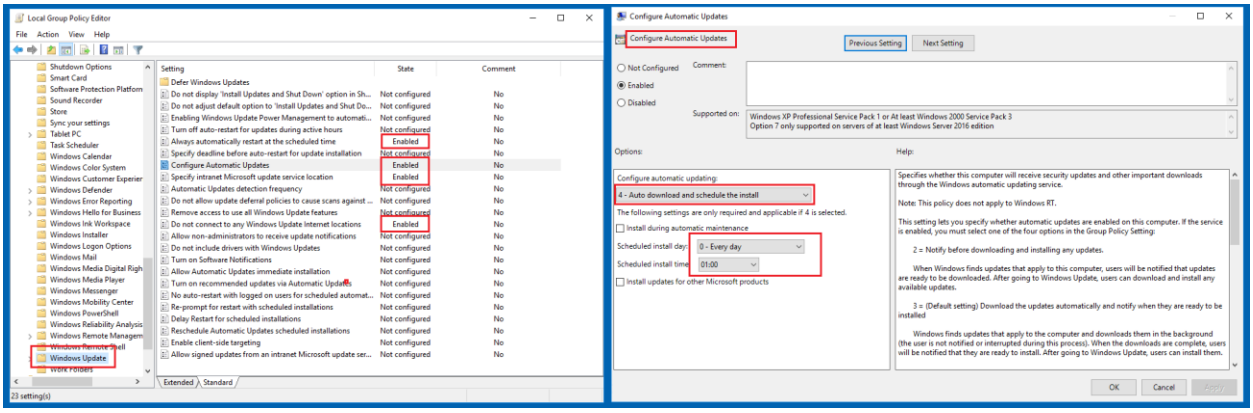
Step 2 – Apply Local Group Policy

On the client, open the Local Group Policy Editor (gpedit.msc)

```
Local Computer Policy > Computer Configuration > Administrative Templates > Windows Components > Windows Updates
```

and set:

- Enable "Specify intranet Microsoft update service location"
    - Detection server: http://wsus2.d1.internal:8530
    - Statistics server: http://wsus2.d1.internal:8530
    - Leave alternate download server not configured.
- Enable "Configure Automatic Updates"
    - "4 - Auto download and schedule the install"
    - Scheduled install day: "daily"
    - Scheduled install time: "1:00 AM"
- Enable: "Do not connect to any Windows Update Internet locations"
  this prevents fallback to Microsoft online
- Enable: "Always automatically restart at the scheduled time". This enforces a restart 15 minutes after installation completes, even if users are logged in.



Apply the policy using:

```
gpupdate /force
```

Note: Server-side targeting is used; do not configure "Enable client-side targeting".

Step 3 – Trigger WSUS Registration

On the client, go to Settings > Windows Update and click Check for updates. This will initiate contact with wsus2 and register the client in WSUS under Unassigned Computers.

Wait for the client to appear.

Server-Side Steps

- In the WSUS console, go to Computers > Unassigned Computers and verify the new client appears.
- Ensure the update(s) intended for the new group are approved.

Note: A client in "Unassigned Computers" will not receive approved updates unless explicitly moved to a targeted group.

Step 4 - Automate Update Detection via Scheduled Script

Create a scheduled task that runs the following PowerShell script once per day at 06:00 PM:

Script Location: \\lunar-file\share\Code\forcewuscan.ps1

Script Contents:

```
Start-Service wuauserv
Start-Service bits
UsoClient StartScan
```

Scheduled Task Configuration:

- Task name: Force Windows Update Scan
- Program/script: powershell.exe

- Arguments:

```
-ExecutionPolicy Bypass -File "\\lunar-file\share\Code\forcewuscan.ps1"
```

Note: applying local group policy and the scheduled script forcewuscan.ps1 work together. Update automation requires two tightly coupled parts:

- A PowerShell-based script (forcewuscan.ps1) to start services and initiate detection
- A Group Policy configuration to schedule and authorize installation

If either part is missing, automation fails:

- forcewuscan.ps1 without policy => detection only, no install
- Policy without forcewuscan.ps1 => no updates detected, install does nothing

The current production schedule uses:

- A daily update detection task at 6:00 PM, triggered by Task Scheduler
- A Group Policy installation schedule set for 1:00 AM

This staggered timing ensures detection occurs while systems are likely idle (post-business hours), and installations/reboots are completed overnight.

Step 5 – Assign to WSUS Computer Group

In the WSUS console, move the new client to the appropriate group (e.g., Windows Server 2016) to control which updates it receives.

Step 6 – Monitor Update Status

To confirm update detection and installation:

- Check the client's status in the WSUS console.
- Check Update History to verify that approved updates are applied
- On the client, inspect: C:\Windows\SoftwareDistribution\ReportingEvents.log for scan/install logs
- On the client, run

```
Get-WindowsUpdateLog
```

This creates a readable WindowsUpdate.log file on the desktop. Confirm that update sources reference wsus2.d1.internal.

Note: If the client is already fully patched and you wish to validate WSUS-based update delivery, you may uninstall the most recent monthly cumulative update then restart and re-scan. This forces WSUS to reoffer the update, allowing for end-to-end testing.

8.6 WSUS Operational Workflow

This section outlines the standard operational process for approving and deploying updates using WSUS. These procedures form the baseline for ongoing patch management in our environment.

WSUS Computer Groups

Client systems are organized in WSUS using server-side targeting, with each supported operating system assigned to its own WSUS computer group. For example:

- Win2016 – Windows Server 2016 systems
- (Planned) Win10, Win2019, etc. – for other OS versions

This structure allows updates to be approved either for specific operating systems or for all computers, depending on the nature of the update.

Note: Clients initially appear in the Unassigned Computers group. An administrator must manually move them into the appropriate OS group in the WSUS console.

Group assignment is critical for update approval.

Update Approval Process

Update approvals are performed using the WSUS console. Updates can be approved for:

- Individual computer groups (e.g., Win2016)
- All Computers group (for critical or universally applicable updates)

Approval convention:

- OS-specific updates (e.g., Windows 10 cumulative updates) are approved only for that OS group.
- Universal updates (e.g., Windows Malicious Software Removal Tool) approved for all computers.

Updates are not auto-approved. Each month's updates (typically released on Patch Tuesday) should be reviewed and approved by an administrator.

Client Update Retrieval

Once an update has been approved for the appropriate group, client systems will detect and download the update automatically based on their local Group Policy settings and Windows Update's internal schedule. In normal circumstances, this process requires no manual intervention.

All WSUS settings are applied via local Group Policy, not domain-level GPOs. This provides decoupling from domain controllers and simplifies troubleshooting.

Manual Overrides (if needed)

In rare cases where a client does not appear to be updating or checking in:

- A manual sync can be triggered using wuauclt /detectnow and wuauclt /reportnow
- Review the client's WindowsUpdate.log using Get-WindowsUpdateLog
- Confirm the client is in the correct WSUS group and the update has been approved for that group

Server Synchronization

During the proof-of-concept, update synchronization with Microsoft Update was triggered manually via the WSUS console. This pulled update metadata and binaries.

Going forward: configure automatic daily synchronization under `Options > Synchronization Schedule` to keep WSUS up to date without manual action.

Client Polling Interval

Clients configured via local Group Policy check for updates every 22 hours, with slight randomization per device to avoid simultaneous polling.

Once an update is approved and synchronized, no manual client action is required for detection.

Recommended Monthly Schedule

The following schedule is used to coordinate Windows updates in production:

- Patch Tuesday: Microsoft releases updates at 10:00 AM (second Tuesday each month)
- WSUS Synchronization: Schedule to run at 11:00 AM to retrieve new content
- Update Approval: Admins should review and approve updates between 1:00 PM and 4:00 PM on Patch Tuesdays
- Daily Client Scan: Automated via scheduled script at 6:00 PM
- Update Installation: Triggered by WSUS policy at 1:00 AM

This schedule has proven reliable and balances timely patching with minimal disruption.

8.7 Troubleshooting Guide

This section outlines known issues encountered during the WSUS rollout and their resolutions.

Issue: Client Shows "You're up to date" but No Updates Installed

Observed on: webserv-test.d1.internal

Symptoms:

- Windows Update (Settings app) reports "You're up to date"
- Control Panel > View Installed Updates shows no recent cumulative updates (e.g., KB5061010)
- WSUS console displays status:
    - "Not yet reported"
    - Tooltip: "This computer has not yet contacted the server"

Likely Cause:

- Windows Update client service (wuauserv) was not started after resetting the SoftwareDistribution folder.
- The client did not contact WSUS due to pending service initialization.

Resolution:

Reboot and manually start or restart Windows Update services and initiate a scan:

```
net stop wuauserv
net start wuauserv
Start-Service BITS
UsoClient StartScan
```

Once these steps are completed, the client typically appears in the correct computer group in WSUS, and pending updates will become visible.

Notes:

- This fix was verified successful on webserv-test.
- It may be necessary after manually clearing C:\Windows\SoftwareDistribution.

Special Case – WSUS Server as Client:

If the WSUS server itself is configured as a client:

- Use its own DNS name in the WSUS URL (e.g. http://wsus2.d1.internal:8530)
- Avoid using localhost or 127.0.0.1
- Follow the same steps as above

After uninstalling an update for test purposes, a reboot and manual scan may be required before WSUS detects the update as needed.

Troubleshooting issues with WSUS Client Registration and Communication

If a client fails to appear in WSUS or does not receive updates, use the checklist below.

Confirm Group Policy Application

On the client, run:

```
gpresult /r
```

If a client fails to appear in WSUS or does not receive updates, use the checklist below.

Confirm WSUS Registry Settings

```
HKEY_LOCAL_MACHINE\Software\Policies\Microsoft\Windows\WindowsUpdate
```

Expected values:

- WUServer = http://wsus2.d1.internal:8530
- WUStatusServer = http://wsus2.d1.internal:8530

If missing, verify local Group Policy and re-apply with `gpupdate /force`.

Trigger WSUS Communication

If the client doesn't check in automatically, try the following on the client

```
wuauclt /detectnow
wuauclt /reportnow
net stop wuauserv
net start wuauserv
```

Note: wuauclt is deprecated on Windows 10/11 but still works on Server 2016. Do not use usoclient unless troubleshooting Windows 10/11 specifically.

Check WSUS Server Health

- Verify WsusPool is running in IIS. If not, verify non-default memory settings (see 8.4).
- Do a manual synchronization in WSUS console to confirm the server can pull updates.
- If client remains under Unassigned Computers, manually assign it to the correct group.

WindowsUpdate.log Review

On the client, run

```
Get-WindowsUpdateLog
```

This generates a log file on the Desktop. Search for lines referencing wsus2.d1.internal to confirm the correct server is being contacted.

Issue: Scheduled Tasks Fail to Trigger After VM Startup

Observed on:

eu-web.d1.internal

Symptoms:

- Scheduled task appears in Task Scheduler history with status “Missed task start rejected”

Likely Cause:

- The VM was not fully initialized when the task was scheduled to run

Resolution:

- Reschedule task at least 15 minutes after VM startup
- Enable “Run task as soon as possible after a scheduled start is missed” in task settings

Notes:

- This scenario is common on slower hypervisors such as Europa

Issue: WSUS Scan Fails Despite Script Execution

Observed on:

eu-web.d1.internal

Symptoms:

- forcewuscan.ps1 logs successful start/finish
- WindowsUpdate.log shows timeout or 0x8024401c errors

Likely Cause:

- WSUS server was offline or non-responsive during scan attempt

Resolution:

- Confirm WSUS availability via browser or ping before scan time
- Restart IIS and verify WsusPool is running

Notes:

- Setting automatic WSUS sync at 11:00 AM helps ensure server readiness after Patch Tuesday

Issue: Updates Install but Auto-Restart Never Occurs

Observed on:

eu-web and webserv-test

Symptoms:

- KB5062560 listed as “Requires a restart to finish installing”
- Uptime exceeds 24 hours with no reboot Likely Cause:
- Active IIS services prevent Windows from reaching idle threshold for automatic restart

Resolution:

- Manual restart required on IIS servers with continuous traffic

Notes:

- Documented as an expected exception for servers with externally accessible websites (see 8.4)

9. Upgrading Windows 10 Systems to Windows 11

9.1 Overview

- Rationale for Windows 11 migration (Windows 10 EOL, infrastructure modernization)
- Summary of Microsoft's free upgrade policy and hardware requirements
- Scope change approval and how it aligns with the project's goals

9.2 System Categorization

9.2.1 Unused Virtual Machines

- Definition and identification
- Plan: recreate from scratch with Windows 10 ISO, then upgrade

9.2.2 In-use Virtual Machines

- Constraints and risks
- Hyper-V settings adjustments for Windows 11 compatibility
- In-place registry bypass process

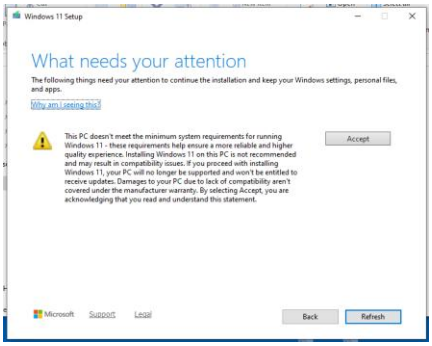
9.2.3 Physical Machines

- Virtualization strategy using Windows Backup + VHDX conversion
- Reconfiguration and upgrade process inside Hyper-V

9.3 Upgrade Readiness and Prerequisite Checks

- VM configuration (TPM, Secure Boot, UEFI, vCPU, RAM, disk)
- Tools for checking compatibility (e.g., PC Health Check, manual review)
- Lessons from win10pro-1 regarding TPM/CPU bypass
- Apply Registry Bypass for Unsupported CPU/TPM
    - Open Regedit and navigate to: HKEY_LOCAL_MACHINE\SYSTEM\Setup\MoSetup
    - If MoSetup doesn't exist, create it.
    - Add a new DWORD (32-bit) Value:
        - Name: AllowUpgradesWithUnsupportedTPMOrCPU
        - Value: 1

- This registry setting will cause the following warning during the upgrade.



- Accept and continue.

9.4 Upgrade Paths and Methods

9.4.1 ISO-based upgrade vs. Installation Assistant

Windows 11 can be installed using the Installation Assistant or an ISO file. The Installation Assistant requires internet access and enforces strict hardware checks. Using an ISO allows manual setup, offline upgrades, and easier bypass of hardware requirements.

9.4.2 Why ISO was chosen

We chose the ISO method for greater control and to bypass CPU and TPM checks on Hyper-V VMs. It enabled consistent upgrades without unexpected failures and allowed reuse of a single ISO across multiple systems.

9.4.3 Steps to Perform Upgrade in Hyper-V

This section outlines the complete steps used to upgrade Windows 10 virtual machines to Windows 11 inside Hyper-V using an ISO-based upgrade method. This was successfully tested on VMs w10p-3 and win10pro-1, both of which had hardware not meeting the Windows 11 requirements (e.g., unsupported CPU, no TPM 2.0).

Preparation – Before Upgrade

1. Download Windows 11 ISO
   - Download the latest Windows 11 ISO from the official Microsoft page: https://www.microsoft.com/en-ca/software-download/windows11
2. Attach the ISO to the VM
   1. In Hyper-V Manager:
      a. Right-click the VM > Settings > DVD Drive
      b. Choose Image file (.iso) and browse to the downloaded ISO
3. Ensure VM Has Enough Resources

- Minimum 2 cores, 4 GB RAM recommended
- Check that Secure Boot is enabled (if supported), but TPM is not required when using registry bypass
- See Windows 11 minimum requirements section below

4. Ensure Free Disk Space
    - Minimum 27 GB free on C: drive to allow Windows setup to expand and work correctly

Registry Bypass – Skip Hardware Checks

1. Launch the VM and log in as an administrator
2. Open Registry Editor (regedit.exe)
3. Navigate to: HKEY_LOCAL_MACHINE\SYSTEM\Setup\MoSetup
4. Create a new DWORD (32-bit) value:
    - Name: `AllowUpgradesWithUnsupportedTPMOrCPU`
    - Value: 1
5. Close Registry Editor
    - No reboot is required

Note: This bypasses the TPM 2.0 and CPU requirements, which is for legacy or older VMs.

Windows 11 Minimum System Requirements (Official)

According to Microsoft (source: https://www.microsoft.com/en-ca/windows/windows-11-specifications), Windows 11 has these official requirements:

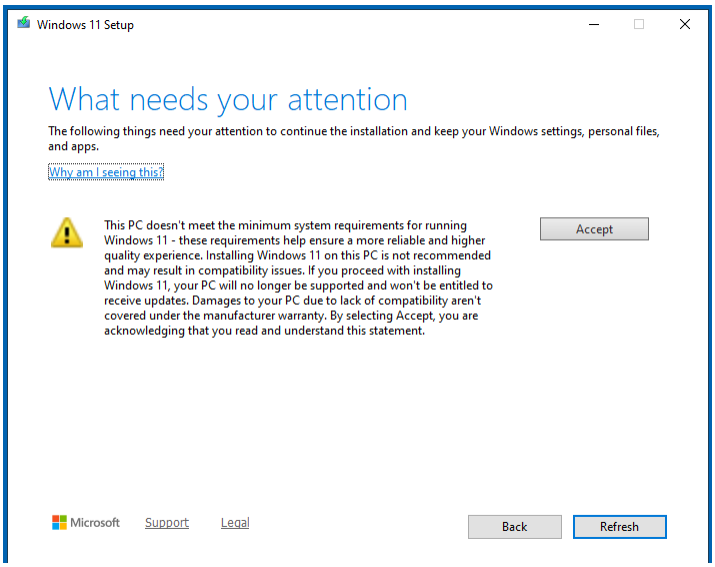
- Processor: 1 GHz or faster with 2 or more cores on a compatible 64-bit processor or System on a Chip (SoC)
- RAM: 4 GB or more
- Storage: 64 GB or larger storage device
- Firmware: UEFI, Secure Boot capable
- TPM: Trusted Platform Module (TPM) version 2.0
- Graphics card: Compatible with DirectX 12 or later with WDDM 2.0 driver
- Display: >9” with HD resolution (720p)
- Internet connection & Microsoft account: Required for setup (Home edition), strongly recommended otherwise

These were met or bypassed in Hyper-V as follows

Processor	• VMs have 2 logical processors assigned, meeting the core and speed requirement.

	• We do not require CPU compatibility checks because we use the registry bypass (`AllowUpgradesWithUnsupportedTPMOrCPU`). RAM
RAM	• We configured each VM with at least 4 GB RAM (or higher).
Storage	• Each VM has well over 64 GB virtual disk, and enough free space (>27 GB) to allow setup.
Firmware (UEFI and Secure Boot)	• Hyper-V Gen 2 VMs support UEFI and Secure Boot. • We have Secure Boot enabled for these VMs, which satisfies this requirement.
TPM 2.0	• Enable VM > Settings > Security > Trusted Platform Module. • Instead, we bypass the TPM requirement using the registry key (AllowUpgradesWithUnsupportedTPMOrCPU). via bypass
Graphics Card	• Hyper-V VMs use a basic virtual graphics adapter; while it doesn't fully meet DirectX 12 with WDDM 2.0, Windows 11 setup does not strictly block on this in practice inside a VM. • We can run Windows 11 with software rendering or basic Hyper-V display.
Display	• Virtual display is always greater than 9", typically defaults to 1024×768 or higher
Internet connection & Microsoft account	• We provide internet to the VM via Hyper-V virtual switch. • We logged in with local admin (no problem).

Note: When using the AllowUpgradesWithUnsupportedTPMOrCPU registry bypass, you may see this



Run the Windows 11 Setup from ISO

1. Open File Explorer
   - Browse to the mounted ISO (e.g., D:\setup.exe)
   - Right-click setup.exe > Run as administrator
2. Follow the Installer:
   - Accept the license terms
   - Choose Keep personal files and apps
   - Continue through the wizard until the upgrade starts
3. Installer Behavior:
   - Upgrade may take 30–60 minutes
   - Several reboots will occur
   - Be patient after the final boot—Windows Update may run in the background for 5–10 minutes before showing the desktop

Post-Upgrade Cleanup

1. Detach ISO
   - In Hyper-V Manager, disconnect the ISO from the virtual DVD drive
2. Optional: Remove the registry bypass key
   - It's not harmful to leave it, but you may remove: AllowUpgradesWithUnsupportedTPMOrCPU from HKLM\SYSTEM\Setup\MoSetup
3. Verify Upgrade
   - Open winver to confirm you're on Windows 11 Pro

Notes

- The upgrade leaves the original user accounts, programs, and data intact.
- This process does not require a domain join.
- You do not need TPM or Secure Boot, thanks to the registry bypass.

9.5 Moving Windows 11 Virtual Machines Between Hypervisors

When migrating Windows 11 Gen 2 VMs (with vTPM enabled) from one Hyper-V host to another, standard export/import operations may fail due to key protector mismatches. This is expected behavior if the VM was configured with a local key protector that’s bound to the original host. Follow these steps to move the VM and restore vTPM functionality.

1. Export the VM on the Source Hypervisor

- Shut down the VM gracefully
- Use Hyper-V Manager or PowerShell

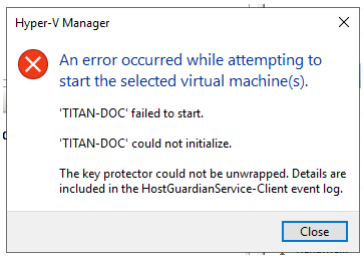
```
Export-VM -Name "VM-NAME" -Path "D:\Exports\"
```

2. Import the VM on the Destination Hypervisor

- Use Hyper-V Manager or PowerShell to import without starting

```
Import-VM -Path "D:\Exports\VM-NAME"
```

3. Launch Attempt Fails with TPM Error



This occurs because the TPM key protector cannot be decrypted on the new host.

4. Workaround: Create New VM Using Existing VHDX

- Create a new Gen 2 VM on the destination host
- Attach the .vhdx file from the exported VM as its system disk
- Do not enable TPM yet

5. Assign New Local Key Protector

- In elevated PowerShell on the destination host:

```
Set-VMKeyProtector -VMName "VM-NAME" -NewLocalKeyProtector
Enable-VMTPM -VMName "VM-NAME"
```

6. Start the VM and Verify Operation

- Boot the VM normally
- Check Device Security in Windows Settings to verify TPM presence
- Run a functional test of the app workloads

DIAGRAM SUGGESTIONS:

Diagram Concept: "Moving a Windows 11 VM with vTPM Between Hypervisors"

Layout suggestion: Divide it into three vertical panels, left-to-right flow.

Panel 1: Source Hypervisor (Before Move)

Elements:

- Host A (Source Hyper-V Host) — labeled “EUROPA” or whatever host you used.
- VM Box labeled “EU-DEV-2 (Windows 11)”

- Text/annotation:
    - "vTPM key protector bound to Host A"
    - "VM shut down and exported"

Panel 2: Transfer

Elements:

- Big horizontal arrow labeled “Export VM files (VHDX + config) ➜ Copy to Host B”

Panel 3: Target Hypervisor (After Move)

First branch (broken path):

- Host B (Target Hyper-V Host, e.g., “SIRIUS”)
- Original VM configuration (dashed box) labeled "Imported VM (vTPM error)"
- Red cross icon with annotation:
    - "Fails to start: 'The key protector could not be unwrapped.'"

Second branch (correct path):

- New VM box labeled “New VM shell (no TPM enabled initially)”
- Arrow to "Attach exported VHDX"
- Arrow to "Set new key protector & Enable vTPM" (can include short PowerShell snippet icon)
- Green check icon labeled "VM starts successfully on Host B"

Additional visual cues

- Use lock icons 🔒 to indicate vTPM key protector.
- Use green and red icons to highlight correct vs. failed paths.
- Label each stage clearly (e.g., “Export”, “Import Attempt (Fails)”, “New VM + Reattach Disk”, “Start Success”).

Caption example

"vTPM-bound Windows 11 VMs cannot be started on a new host directly after import. This diagram shows the correct recovery process using a new VM shell and key re-provisioning on the new Hyper-V host."

Recommended Visio template

"Basic Diagram" or "Basic Flowchart"

- Both are flexible and come with simple shapes (rectangles, arrows, annotations) which work well for illustrating VM migration steps.

- You won’t be forced into strict swimlanes or predefined IT icons, so you can freely lay out hosts, VMs, and steps.

Recommended stencils

- Basic Shapes — rectangles, rounded rectangles (great for VMs), circles for locks or status icons.
- Arrows — for flow and process steps.
- Callouts — for text annotations explaining what happens at each step.
- Icons (optional) — if you want, you can import simple lock 🔒 or warning ⚠️ icons as images, or use "Star and Banner" shapes for highlights.

Suggested steps in Visio

1. Draw three vertical "panels" (or sections): you can use rectangles with a subtle gray or light color fill as "background" for each panel (Source Host, Transfer, Target Host).

2. Add host shapes: use larger rectangles labeled "EUROPA" (source) and "SIRIUS" (target).

3. Add VM shape: rectangle or rounded rectangle labeled "EU-DEV-2 (Windows 11)".

Add a small lock symbol or text "vTPM key protector" inside or beside.

4. Add arrows for process steps: Export ➔ Transfer ➔ Import Attempt (failed) ➔ New VM shell ➔ Disk attach ➔ TPM commands ➔ Success.

5. Add red cross and green check icons: use shapes (cross or check mark from stencils) or just create with text and color for simplicity.

6. Use callouts: e.g., "Fails with error: 'Key protector could not be unwrapped.'", "Correct method: New VM shell + disk + new key protector."

7. Finalize labels & captions: add a caption below the diagram: "vTPM migration flow for Windows 11 VMs between Hyper-V hosts."

Alternative (if you want pre-made Hyper-V icons)

You could also start with the "Detailed Network Diagram" template if you'd like to use network/server shapes, but for this simpler host-to-host process flow, "Basic Diagram" is usually quicker and easier.

9.6 WSUS client integration (optional step for slow updaters)

9.7 Checklist: Reducing VHDX size for Windows 10 VMs

From within the guest VM

1. Disable system files that prevent shrinking
   - Disable pagefile:
     - Go to System Properties > Advanced > Performance > Settings > Advanced > Virtual memory.
     - Choose "No paging file", click Set, then OK, and reboot if required.
   - Disable hibernation:
     - Open Command Prompt as Administrator, run:

```
powercfg /hibernate off
```

   - Disable Windows Recovery Environment (WinRE):
     - Open Command Prompt as Administrator, run:

```
reagentc /disable
```

(This makes the last recovery partition removable.)

2. Delete the last recovery partition
   - Open DiskPart:

```
diskpart
list disk
select disk 0
list partition
select partition X    ← (Select the last small recovery partition at the end)
delete partition override
exit
```

3. Clean up and defrag
   - Run Disk Cleanup as Administrator.
     - Check temporary files, Windows Update cleanup, Recycle Bin, etc.
   - Open Defragment and Optimize Drives, and optimize C:.

4. Shrink C: volume
   - Open Disk Management.
   - Right-click C: and choose Shrink Volume.
   - Enter the amount to shrink so that C: volume fits your target size (e.g., 68 GB used).

From the Hyper-V host (Sirius)

5. Shrink the VM disk
    - Shut down the VM completely.
    - In Hyper-V Manager:
        - Right-click VM > Settings > Hard Drive > Edit.
        - Choose Shrink, specify new smaller size (e.g., 75 GB).

6. Compact the VHDX
    - In the same Edit wizard, choose Compact to reduce physical file size.

From within the guest VM (after restart)

7. Re-enable pagefile
    - Go back to Virtual memory settings.
    - Select "System managed size", click Set, then OK.

8. Verify disk layout
    - Open Disk Management, confirm:
        - Disk 0 size matches (e.g., ~75 GB).
        - No unallocated space or leftover partitions at the end.

Additional Notes

- You do not need to re-enable hibernation or System Restore (unless you have a special requirement).
- The host VHDX file will stay small and not exceed the new disk size, as long as the guest OS disk usage remains below the maximum virtual disk size.

9.8 Using Windows Backup and Restore

An experiment was conducted to use Windows Backup and Restore Windows 7 in Control Panel to move a VM with an unnecessarily large virtual disk to a new VM, in hopes of reducing disk size. While this method did not allow us to shrink the virtual disk (because the new VM's disk had to be at least as large as the original), it proved useful as a way to create a fully functional duplicate VM.

After several failed attempts caused by issues such as mismatched Hyper-V configuration versions (e.g., version 8.0 vs. 9.0), differences in generation (Gen 1 vs. Gen 2), and authentication problems related to domain membership, a working procedure was established.

First, the original VM was joined to the corporate Active Directory domain to resolve network authentication problems when accessing the file share. Then, using Control Panel's Backup and Restore tool, a full system image was created and saved to a network share. It was crucial that the share allowed access by domain credentials (in this case, the domain administrator account), and that the VM being backed up had the Windows 10 installation ISO attached as its virtual DVD drive. This ISO would later be required to boot the new VM into recovery mode.

Next, a new Hyper-V VM was created, carefully matching the original VM's generation, configuration version, and hardware settings as closely as possible. In particular, the new virtual hard disk needed to be at least as large as the original VM's disk as shown in Disk Management — otherwise, the restore would fail. The same Windows 10 installation ISO was also attached to the new VM's virtual DVD drive.

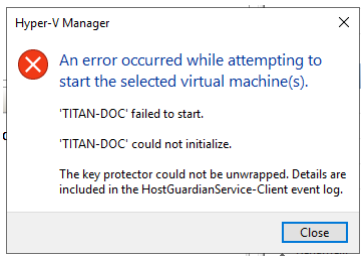
Finally, the new VM was started and booted from the ISO by pressing any key when prompted. Instead of performing a fresh installation, the recovery option was chosen: Repair your computer > Troubleshoot > System Image Recovery. The wizard was then guided to the system image stored on the network share, using the same domain administrator credentials when prompted. All default options in the wizard were accepted, and the restore completed successfully.

While this process does not address disk size reduction, it provides a reliable approach to migrating or duplicating VMs, especially when hardware or host changes are needed.

9.9 Troubleshooting Common Errors

9.9.1 Resolving a Corrupted vTPM That Blocks VM Startup

Error: "The key protector could not be unwrapped."



Problem: When enabling Secure Boot and virtual TPM on the TITAN-DOC VM, it refuses to start with the error “The key protector could not be unwrapped.”

At the same time, Windows 11 Setup will not proceed without TPM 2.0 enabled—creating a catch-22 that blocks both normal VM startup and in-place upgrade.

Root Cause: The VM's virtual TPM (vTPM) metadata or associated key protector had become corrupted. This prevented Hyper-V from initializing the vTPM and caused the VM to crash when attempting to boot with TPM enabled.

Solution Steps:

1. Preserve the original virtual disk: Shutdown and rename the broken VM (e.g. TITAN-DOC → TITAN-DOC-OLD).
2. Make a copy of its virtual hard drive (e.g. copy C:\Imports\TITAN-DOC\Virtual Hard Disks\MERCURY-DOC.vhdx to a temp location such as the desktop)
3. Create a clean clone VM: in Hyper-V Manager, create a new Generation 2 VM (e.g. TITAN-DOC-CLONE) matching CPU, memory, Secure Boot, and firmware settings, but do not attach the OS disk yet.
   Leave TPM disabled during creation.
4. Move the copy of the original virtual hard drive file from the temp location into the new VM’s Virtual Hard Disk folder.
5. Attach the copy of the original disk: under SCSI Controller > Hard Drive, browse to and select the copy of the original virtual hard drive file from Virtual Hard Disk folder.
6. Reset the vTPM protector on the Hyper-V host using PowerShell (run as Administrator):

```
Set-VMKeyProtector -VMName "TITAN-DOC-CLONE" -NewLocalKeyProtector
Enable-VMTPM -VMName "TITAN-DOC-CLONE"
Start-VM -Name "TITAN-DOC-CLONE"
```

7. The cloned VM started successfully with TPM enabled.
8. Log in to confirm the OS, domain membership, and applications are intact.

This solution avoids any changes to Hyper-V host security or guarded fabric features, and sidesteps vTPM corruption entirely.

9.9.2 Microsoft Authentication MFA Infinite Loop

Error: An issue was encountered when logging in as local administrator and attempting to open Visual Studio 2017 Community Edition (which is the version used for supporting the legacy application). The Microsoft credentials were stale and could not be renewed. The multifactor authentication requiring entering a verification code sent to my email address, went into an

infinite loop: the email address was confirmed, a code was sent, I retrieved the code from the email and typed it into the authentication dialog in VS correctly, and the process simply repeated (another code was emailed and entered, repeatedly). However when logging into DEVBOX-1 as domain administrator, I was able to use VS 2017 without any problems, my credentials were not challenged at all, they were somehow already in VS correctly (probably some kind of caching mechanism was used by VS). Why would it work fine when logged in as domain administrator, but go into an infinite loop when logged in as local administrator? Could Active Directory be somehow interfering with the authentication to my online Microsoft account? The work around is for me to always log in to DEVBOX-1 as domain administrator when I need to work with the code. However it would be helpful to understand the root cause.

What's happening?

When you run VS under your Domain Admin session, it picks up your existing AD-issued OAuth token (via Integrated Windows Authentication) to silently acquire "Microsoft Account" tokens. Under a Local Administrator session, VS attempts a full modern auth flow against your Microsoft account but Windows itself isn't AAD-joined, so the authentication handshake never completes and you get caught in that verification loop.

What's happening under the hood

Visual Studio 2017 uses the Web Account Manager (WAM) and the Active Directory Authentication Library (ADAL) to sign you into your work ("Microsoft") account. When you're signed in with a domain account, Windows can leverage Integrated Windows Authentication (IWA) or cached tokens behind the scenes, so you never see the MFA prompt.

When you switch to a local administrator profile, however, none of those domain-joined SSO hooks exist. VS falls back to an embedded login flow, repeatedly brokers a challenge to your tenant's AD FS/Entra endpoint, then loops when it can't fully complete IWA or cache the token properly. Domain-joined logons get Kerberos tickets and silent token refresh; local accounts do not.

Why AD "seems" to interfere

It isn't Active Directory deliberately blocking Microsoft account logins—it's the absence of AD-joined SSO for your local admin user. Your tenant's conditional-access policy or AD FS may require a specific flow (IWA, OAuth via the system browser, etc.). Without a valid SPN/Kerberos ticket or a properly registered WAM broker, the local profile gets stuck in a challenge/response loop

Workarounds / Recommendations

- Stick to your domain account for dev work in VS. That path "just works" because your d1.internal domain is trusted by Azure AD.

- If you ever must run as a local user, you can add your Microsoft identity explicitly under Settings → Accounts → Access work or school → + Connect, and then sign in there. That registers the device for modern auth flows.
- Alternatively, create a Personal Access Token in your Azure DevOps/TFS profile and use that instead of interactive sign-in.

Unless you have a hard requirement to use the local account, the simplest fix is “always open VS as your domain user.”

How to fix it

1. Switch VS to use your system browser In Visual Studio: Tools → Options → Environment → Accounts → “Add and reauthenticate accounts using:” → select System web browser, then sign out/sign back in. This forces the MFA flow through Edge/Chrome, which handles modern OAuth better than the embedded control.
2. Clear stale cached credentials
    a. Open Credential Manager (Control Panel → Credential Manager → Windows Credentials).
    b. Remove entries related to Visual Studio or Azure AD (e.g. “VS.Licenses” or “Generic Token Cache”).
3. Ensure the WAM broker is installed. In an elevated PowerShell prompt inside the VM, run:

```
# For Entra (work) accounts
if (-not (Get-AppxPackage Microsoft.AAD.BrokerPlugin)) {
  Add-AppxPackage -Register `
    "$env:windir\SystemApps\Microsoft.AAD.BrokerPlugin_cw5n1h2txyewy\Appxmanifest.xml" `
    -DisableDevelopmentMode
}
```

4. Then reboot and retry the VS sign-in.

After these steps, your local admin profile will handle the MFA handshake properly—just as smoothly as your domain admin account does.

9.10 Blocker on the Windows 11 upgrade for the DEVBOX-1 Gen 1 VM

We upgraded DEVBOX-1 from Gen 1 to Gen 2 to prepare it for the Windows 11 upgrade. We did this by creating a clone DEVBOX-1-CLONE as a brand-new Gen 2 VM and mapping it's C: drive to a modified version of DEVBOX-1's virtual harddisk .VHDX file. We managed to get the Gen 2 clone running but somehow broke the Windows Recovery (WinRE) tools. An extreme amount of time was spend trying many different approaches but we kept running into errors. We decided to stop the work on trying to fix WinRE because the rest of the Windows 10 functionality seemed to be working. We switched our attention from repairing WinRE, to the real goal of upgrading to Windows 11. In order to resume the fix for WinRE at a later time, the following AI prompt was created.

Can you skip the WinRE repair for now?

We can safely move on to the Windows 11 upgrade without a working WinRE. Skipping the DISM/extraction steps today:

- Won't block the in-place upgrade to Windows 11
- Won't prevent normal day-to-day use of the VM
- Leaves us without "Reset this PC" or Startup Repair—but we've got an exported VM backup, so we're covered if something goes wrong

We can circle back later to restore WinRE once we're past the critical path of getting Windows 11 live.

A future prompt to finish WinRE restoration

Once we do the in-place upgrade to Windows 11, a fresh WinRE image should get laid down and registered as part of setup—so the Recovery Environment will very likely be fixed automatically. We can verify this by issuing the following command in an elevated cmd prompt.

```
reagentc /info
```

If the response is something like "Windows RE status: Enabled", we can consider the WinRE restoration a non-issue going forward. Otherwise we will have to pick this up again.

When we're ready to pick this up again, paste the following into ChatGPT (or here), and we'll get a fresh step-by-step:

```
**"I have a Hyper-V Generation 2 VM clone running Windows 11 on a GPT disk. The Recovery
partition exists but WinRE isn't registered. I've mounted my Windows 11 ISO as D: and I have
a Recovery volume on partition 4. Please provide detailed instructions to:
    1.  Assign a temporary drive letter to the Recovery partition.
    2.  Extract Winre.wim from D:\sources\install.esd (or install.wim) into that partition.
    3.  Register and enable WinRE with reagentc.
    4.  Remove the temporary drive letter.
```

```
Include both DISM (with scratch dir) and 7-Zip options in case of space or permissions
issues."**
```

Alternatively, Just Leave WinRE Disabled

Given the tremendous amount of time and effort spent trying unsuccessfully to enable WinRE on the VM's disk (the VM's internal recovery environment), we could just leave WinRE in the disabled state permanently. We have made a backup of the VM using Hyper-V export. If it's ever needed, we can boot WinRE from the Windows installer ISO. When you need Safe Mode, Command Prompt, or an image restore, just boot the VM from Windows 11 installer ISO and choose Repair your computer → Troubleshoot → Advanced options.

9.11 Catastrophic Failure of Windows 11 and The Backup Export File is Corrupted

Error: catastrophic failure of the VM and corrupt Hyper-V export

In the event the Windows 11 VM fails or is lost, and there is a problem recovering it from the last backup, you can still create a new Windows 11 VM using the Windows 10 retail license key and the Windows installation disks, and then rebuild the computer from scratch. You can either

1. install Windows 10 and activate it with the license key, and then do a free in-place upgrade to Windows 11
2. or you can install Windows 11 directly

Note: If the key isn't accepted at install time, fall back to path #1 (install Win 10 then upgrade).

The retail Windows 10 Pro key remains a valid fallback. Microsoft keeps its activation servers alive long after an OS reaches end of support. Even if Microsoft stops issuing security updates for Windows 10, that doesn't revoke your license or prevent you from reinstalling.

10. Operational Standards

Below are concise operational checklists used during migration and any future setup:

10.1 Standards Followed

Where practical, we adhere to the following industry and internal standards when managing systems (however we favor a pragmatic rather than a dogmatic approach)

- Microsoft Best Practices for Active Directory, DNS, and file sharing security.
- Principle of Least Privilege for all file share and administrative permissions.
- Hyper-V Management Guidelines to ensure safe VM exports and imports.
- Repeatable Setup using internal checklists tailored to our domain structure (d1.internal) and decentralized Hyper-V hosts.
- Workgroup Isolation Policy: All Hyper-V hosts remain in WORKGROUP mode for administrative simplicity and reduce domain dependency at the virtualization layer.

10.2 Repeatable Setup Checklists

File Share Permissions

- Use "Everyone: Full Control" at the share level (if not sensitive).
- Use NTFS-level permissions for actual access control. (Specific permissions are determined by operational needs.)

AD-Integrated DNS Setup

- Enable dynamic updates (secure only).
- Define forwarders to public resolvers (e.g., 8.8.8.8, 1.1.1.1).
- Replicate DNS zones to:
    - dc-01
    - dc-02
- To verify reverse lookup zones exist:

```
Get-DnsServerZone -ComputerName dc-01 | Where-Object { $_.IsReverseLookupZone }
```

Domain Join and Workstation Profile Transition

- Backup local user data (Desktop/Documents/Downloads).
- Join workstation to d1.internal domain.
- Log in once with domain user to create profile.
- Reconfigure network drives and printers manually.

Scheduled Task Migration

Each task scheduled on lunar-batch was updated with the appropriate domain account username and password.

Hyper-V Export and Disaster Recovery Preparation

Exporting both domain controllers together (one-time disaster recovery snapshot):

1. Shut down both DCs via Hyper-V Manager.
2. On each hypervisor host:
   - Right-click the VM > Export.
   - Target: External volume (e.g., E:\VM-Exports\YYYYMMDD\).
3. Confirm export folder includes the full VM config, VHDs, and snapshots (if any).
4. Label folders with computer name and export date.

For all other hyper-v vm exports, follow company policy.

11. Testing and Validation

Always test everything! – well known old saying

11.1 Domain Resilience Tests

- Each domain controller was shut down individually to verify that authentication, DNS resolution, and Group Policy processing continued without interruption.
- Verified login success from domain-joined workstations during each simulated DC outage.

11.2 Post-Migration Validation

- Confirmed domain join success and user profile creation on migrated workstations.
- Verified DNS resolution using nslookup for local and external domains.
- Tested file share access and permissions using domain credentials.
- Confirmed TFS, SQL Server, and IIS services were reachable and functional.

11.3 Layered Testing

Layered testing refers to progressively testing systems and configurations at increasing levels of complexity and scale to reduce risk and catch issues early. Each layer reduced uncertainty before moving on to the next, minimizing the blast radius of potential errors. It's a phased approach to validation.

Various different variations of layered testing were used in this project. Examples of layered testing that were used: WSUS proof-of-concept, domain migration.

11.3.1 WSUS Proof-of-Concept – Layered Testing

1. Initial Setup Validation:
   - Confirmed WSUS2 could be installed and configured using WID and IIS.
   - Verified default update classifications and product categories.
2. Domain Join and Communication Check:
   - Joined a single test system (win2016-5) to the d1.internal domain.
   - Applied local group policy and confirmed registration with WSUS2.
3. Approval and Delivery Simulation:
   - Approved a Windows Update and ensured no direct downloads from Microsoft.
4. Update Visibility and Targeting:
   - Tested client group membership targeting on selected updates.
5. Installation and Reporting:
   - Installed update via GUI (no forced CLI tools) and confirmed status report.

11.3.2 Domain Migration – Layered Testing

- Layered environment testing occurred by building and validating the d1.internal domain in isolation before migrating any production systems.
- Hyper-V rebalancing of virtual machines between hypervisors tested on cloned virtual machine exports.
- Staged migration of machines allowed validation of DNS, AD authentication, and file share access step-by-step.

12. Project Outcomes

12.1 IT Functionality Improved or Modernized

The domain migration project delivered multiple infrastructure and operational upgrades that increased manageability, security, and resilience:

- **Active Directory consolidation**: Legacy domain dtek.internal was fully retired, and all eligible systems now operate under the unified d1.internal domain with standardized Group Policy and account controls.
- **DNS structure and replication improved**: DNS zones were cleaned up, reverse lookup zones validated, and replication between domain controllers confirmed, improving name resolution reliability and administrative clarity.
- **Workstation standardization**: All domain-joined workstations now follow a consistent profile setup and permissions model. Manual transitions were verified and documented to simplify future onboarding.
- **Legacy systems preserved and stabilized**: A fragile, multi-tier application stack (involving Visual Studio, IIS, SQL Server, and TFS) was successfully migrated with no loss of functionality. Key dependencies were carefully re-established under the new domain.
- **Disaster recovery preparation introduced**: Clean Hyper-V exports of both domain controllers were created and archived, establishing a solid baseline for future DR planning.
- **Documentation culture reinforced**: Migration processes, DNS configuration, file share policies, and scheduled task management were all documented to a level suitable for junior administrators—building internal knowledge continuity.
- **Windows updates centralized and automated**: Monthly patches are now downloaded once to the WSUS server and distributed internally, reducing Internet bandwidth use and eliminating manual update checks on individual systems.

12.2 Lessons Learned and Recommendations

This project revealed several insights and improvement opportunities for future infrastructure changes:

- **System interdependencies must be mapped early**. The tight coupling of legacy systems required significant effort to trace and preserve during the transition. Documenting app-to-service dependencies should become a standard pre-migration step.

- **Reverse lookup zones are often overlooked**. Their absence can lead to subtle network issues. We recommend including zone verification in the checklist for any future DC deployment.
- **Isolated Hyper-V hosts simplify DR but require intentional exports**. Keeping Hyper-V in WORKGROUP mode worked well operationally, but it places responsibility for exports squarely on local admins. Regular DR export schedules should be institutionalized.
- **Incremental wins matter**. While this project involved legacy systems, it provided opportunities to clean up stale DNS entries, update file permissions, and align naming conventions. Even legacy migrations can result in a more modern and maintainable environment when handled deliberately.
- **Keep junior administrators in mind**. Many of the operational checklists developed during this project can now serve as reusable training material—reducing onboarding friction and institutionalizing best practices.
- **Roll out WSUS in stages.** Starting with a single test system allowed safe evaluation before affecting production servers—minimizing risk while confirming real benefits.
- **System image restore is not a disk shrinking tool.** Using Windows Backup and Restore (Windows 7) proved valuable for creating fully functional duplicate VMs but cannot be relied on to reduce disk size.

13. Outstanding Tasks and Future Plans

The following items were intentionally deferred or identified during the project as opportunities for future improvement, testing, or documentation.

13.1 Disaster Recovery Simulation

- A full domain recovery test has not yet been performed.
- Simulating loss of both domain controllers and testing full restore from Hyper-V exports is a critical task.
- This will help validate current backup strategy and improve confidence in disaster readiness.

13.2 File Share Permissions Audit

Although the share-level and NTFS permissions have been standardized, a complete audit of all sensitive shares should be performed:

- Check for over-permissive access
- Reconfirm Principle of Least Privilege adherence

13.3 TFS Migration to Git repo in Azure DevOps

To modernize version control and simplify access, all Team Foundation Server (TFS) projects on lunar-tfs will be migrated to Git repositories hosted in Azure DevOps. This will improve developer experience, support remote collaboration, and align with industry trends.

13.4 Final Cleanup of File Share Permissions on lunar-file

Goal: Ensure all users—regardless of domain or WORKGROUP membership—have full access to shared folders hosted on lunar-file, using the principle:

- Everyone: Full Control (Share-level)
- NTFS-level: Everyone: Full Control

This ensures simplicity and maximum accessibility for non-sensitive data.

Step 1: Confirm and Take Ownership (Including All Child Items)

1. Log in to lunar-file as a local administrator.
2. Navigate to the root of each shared folder (e.g., D:\Shared).
3. Right-click the folder → Properties → Security tab → Advanced.
4. At the top, click Change next to the Owner field.
5. In the "Select User or Group" dialog:
    - click "Locations..." and choose lunar-file to scope to the local computer.

    - Type: Administrator → click Check Names → verify it resolves to lunar-file\Administrator → click OK.
6. Back in the Advanced Security Settings window: check the box labeled "Replace owner on subcontainers and objects".
7. Click Apply. This will recursively take ownership of all child files and folders.
8. Click OK to close all dialogs.

Why this matters: If this step is skipped, subsequent NTFS permission changes may silently fail on files/folders you don't yet own—leading to inconsistent access or even total lockout when old entries are removed.

Step 2: Remove Unnecessary NTFS Permissions

1. Go back to Properties → Security tab → Advanced.
2. Click Disable inheritance.
3. When prompted, choose Convert inherited permissions into explicit permissions.
4. Carefully remove all entries except:
    - Administrators: Full Control
    - SYSTEM: Full Control
5. Now add: Everyone: Full Control

Be very cautious not to remove your current user's access before applying the new entries.

You may wish to apply these changes one folder at a time to avoid getting locked out.

Step 3: Reset Share Permissions

1. Go to Computer Management → Shared Folders → Shares.
2. Right-click the share → Properties → Share Permissions tab.
3. Remove all entries.
4. Add: Everyone: Full Control

Step 4: Repeat for All Shared Folders

Repeat the NTFS and Share steps above for each folder shared on lunar-file.

Step 5: Confirm Effective Access

- On a separate domain-joined machine (d1.internal) and a WORKGROUP machine, attempt to access \\lunar-file\sharename.
- Try file creation, editing, and deletion to confirm full access.
- If issues occur, double-check both NTFS and share-level permissions.

13.5 Verification and Testing of webserv-test under d1.internal

Conduct end-to-end validation of webserv-test to confirm post-migration functionality:

1. Ensure the server is joined to d1.internal and rebooted.
2. Confirm SQL connectivity using Invoke-Sqlcmd or equivalent testing via SSMS.
3. Validate IIS application pools are using domain service accounts or ApplicationPoolIdentity, and that identity permissions are still valid.
4. Access hosted websites and confirm correct functionality.
5. Check Windows Firewall rules via GUI (Network and Sharing Center → Windows Defender Firewall → Advanced Settings).

13.6 Review and Removal of Legacy SQL Server Logins from dtek.internal Domain

Remove obsolete dtek.internal logins from SQL Server.

On lunar-sql, delete the following under Security > Logins:

- DTEK\Administrator
- DTEK\DEVBOX-1$
- DTEK\EU-WEB$
- DTEK\WEBSERV-TEST$

On lunar-tfs, delete DTEK\Administrator

Caution: Do not delete any login without first verifying the system's successful operation post-migration.

13.7 Pilot Upgrade of Windows 10 Virtual Machine to Windows 11

A pilot test will be performed to determine if domain-joined Windows 10 virtual machines hosted on Hyper-V can be upgraded to Windows 11.

- VM selected for pilot: w10p-3 (hosted on sirius).
- w10p-3 is a minimally configured Windows 10 workstation, joined to d1.internal.
- Objectives: Confirm feasibility, identify required configuration changes, evaluate process and stability.

Planned steps:

1. Checkpoint w10p-3 before upgrade.
2. Ensure VM uses Generation 2 with Secure Boot and TPM enabled.

3. Verify virtual hardware meets Windows 11 requirements (e.g. CPU, RAM, disk).
4. Upgrade the VM configuration version if needed.
5. Mount the Windows 11 ISO and run in-place upgrade.
6. Post-upgrade, confirm domain membership, GPO application, and system functionality.

Success criteria:

- VM remains domain-joined post-upgrade.
- No device or driver errors.
- System performs reliably with no loss of function.

This pilot will help guide whether future upgrades of additional VMs to Windows 11 are technically viable and worth pursuing.

13.8 WSUS Adoption and Future Expansion

With the WSUS proof-of-concept now validated, the following next steps will help transition the environment toward broader adoption and stable, long-term operations.

Windows 10 Pilot

- Create a dedicated WSUS computer group for Windows 10 devices.
- Select a Windows 10 test client that has not yet received the June 2025 updates.
- Apply the same local WSUS configuration as used in the proof-of-concept.
- Monitor update detection, installation, and reporting via the WSUS console.

Gradual Onboarding of Production Systems

- Begin onboarding additional Windows Server 2016 production systems using the same local policy and group assignment process.
- Monitor update behavior closely, particularly on critical infrastructure roles.
- As confidence grows, extend WSUS coverage to include all supported OS versions in the domain.

Update Organization and Staging

- Define staging computer groups for each supported OS version (e.g., Win10-Test, Win10-Prod, Win2016-Prod, etc.).
- Use these groups to stage updates before full deployment.
- Consider establishing a convention for update approvals (e.g., weekly review, monthly rollout).

Ongoing WSUS Maintenance

- Configure automatic synchronization in the WSUS console to ensure updates are regularly pulled from Microsoft Update.
- Periodically review and adjust the list of enabled product categories and classifications as organizational needs evolve.
- Re-evaluate IIS performance tuning and storage usage if the WSUS workload increases over time.

Operational Monitoring

- Confirm that newly added clients are automatically polling the WSUS server and checking in as expected.
- Use the WSUS console's reports and computer group views to track compliance and detect any client registration issues.
- Document any recurring manual steps or interventions that could be scripted or automated in future phases.

14. References

- **Active Directory Domain Services Overview – Introduction to AD DS**
  https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/get-started/virtual-dc/active-directory-domain-services-overview
- **Active Directory Users and Computers – Managing user accounts**
  https://learn.microsoft.com/en-us/windows-server/identity/ad-ds/manage-user-accounts-in-windows-server
- **Azure Cosmos DB Emulator – Command-line and PowerShell reference**
  https://learn.microsoft.com/en-us/azure/cosmos-db/emulator-windows-arguments
- **Azure Cosmos DB Emulator – Development and testing with the emulator**
  https://learn.microsoft.com/en-us/azure/cosmos-db/how-to-develop-emulator
- **Azure Cosmos DB Emulator – Overview and usage**
  https://learn.microsoft.com/en-us/azure/cosmos-db/emulator
- **Azure Cosmos DB Emulator – Release notes**
  https://learn.microsoft.com/en-us/azure/cosmos-db/emulator-release-notes
- **Azure Cosmos DB Emulator – Troubleshooting guide**
  https://learn.microsoft.com/en-us/troubleshoot/azure/cosmos-db/tools-connectors/emulator
- **Configures a key protector for a virtual machine**
  https://learn.microsoft.com/en-us/powershell/module/hyper-v/set-vmkeyprotector?view=windowsserver2025-ps
- **DNS Client Configuration – Best practices for DNS client settings on domain controllers**
  https://learn.microsoft.com/en-us/troubleshoot/windows-server/networking/best-practices-for-dns-client-settings
- **DNS Manager – Managing DNS resource records**
  https://learn.microsoft.com/en-us/windows-server/networking/dns/manage-resource-records
- **Dynamic DNS Updates – How DNS dynamic updates work**
  https://learn.microsoft.com/en-us/windows-server/networking/dns/dynamic-update
- **Enables TPM functionality on a virtual machine**
  https://learn.microsoft.com/en-us/powershell/module/hyper-v/enable-vmtpm?view=windowsserver2025-ps

- **Generation 2 virtual machine security settings for Hyper-V**
  https://learn.microsoft.com/en-us/windows-server/virtualization/hyper-v/learn-more/generation-2-virtual-machine-security-settings-for-hyper-v
- **Hyper-V TPM config**
  https://techcommunity.microsoft.com/blog/itopstalkblog/how-to-run-a-windows-11-vm-on-hyper-v/3713948
- **IIS Configuration – Configure application pool identity**
  https://learn.microsoft.com/en-us/iis/manage/configuring-security/application-pool-identities
- **"key protector could not be unwrapped" Error**
  https://learn.microsoft.com/en-us/answers/questions/2287161/server-2022-2025-hyper-v-new-instance-error-key-pr
- **Microsoft TPM/CPU requirements for Windows 11**
  https://learn.microsoft.com/en-us/windows/whats-new/windows-11-requirements
- **PowerShell – DNS Zone Listing – Get-DnsServerZone**
  https://learn.microsoft.com/en-us/powershell/module/dnsserver/get-dnsserverzone
- **PowerShell – DNS Zone Management – Add-DnsServerPrimaryZone**
  https://learn.microsoft.com/en-us/powershell/module/dnsserver/add-dnsserverprimaryzone
- **PowerShell – Invoke-Sqlcmd Cmdlet – Invoke-Sqlcmd**
  https://learn.microsoft.com/en-us/powershell/module/sqlserver/invoke-sqlcmd
- **PowerShell – SQL Server Cmdlets – SQLServer module overview**
  https://learn.microsoft.com/en-us/powershell/module/sqlserver/
- **PowerShell – SQL Server Module Installation – Installing the SqlServer module**
  https://learn.microsoft.com/en-us/sql/powershell/download-sql-server-ps-module
- **Project-Specific Documentation – README for CosmosDB Emulator Deployment**
  https://rob-das-win.azurewebsites.net/html/README.html#development-database-azure-cosmos-db-emulator
- **System Properties (sysdm.cpl) – Accessing system settings**
  https://learn.microsoft.com/en-us/windows/win32/shell/executing-control-panel-items
- **TFS Migration – Migrating from TFS to Azure DevOps Services**
  https://learn.microsoft.com/en-us/azure/devops/migrate/migration-overview

- **Task Scheduler – Overview and usage** https://learn.microsoft.com/en-us/windows/win32/taskschd/task-scheduler-start-page
- **Windows 10 to 11 upgrade** https://www.microsoft.com/en-ca/software-download/windows11
- **Windows 11 system requirements** https://www.microsoft.com/en-ca/windows/windows-11-specifications
- **Windows Server Manager – Overview and usage** https://learn.microsoft.com/en-us/windows-server/administration/server-manager/server-manager
- **Windows Server Update Services (WSUS)** https://learn.microsoft.com/en-us/windows/deployment/update/waas-manage-updates-wsus
- **WSUS and Configuration Manager** https://learn.microsoft.com/en-us/troubleshoot/mem/configmgr/update-management/wsus-maintenance-guide