AD Domain Migration and Infrastructure Realignment

Project Report

by Rob Das

DTEK Consulting Services Ltd.
Alberta, Canada

January 2025 – November 2025

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

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, upgrading from Windows 10 to 11, and establishing a repeatable, standards-based configuration for long-term stability. The project followed a low-code approach to simplify long term maintenance and support.

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

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 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:

- Replace multi-purpose physical servers with role-dedicated hypervisors.
- Create a new virtualized file server.
- Migrate user profiles and application services.
- Establish operational standards and checklists for long-term maintainability and disaster recovery.
- Introduce centralized Windows Update management to streamline Windows patching and reduce admin costs.
- Upgrade all end-of-life Windows 10 machines with either Windows 11 or Linux Mint.

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.

Note: for security reasons some of the internal computer names have been changed.

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 and infrastructure realignment 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 member*: systems requiring Active Directory services were joined to d1.internal
- *workgroup system*: 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) – to configure, sync, update approval
- Local Group Policy Editor (gpedit.msc) – for configuring WSUS client behavior
- ChatGPT – for architectural planning, PowerShell scripting, troubleshooting
- Linux – Terminal, Network Settings, File Manager (nemo), Backup (timeshift), RDP clients (remmina, xrdp), LibreOffice, Text Editor (xed), Sound/Video Device Manager

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 chapter 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

This section discusses 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: Least-Privilege

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.

5.5 Linux Mint Laptops

All laptops currently running Windows 10 that do not meet the minimum requirements for Windows 11 are being upgraded to Linux Mint. This change is driven primarily by the upcoming end-of-life for Windows 10 in October 2025, ensuring continued secure and supported operation without requiring hardware replacement. The primary users of these systems are developers and systems administrators, who rely on them for remote access to Windows-based infrastructure, file management, office productivity tasks, web browsing, and occasional local audio/video playback.

The migration approach focuses on replicating the essential capabilities previously available under Windows 10, using Linux Mint’s built-in features and user-friendly GUI tools. This includes configuring RDP access to Windows systems, enabling full multimedia support, setting up office document editing, and ensuring seamless integration into existing network and authentication environments. Select Linux tools that are similar to their Windows equivalents are installed and enabled to maintain operational continuity while minimizing retraining needs. The migration process ensures that users retain full access to internal systems, development environments, and administrative utilities, with minimal disruption to workflows.

5.6 Printing to TITAN\Canon

The Canon TS5300 printer is currently hosted on TITAN, a Windows Server 2019 machine operating in a standalone WORKGROUP configuration. TITAN is presently serving as the

company’s network printer server, providing access to the Canon printer for both Windows 11 domain-joined workstations and Linux Mint laptops. This section outlines the configuration required to enable printing from both environments.

For Windows 11 machines joined to the D1 domain, the printer is accessed via the shared path \\TITAN\Canon. Initial setup must be performed under a local administrator account, such as GALAXY\Administrator, to install the Canon TS5300 driver and register the shared printer. Once installed, domain users can add the printer through standard Windows printer discovery or by manually specifying the network path. No additional driver installation is required for domain users after the initial setup.

Linux Mint laptops connect to the printer using the CUPS printing system with Samba support. Users should open the Printers utility, select “Windows Printer via Samba,” and enter the path smb://TITAN/Canon. If prompted, credentials for TITAN must be supplied. The Canon TS5300 driver should be selected if available; otherwise, a generic Canon Inkjet driver may be used. A successful test page confirms connectivity and driver compatibility.

This configuration ensures that both Windows and Linux clients can print reliably to the shared Canon printer hosted on TITAN.

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
WORKGROUP	Database Server (Cosmos DB)	1

Domain / Workgroup	Role	Number of Computers
	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 Computers	3

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

One of the most intricate and time-consuming parts of the domain migration and infrastructure alignment 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 Patch Management

Windows Server Update Services (WSUS), a software update management system, allows systems administrators 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 Current WSUS Automation Schedule

In order to deploy updates in an automated fashion, we have adopted the following schedule.

Event	**Time (MST)**	**Purpose**
Microsoft Patch Tuesday	Second Tuesday each month at 11:00 AM	Monthly release of updates by Microsoft
WSUS Daily Sync	Daily at 9:00 AM	Pulls updates from Microsoft
Admin Approval Window	2:00-4:00 PM on the day after Patch Tuesday	Manual review and approval of updates
Daily Client Scan, triggered via PowerShell script *S:\Code\forcewuscan.ps1*	Daily at 6:00 PM Exception: WSUS server daily at 5:00 PM	Initiates update detection on clients
Update Installation, triggered via Policy	Daily at 1:00 AM Exception: WSUS server daily at 9:00 PM	Applies approved updates
Daily Client Restart via PowerShell script *S:\Code\forcewurestart.ps1*	Daily at 6:00 AM Exception: WSUS server daily at 10:00 PM	Forces restart to complete update installation

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

Daily items are independent of each other. If ran in the wrong order (perhaps due to an outage) they will automatically catch up the following day, once the correct order has been re-established.

However, even with this level of automation, Windows Updates can still be compromised by unexpected shutdown in the middle of an update (possibly due to power outages). To resolve this, see the troubleshooting guide at the end of this chapter.

Since Windows Server 2012 R2 reached end of extended support in 2023 and is no longer eligible for security or quality rollups, these systems are excluded from the WSUS automation framework. However, Microsoft continues to release the monthly Windows Malicious Software Removal Tool (MSRT) for 2012 R2. To ensure these updates are applied with minimal overhead, these servers are configured through the Control Panel → Windows Update settings to “Install updates automatically”.

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.

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.

8.4 Non-Default Configuration

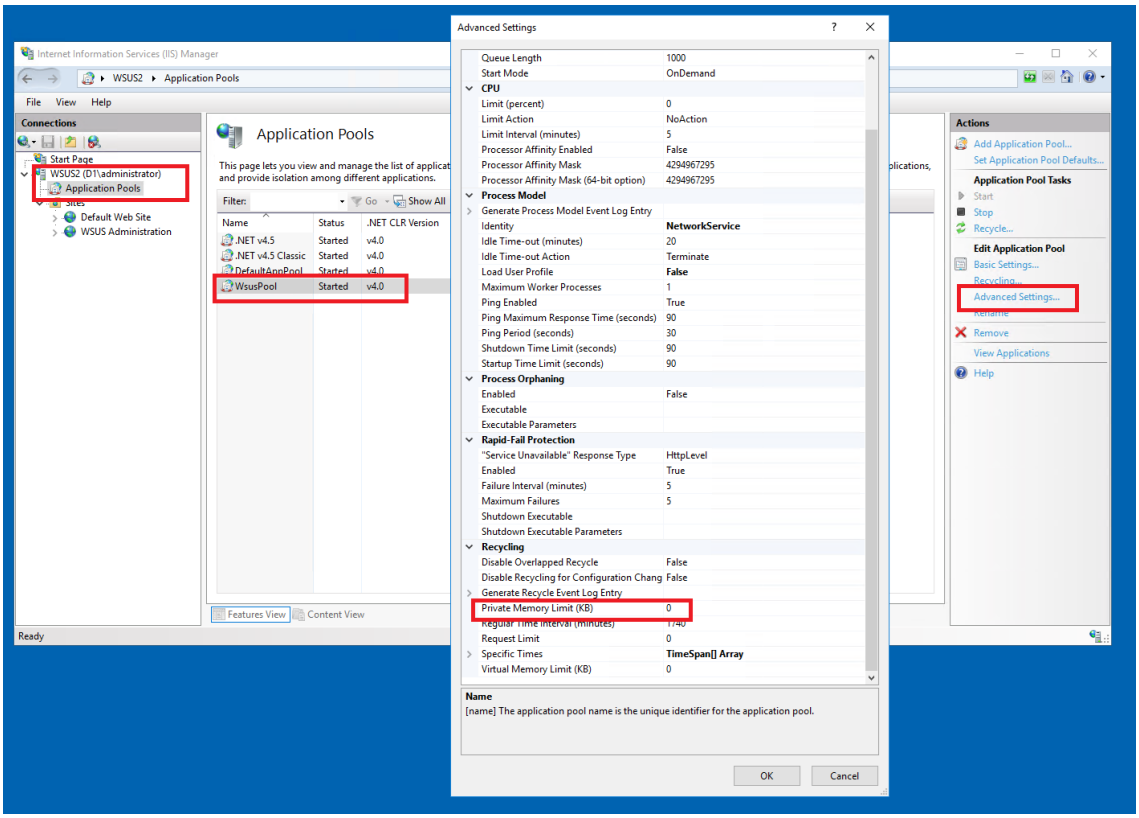
To optimize WSUS for its virtualized environment and ensure scalability, several settings were adjusted from their defaults.

Hyper-V VM Settings:

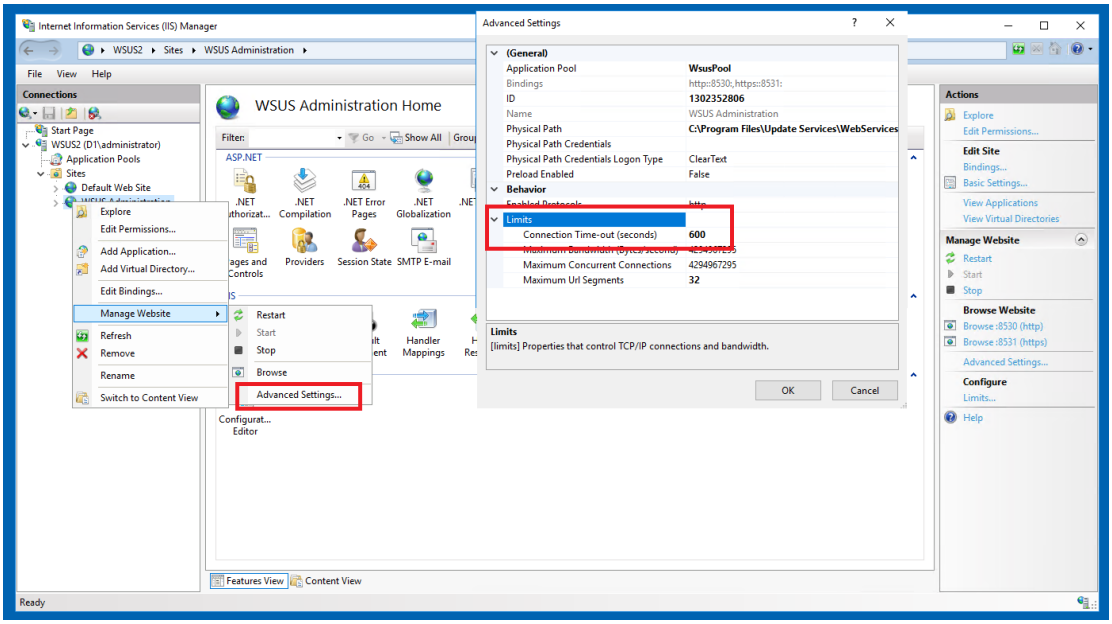
- Dynamic Memory: 2048–8192 MB
- Processor: 1 virtual CPU

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

IIS Configuration:



- WsusPool Private Memory Limit: 0 (unlimited)



- Connection Timeout: 600 seconds

After adjusting, restart the WSUS IIS site: right-click WSUS Administration → Manage Website → Restart, or run iisreset from an elevated command prompt.

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.

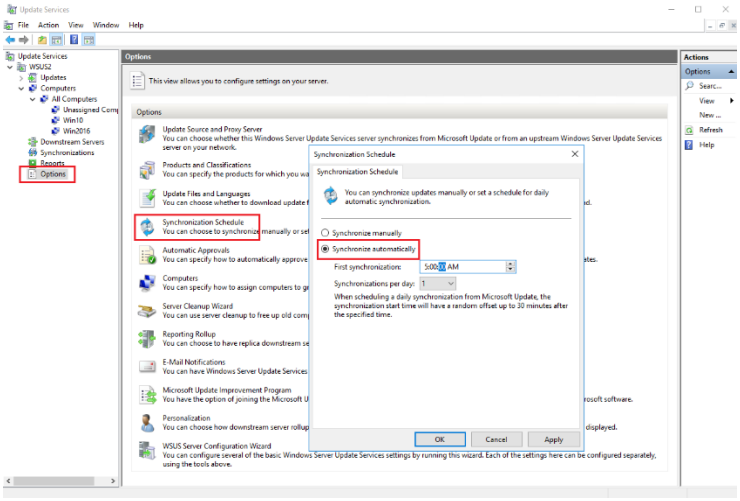
Enabling WSUS Synchronization Schedule

Open

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

Select

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



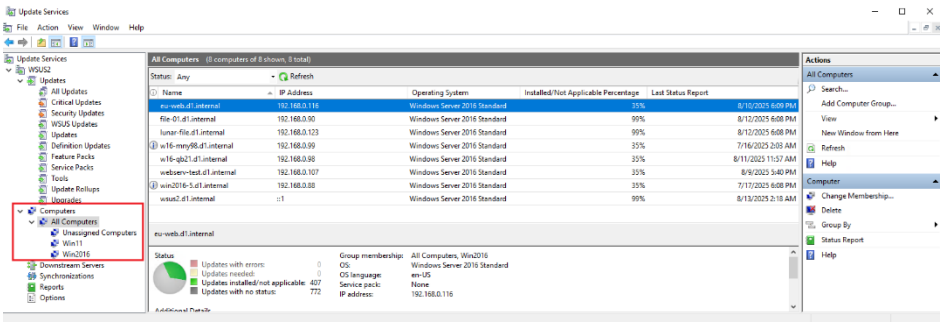
WSUS will automatically fetch new updates from Microsoft every morning.

8.5 Steps for Adding Additional Clients

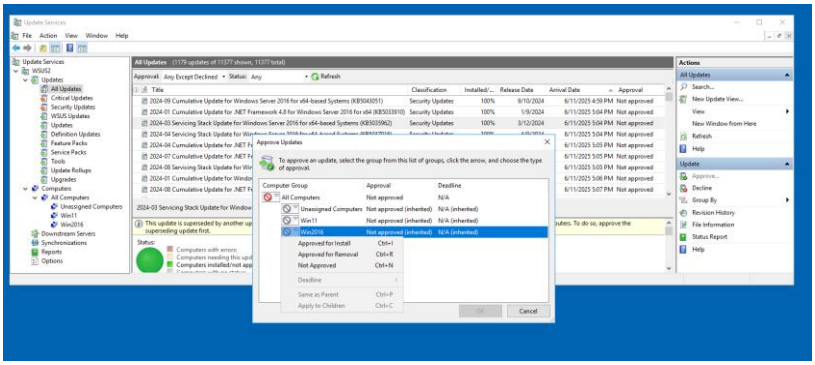
Clients are organized by OS on the WSUS server using WSUS computer groups.

Server Steps

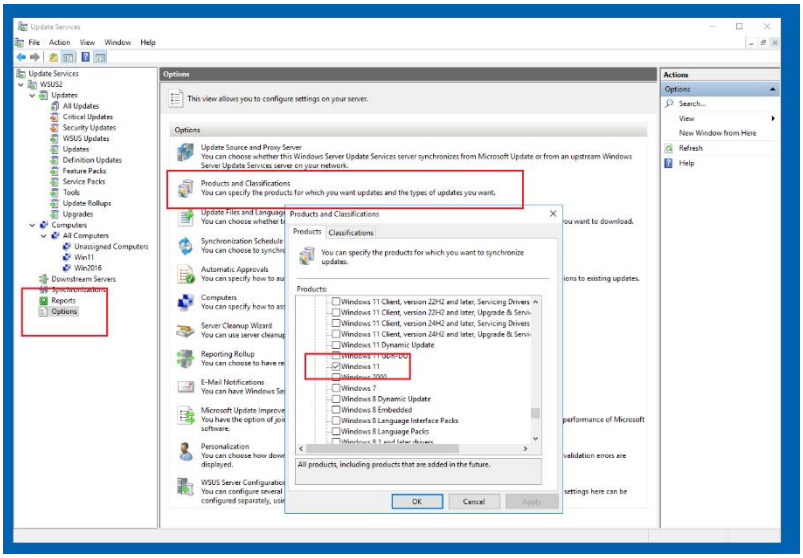
Currently we have separate computer groups, Win11 and Win2016, for Windows 11 updates, and Windows Server 2016 updates.



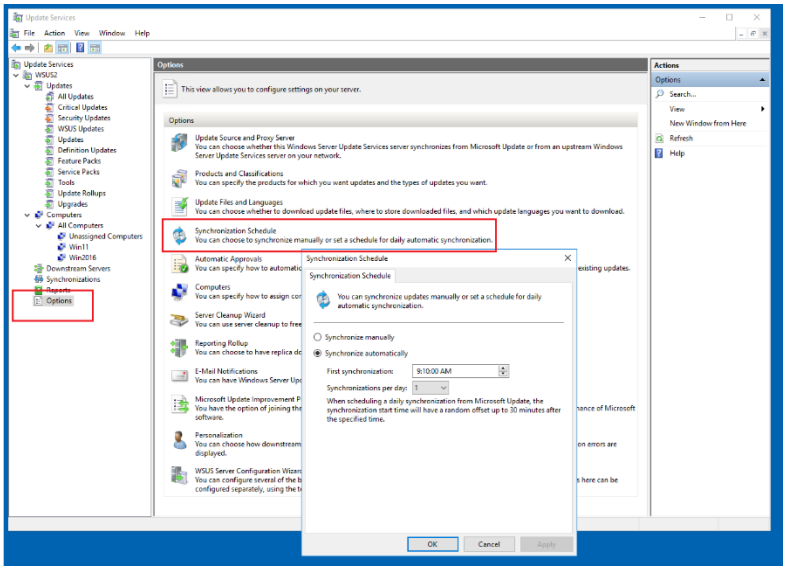
When a client first contacts the server, it is automatically added to the Unassigned Computers group, and must be manually moved into one of the computer groups so that it receives the correct Windows updates. The Windows updates must be approved each month and targeted to the correct computer group. This is done manually by the sys admin the after they are published by Microsoft on “Patch Tuesdays”.



When adding new computer groups such as Win11, you need to select the products that determine the updates for the new computer group. This is done in the WSUS console > Options > Products and Classifications.



This will tell WSUS what to download during synchronization. Synchronization is when the WSUS server reaches out to Microsoft's online update service and fetches the updates corresponding to the products that were selected. Synchronizations are scheduled from the WSUS console > Options > Synchronization Schedule



Client Steps

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 script. “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

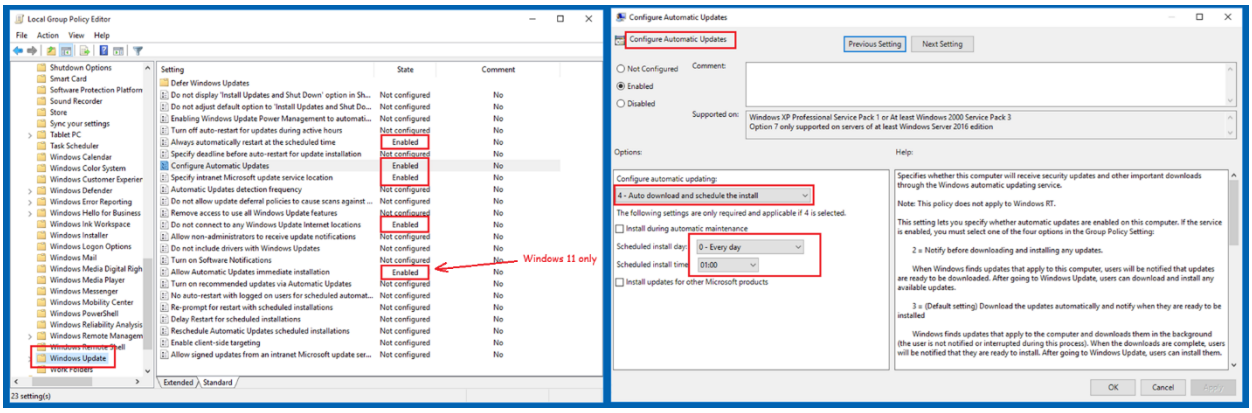
On the client, open the Local Group Policy Editor (gpedit.msc)

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

and set these policies:

1. Enable "Specify intranet Microsoft update service location"
    a. Detection server: http://wsus2.d1.internal:8530
    b. Statistics server: http://wsus2.d1.internal:8530
    c. Leave alternate download server not configured.
2. Enable "Configure Automatic Updates"
    a. "4 - Auto download and schedule the install"
    b. Scheduled install day: "daily"
    c. Scheduled install time: "1:00 AM"
3. Enable "Do not connect to any Windows Update Internet locations"
    this prevents fallback to Microsoft online

4. Enable "Always automatically restart at the scheduled time". This enforces a restart 15 minutes after installation completes, even if users are logged in.
5. ***For Windows 11 systems only:***
   Enable "Allow Automatic Updates Immediate Installation". This is needed to ensure all eligible updates install automatically without requiring user interaction, to align Windows 11 with the automated update behaviour used by Windows Server.



Apply the policies using:

```
gpupdate /force
```

Important: We use server-side targeting; 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.

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

Using Windows Task Scheduler on the client computer, create a scheduled task that runs a designated PowerShell script according to the WSUS automation schedule shown previously in this chapter. The script is on the network at this path:

```
\\lunar-file\share\Code\forcewuscan.ps1
```

Configure the task as follows.

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

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

This initiates the monthly “patch Tuesday” update process, which sometimes gets ignored by the overnight “Configure Automatic Updates” policy.

Alternatively, you can check to see if there is a task scheduler import script named \\lunar-file\share\Code\Force Windows Update Scan.xml

Step 5 - Automate the Restart Every Morning

On some client computers, the “Always automatically restart at the scheduled time” policy does not reliably enforce restarts after update installation. To ensure updates are completed, create a scheduled task in Windows Task Scheduler to force a system restart according to the WSUS automation schedule shown previously in this chapter.

Configure the task as follows.

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

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

This guarantees that computers finish installing updates if a Windows Update reboot is pending, without requiring user interaction.

Alternatively, you can check to see if there is a task scheduler import script named \\lunar-file\share\Code\Force Windows Update Restart.xml

Step 6 – 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 7 – Monitor Update Status

To confirm update detection and installation:

- Check the client’s status in the WSUS console.
- Check the script log file \\lunar-file\share\Code\forcewu-log.txt
- Check Update History on the client 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. We have standardized on the Server-Side Targeting approach, not the Client-Side Targeting approach, for simplicity. And we have adopted conventions based around our Current WSUS Automation Schedule, shown earlier in this chapter.

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 for Windows Server 2016 systems.

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

Update Approval Process

Update approvals are performed using the WSUS console. Updates can be approved for individual computer groups (e.g., Win2016), or all Computers group (for critical or universally applicable updates)

Updates are not auto-approved. Each month's updates (typically released on Patch Tuesday) are reviewed and approved by a sys admin.

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

Participating WSUS systems are configured for automatic daily synchronization.

Client Polling Interval

The WSUS clients automatically poll the server for updates according to the schedule published above in the section entitled Current WSUS Automation Schedule.

Status Update (2025-Aug-23) – WSUS Automation for Windows Server 2019

The automated WSUS update workflow that works for Windows 11 and Server 2016 has been set up on a test Windows Server 2019 system. Full validation could not be performed because all current updates were already installed. The system is configured and ready, and the next update cycle will allow verification of automatic update functionality. If automation fails in the future, manual updates can be applied as a fallback.

Status Update (2025-Oct-17) – WSUS Automation for Windows Server 2019

Windows Server 2019 has now been fully validated.

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.

Note: This fix 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 on the WSUS server and verify WsusPool is running

Issue: Updates Install but Auto-Restart Never Occurs

Observed on:

IIS web servers eu-web and webserv-test

Symptoms:

- Update 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:

Manually restart the WSUS client computer.

Issue: Windows Update Corrupted

Symptoms:

Resolutions:

1. Reboot and try again.
2. Run Windows Update Troubleshooter and try again.
3. Delete the `SoftwareDistribution` folder and try again. This resolution is complicated by the fact that Windows services constantly use this folder, so you need to temporarily stop those services, delete the folder, and then restart the services. The services in question are the Windows Update service (WUAUSERV) and the Background Intelligent Transfer Service (BITS). Here are the steps to delete the folder, after which you can try the windows update again.

```
net stop wuauserv
net stop bits
delete C:\Windows\SoftwareDistribution
net start wuauserv
net start bits
```

9. Windows 10 Replacement

9.1 Overview

This chapter documents the replacement and upgrade of all systems running Windows 10, driven primarily by Microsoft's announced end-of-life for Windows 10 in October 2025 and the organization's broader infrastructure modernization goals. The Windows 10 replacement initiative was incorporated into the overall AD domain migration and infrastructure realignment project as a scope change, recognizing Windows 10 replacement's contribution to long-term infrastructure goals. Microsoft's free upgrade policy for eligible Windows 10 installations was leveraged to reuse existing retail licenses, upgrading virtual machines directly to Windows 11 and enabling physical laptops that did not meet Windows 11 hardware requirements to have their licenses reused on new VMs. Those laptops were repurposed with Linux Mint to continue delivering essential functionality. This approach ensured continuity of service Windows 10 users, maintained compliance with licensing requirements, and supported the modernization of the organization's IT environment.

9.2 System Categorization

During the planning and execution of the Windows 10 to Windows 11 migration, systems were grouped based on their current use, lifecycle stage, and hardware or licensing constraints. This helped determine whether in-place upgrades, clean installations, or system repurposing would be the most appropriate path forward.

9.2.1 Dormant Virtual Machines

This category includes virtual machines that were not actively in use at the time of the migration but are maintained in a ready state for future assignments. These systems serve as a flexible pool of workstations that can be quickly provisioned for new users or temporary roles. When unassigned, they are routinely wiped and reset so that the next user starts with a clean baseline. Since these machines were already in a reset state, the migration approach was straightforward. Each one was deleted and recreated from scratch to ensure it met Windows 11's minimum requirements other than the hypervisor's unsupported CPUs, which was addressed using the Microsoft-documented `AllowUpgradesWithUnsupportedTPMOrCPU` registry override.

9.2.2 In-Use Virtual Machines

These systems were actively used in production or development environments and could not be interrupted or reset without disrupting ongoing work. The upgrade strategy in this case prioritized stability, minimal downtime, and preservation of user environments. Each virtual machine was inspected and adjusted to meet the Windows 11 requirements. Although the underlying hypervisor hardware still lacked a supported CPU, this was resolved using the same

documented registry override. The upgrade was performed in-place using Windows 11 installation media mounted directly within the VM, preserving installed applications, user data, and configuration. Activation was verified post-upgrade to ensure that licensing was retained, and no issues were encountered across any of the upgraded VMs.

9.2.3 Physical Machines

Physical laptops posed a different challenge due to their hardware limitations and licensing restrictions. In these cases, systems were upgraded by replacing the original Windows 10 installation entirely.

The original laptops were wiped and reimaged with Linux Mint to permanently remove Windows and avoid any risk of dual-boot configurations or licensing conflicts.

New Windows 11 installations were then performed on virtual machines, using clean installations and repurposed digital licenses transferred from the original systems. This solution proved efficient, secure, and cleanly separated the Linux and Windows environments.

9.3 Upgrade Readiness and Prerequisite Checks

Before upgrading a Windows 10 computer to Windows 11, a series of compatibility and configuration checks must be performed to ensure the system meets—or can be made to meet—the minimum requirements.

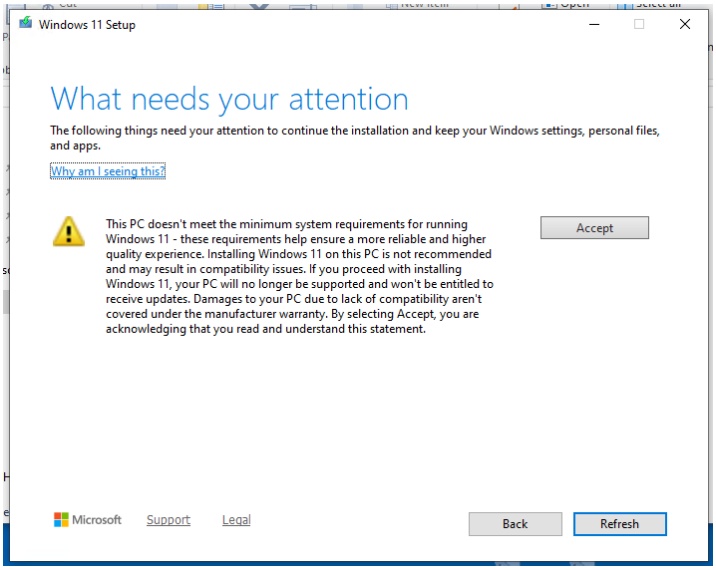
For virtual machines, this includes verifying that Secure Boot and UEFI are enabled, that a virtual TPM device is present, and that the system has sufficient vCPU, RAM, and disk space allocated. These settings are typically configured within the Hyper-V Manager prior to initiating the upgrade.

Compatibility can be assessed using Microsoft's PC Health Check tool or through manual review of system specifications. In cases where hardware does not meet official requirements—such as older CPUs or missing TPM modules—a registry bypass can be applied to allow the upgrade to proceed.

Apply the official Microsoft registry modification as follows:

1. Open Registry Editor and navigate to `HKEY_LOCAL_MACHINE\SYSTEM\Setup\MoSetup`
2. If the `MoSetup` key does not exist, create it.
3. Add a new DWORD (32-bit) Value with the following details:
    - Name: `AllowUpgradesWithUnsupportedTPMOrCPU`
    - Value: 1

This registry setting allows the upgrade to proceed but will trigger a warning message during the Windows 11 setup process:



Accepting this warning allows the upgrade to proceed, with the understanding that the system will operate outside of Microsoft's supported configuration.

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.

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 for Installation (Official)

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

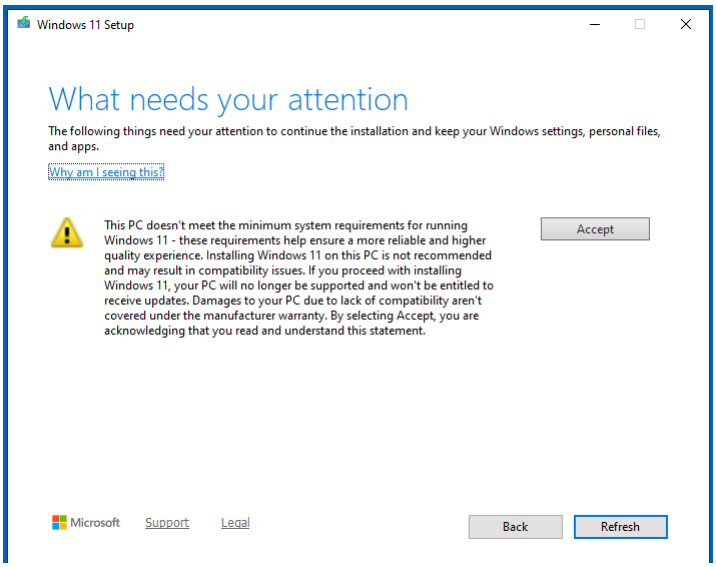
- Processor: 1 GHz or faster with 2 or more cores on a compatible 64-bit processor or System on a Chip (SoC)
- RAM: 4 GB or more
- Storage: 64 GB or larger storage device
- Firmware: UEFI, Secure Boot capable
- TPM: Trusted Platform Module (TPM) version 2.0
- Graphics card: Compatible with DirectX 12 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 VM Between Hypervisors

Migrating Windows 11 Gen 2 virtual machines between Hyper-V hosts involves varying levels of complexity depending on the host OS, configuration version, TPM settings, and Secure Boot policies. This section outlines four common scenarios based on whether Windows 11 requirements must be retained and whether hosts differ in OS level.

9.5.1 Migration Between Server 2019 Hosts Preserving TPM/Secure Boot

Scenario: Move a Windows 11 VM that was created with and must retain Windows 11 minimum requirements (TPM, UEFI, Secure Boot) from one Hyper-V host running Windows Server 2019 Datacenter to another.

Steps:

1. Export the VM from the source Hyper-V host.
2. On the destination Hyper-V host, create a new Generation 2 VM with the same VM name. Make sure the new VM settings meet Windows 11 minimum requirements for installation, except
    - do not enable TPM yet
    - do not attach a system disk (Hard Disk), instead attach the exported .vhdx file as the system disk.
3. Run the following PowerShell commands in an elevated window to reassign the local key protector:

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

4. Start the VM.

9.5.2 Migration of Non-Compliant Windows 11 VM Between Server 2019 Hosts

Scenario: The VM was originally installed with full Windows 11 hardware requirements (TPM, Secure Boot, etc.), but has since been modified—TPM is disabled, Secure Boot is turned off, and

there is no intention to re-enable those features. Both source and destination Hyper-V hosts are running Windows Server 2019 Datacenter, and the VM's configuration version is supported by both.

Steps:

1. On the source Hyper-V host, shut down the Windows 11 VM.
2. Check and verify that TPM and Secure Boot are disabled. If you make changes to settings, smoke-test (boot/login/shutdown) before exporting.
3. Use Hyper-V Manager to Export the VM to a shared location or removable storage.
4. On the destination Hyper-V host, open Hyper-V Manager and use Import Virtual Machine.
5. Complete the import wizard and start the VM.

Why this works:

Because the VM no longer depends on Windows 11 hardware requirements, it behaves like any other Generation 2 virtual machine. As long as both hosts use the same Hyper-V version (2019 Datacenter), the VM configuration remains valid and compatible across hosts.

No need to manually create a new VM or reattach the VHDX—the export/import process preserves all configuration settings.

Note: If the VM still has TPM and Secure Boot enabled at the time of migration, they can be manually disabled prior to export (using PowerShell or VM settings). This pre-export change allows the migration to proceed as described, ensuring compatibility with Server 2019 hosts.

9.5.3 Move Windows 11 VM Hosted on Windows 2019 to Windows 2016

Scenario: The Windows 11 VM was created on a Hyper-V host running Windows Server 2019 Datacenter, which supports Configuration Version 9.0 and full Windows 11 hardware requirements (TPM, Secure Boot, etc.). The destination host is a Windows Server 2016 Standard Hyper-V server, which supports only Configuration Version 8.0 and has more limited Secure Boot certificate handling. The goal is to successfully migrate the VM, knowing that Secure Boot and TPM will not be enforced or needed.

Assumption: TPM and Secure Boot enforcement are not required after migration.

Steps:

On the Source Host (Server 2019)

- Shut down the Windows 11 VM gracefully.
- In the VM's settings:
    - Navigate to Security.

    - Uncheck Enable Secure Boot.
    - Uncheck or remove any configured TPM settings.
- Export the VM
    - Right-click the VM → Export.
    - Choose an export location.

On the Destination Host (Server 2016)

- Copy the .vhdx file from the exported folder to the destination host.
- Open Hyper-V Manager.
    - Create a new Generation 2 VM:
    - Name: W11-SOL
    - Memory: 4096 MB (adjustable later)
    - Connect to a virtual switch
    - On the Connect Virtual Hard Disk screen:
        - Choose Use an existing virtual hard disk
        - Browse to the .vhdx file in the the export.
    - On the Summary, click Finish
- After creation:
    - Open the VM's settings
    - Go to Security
    - Ensure Secure Boot is unchecked
    - Confirm TPM is not enabled
    - Match other settings from vm on host server
- Start the VM and Verify Operation

Why This Works

Because Secure Boot and TPM are disabled prior to export, the VM no longer relies on certificates or key protectors tied to the original host. Windows Server 2016 supports Generation 2 VMs and can run Windows 11 without enforcing installation-time hardware requirements. The VHDX itself contains the OS and system files, and creating a fresh VM around it avoids compatibility conflicts with newer configuration versions.

9.6 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.

9.7 New Windows 11 VM With Windows 10 License

This section describes how to reuse a Windows 10 retail license key from a decommissioned computer to create a new Windows 11 virtual machine running inside a Hyper-V host. A common use case for this is when you want to use the free upgrade from Windows 10 to Windows 11 on a physical Windows 10 computer that doesn't meet the minimum requirements for Windows 11. It assumes you no longer need the physical computer to continue to function in its current role – all data and installed application on the physical computer will be lost and you will end up with a pristine Windows 11 machine.

There will be four high level steps

1. Create a VM
2. Install Windows 10 using the retail license key from the physical computer
3. Run the free ISO based upgrade from Windows 10 to Windows 11

4. Replace the OS on the physical computer

9.8 Windows 10 Physical Computer to Linux Mint

This section outlines how to repurpose a Windows 10 Pro physical laptop computer—commonly used by system administrators and developers to remotely access LAN resources—into a Linux Mint workstation that replicates RDP functionality. The converted system must support a core set of representative use cases to ensure continuity of operations.

- **User Login and Desktop Access** – the converted system must allow a user to log in and access a functional desktop environment with working input and display devices.
- **Network Connectivity and Remote Desktop Access** – the converted system must support stable RDP connections to key Windows-based systems used by sys admins and developers, connecting to other computers on both the primary wired network and the secondary Wi-Fi network. The converted system itself should also be accessible remotely through the network.
- **File System Operations** – the converted system can perform basic file management tasks on local and network drives.
- **Text File and Office Document Editing** – the converted system can edit and save plain text files using a graphical editor, and can edit and save Word and Excel documents.
- **Web Browsing** – the converted system can access internal and external websites through a web browser.
- **Audio and Video Usage** – the converted system can play media, records audio, and support video conferencing.

9.8.1 Installing Linux Mint

To begin transforming a Windows 10 Pro laptop into a Linux Mint-based RDP client, the existing Windows operating system must be replaced. This is typically done using a bootable USB stick prepared with the Linux Mint installer. After downloading the latest installation ISO from linuxmint.com, use a third-party tool such as Rufus to burn the ISO into a USB stick. The physical laptop is booted directly from USB, and the installation proceeds through Mint's graphical setup tool. During this process, the installer does not prompt the user to assign a hostname or configure network settings beyond basic Wi-Fi support. These tasks must be handled post-installation once the system is up and running. At this stage, Linux Mint becomes the primary operating system on the laptop, ready to be configured for remote access tasks.

9.8.2 Fixing Network and Hostname Configuration

After installation, the Ethernet interface must be verified and configured to properly integrate with the organization's LAN, including using the internal DNS servers hosted on domain controllers. This ensures the system can reliably resolve both internal and external domain names and participate effectively in local network operations. To configure this:

1. Open the Network Settings from the system tray and disable Wi-Fi to prioritize the wired Ethernet connection.
2. Edit the active wired connection and retain the IPv4 method as “Automatic (DHCP)” to allow dynamic IP assignment. Avoid static IPs to preserve flexibility and reduce administrative overhead.
3. Override the default DNS settings by manually entering the domain controller IP addresses along with the local router: 192.168.0.91, 192.168.0.92, and 192.168.0.1. This prevents the system from relying on inaccurate defaults.
4. Save the changes and reboot the system to apply new network settings.

To complete the foundational setup, assign the correct hostname to the laptop using the following command:

```
sudo hostnamectl set-hostname SOL
```

Replace SOL with the appropriate system name as needed. This sets the internal identifier used by the system and ensures alignment with the naming conventions used throughout the infrastructure.

9.8.3 Outbound RDP Connections – Remmina

To provide outbound RDP connectivity from Linux Mint to Windows workstations, the Remmina remote desktop client must be installed. Remmina is available in the standard Mint repositories and supports a variety of remote protocols including RDP, SSH, and VNC. Installation should include both the core application and the RDP plugin.

Install Remmina using the following command in the terminal:

```
sudo apt update
sudo apt install remmina remmina-plugin-rdp -y
```

Once installed, launch the application from the Mint menu and create a new connection profile. This profile simplifies future use by associating a machine name with its connection details.

To create a connection to a Windows machine on the LAN, enter the IP address (e.g., 192.168.0.93), along with the local Windows username (e.g., Administrator). The domain field should be left blank for machines that are part of a WORKGROUP rather than an Active Directory domain. Once the profile is saved, it can be reused with a single click, providing stable, full-screen remote access functionality similar to Windows RDP.

9.8.4 Inbound RDP Connections – XRDP

To install the XRDP service to allow inbound RDP sessions from windows computers

```
sudo apt update
sudo apt install xrdp -y
```

To enable and start XRDP on a Linux Mint computer

```
sudo systemctl enable xrdp
sudo systemctl start xrdp
```

To verify it's running

```
sudo systemctl status xrdp
```

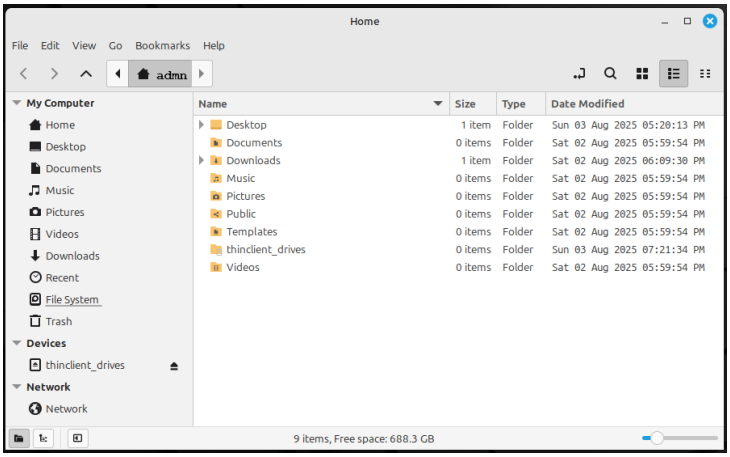
Potential security concern

Xrdp opens inbound port 3389 for inbound rdp connections. If the Linux Mint computer does not have a firewall such as UFW, and it connects to the Wi-Fi subnet, then a compromised device connected to the Wi-Fi subnet could attack the wired corporate network through the Linux Mint computer. Therefore in order to connect the Linux Mint computer to Wi-Fi, UFW must be installed and configured with an inbound rule blocking port 3389 from being accessed through the Wi-Fi subnet.
For this reason, Wi-Fi is not being set up for the Linux Mint computers at this time. A project scope extension will be needed for the following tasks

- Install UFW – this is a firewall for Linux Mint
- Install GUFW – firewall configuration GUI
- Use GUFW to configure UFW to block all inbound connections from Wi-Fi
- Connect to Wi-Fi

9.8.5 Accessing the File System in Linux Mint

Linux Mint uses a file manager called Nemo, which serves as the equivalent of Windows File Explorer. It provides a graphical interface for browsing the local file system, launching applications, and accessing network resources.



To open Nemo, click the Files icon from the Start menu or taskbar. The interface is similar to Windows Explorer, with a sidebar for quick access to common locations and a main pane for folder contents. At the top, the location bar can be toggled into a text field by pressing Ctrl + L, allowing users to enter exact paths. For example, typing smb://file-01/MyShare will connect to a Windows file share named MyShare on the server file-01.

When connecting to a network share, Nemo may prompt for credentials. Users should select “Forget password immediately” unless they explicitly wish to save credentials for future sessions. Once connected, the share will appear in the sidebar and may also temporarily appear on the desktop as a mounted volume. These desktop icons disappear when the share is unmounted or the system is rebooted.

Nemo supports double-click launching of files, and integrates with applications such as LibreOffice for documents and Image Viewer for media. It is the recommended tool for file navigation and network access on Linux Mint workstations.

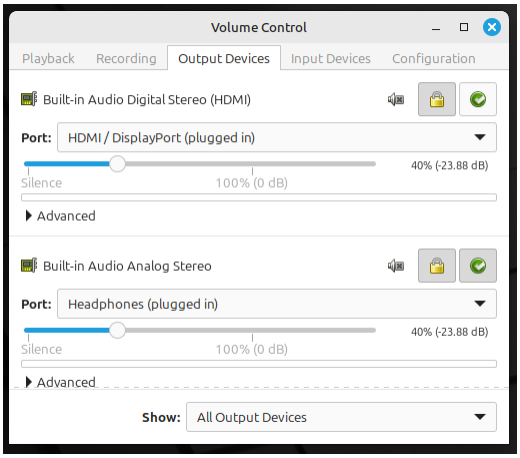
If credentials appear to be saved despite the “Forget password immediately” setting, users can force the system to forget them by deleting all files in the keyring directory. This can be done by opening a terminal and running

```
rm ~/.local/share/keyrings/*
```

which removes any saved passwords or authentication tokens associated with the current user. To ensure that no credentials remain cached in memory, it is recommended to reboot the system after performing this step. Once restarted, users can verify that the credentials have been cleared by reconnecting to the network share using Nemo; if prompted again for a username and password, the previous credentials have been successfully forgotten.

9.8.6 Managing Sound

To enable sound input and output on Linux Mint, we installed a set of packages that manage audio devices and provide recording functionality. The primary tool for device management is PulseAudio Volume Control,



installed via the command:

```
sudo apt install pavucontrol
```

This utility offers a graphical interface for configuring microphones, speakers, and other audio hardware.

For recording and playback, we used GNOME Sound Recorder,

installed with:

```
sudo apt install gnome-sound-recorder
```

These are referred to as packages in Linux terminology, though they may also be described as applications when used in a desktop environment.

By default, GNOME Sound Recorder stores audio files in a hidden directory located at

```
~/.local/share/org.gnome.SoundRecorder/
```

This path is not immediately visible in the file manager unless hidden files are revealed (typically with Ctrl + H). To simplify future access, we recommend creating a desktop shortcut to this folder once it's been located.

9.8.7 Linux Mint Backup and Restore with Timeshift

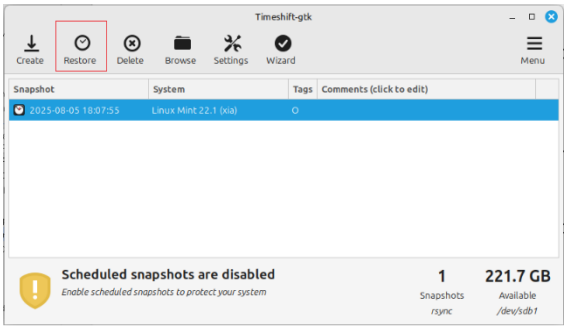
Reliable backups are essential for maintaining system stability, especially when making configuration changes, installing new software, or applying updates. In our Linux Mint deployment, we rely on Timeshift, which comes pre-installed as part of the distribution. It provides snapshot-based backups that preserve both system files and user data, allowing us to revert to a known-good state if needed.

Timeshift comes pre-installed with Linux Mint and is fully integrated with the system. During snapshot creation, it is critical to ensure that personal files are included by selecting the /home option from the dropdown in the setup or snapshot screen. By default, Timeshift excludes user data under /home, which would result in an incomplete backup. Snapshots were manually triggered before major changes, serving as milestone checkpoints in our workflow.

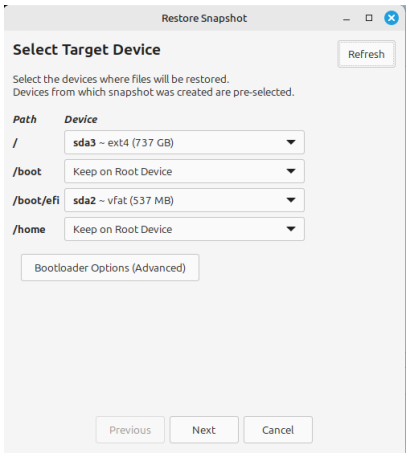
Restoring from a snapshot can be unintuitive. After rebooting, Timeshift may fail to restore user data unless specific steps are followed. While testing restores, we encountered failures where the process silently aborted after the "dry run" phase, especially immediately after rebooting the system. Repeating the restore a few times typically resolved the issue. We also identified some other specific actions that made success more likely. We discovered that opening the Bootloader Options (Advanced) dialog and immediately closing it without making any changes often triggers the full restore wizard. Additionally, on the Target Device screen, change the /home dropdown from “Keep on Root Device” to the actual system partition, e.g. “sda3 – ext4 (737 GB)”. Once these steps were followed, the restore succeeded and returned the system to the exact state captured in the snapshot.

Steps to Restore a Snapshot Successfully

After launching Timeshift and selecting the desired snapshot, follow these steps.



1. Click Restore.
2. On the Target Device screen, change /home from “Keep on Root Device” to the correct partition.



3. Click Bootloader Options (Advanced) and then Close the dialog without making changes.
4. Click Next and verify that the Confirm Actions screen appears.
5. Click Next again to begin the restore process.
6. After completion when the system has rebooted, confirm that both system and user data have been restored.

For step 2, to identify the correct partition that currently holds /home, run the following command in a terminal:

```
df -h /home
```

This will return output similar to:

```
Filesystem      Size  Used Avail Use% Mounted on
/dev/sda3       687G   12G  640G   2% /
```

Look for the partition where the MOUNTPOINT is / this is your root system.

If the restore silently fails, repeat the steps. Sometime multiple tries are needed, but eventually it should work.

9.8.8 Securing Wi-Fi in Linux Mint Using a Firewall

The wi-fi subnet has a larger attack surface than the company’s internal domain which is behind a wired router. Since Linux Mint doesn’t come with Windows Defender Firewall, we need to use a third-party firewall to secure the Linux Mint laptops when exposing them to the wi-fi subnet. A simple, easy-to-use firewall available for Linux Mint is ufw and gufw. These two packages

should come with Linux Mint by default. But if you need to install them, run the following in a Linux Mint terminal.

```
sudo apt install ufw
sudo apt install gufw
```

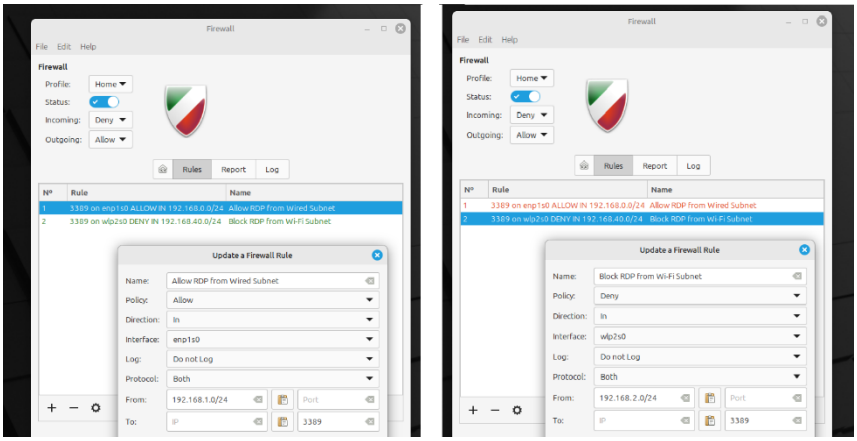
This installs the Uncomplicated Firewall (UFW) and its graphical user interface (GUFW), which together provide a simple firewall experience similar to Windows Defender Firewall.

Next, enable them.

```
sudo ufw status
sudo ufw enable
```

To further harden the laptop's network posture while connected to both wired and wireless interfaces, GUFW, the graphical frontend for UFW, can be used to define interface-specific firewall rules. This approach allows us to selectively permit or deny Remote Desktop Protocol (RDP) traffic based on both the source subnet and the network interface.

The rules are created using the Advanced tab in GUFW, which provide granular control beyond what is available in the Preconfigured or Simple tabs. This tab allows us to explicitly define the source IP range, destination port, interface and protocol to which the rule applies. The use of GUFW's Advanced tab provides a low-code, GUI-driven method for implementing precise firewall rules without requiring terminal commands or manual edits to configuration files.



We only need to define inbound rules, since the goal is to protect the laptop from unsolicited traffic originating from the unsecured Wi-Fi segment of the network. Outbound traffic from the laptop is already governed by default policies that are appropriate for our use case, and we are explicitly allowing trusted computers on the secure wired subnet to initiate connections. This approach ensures that the laptop remains accessible to authorized systems while remaining shielded from potentially hostile traffic.

Additionally, we specified the appropriate network interface for each rule—either the wired interface (enp1s0) or the wireless interface (wlp2s0)—to ensure that the firewall behavior was correctly scoped to the physical connection in use. This configuration allows us to enforce differentiated access controls based on network topology, ensuring that RDP traffic is permitted only from trusted sources and blocked from untrusted ones.

9.8.9 Regression Tests for a Linux Mint Workstation

9.8.9.1 User Login and Desktop Access

Log in to the laptop using a valid user account and confirm that the desktop environment loads successfully. Verify that the keyboard, mouse, and display are responsive and functioning.

Expected: The login completes without error, the desktop appears within 10 seconds, and the keyboard, mouse, and display all respond promptly to user input.

9.8.9.2 Wired Network Access and RDP Connectivity

Connect the laptop to the primary wired network using an Ethernet cable. Confirm that the system obtains a valid IP address and can access internal and external websites. From another system on the wired network, attempt to connect to the laptop using RDP.

Expected: The laptop connects to the wired network within 10 seconds and can browse websites. RDP connections from other systems on the wired subnet are accepted.

9.8.9.3 Wi-Fi Network Access and RDP Blocking

Connect the laptop to the secondary Wi-Fi network using the designated SSID and credentials. Confirm that the system obtains a valid IP address and can access internal and external websites. From another system on the Wi-Fi network, attempt to connect to the laptop using RDP.

Expected: The laptop connects to the Wi-Fi network within 15 seconds and can browse websites. RDP connection attempts from other systems on the Wi-Fi subnet are blocked by the firewall.

9.8.9.4 File System Operations

Open the file manager and navigate to the local file system (e.g., /home/sysadmin/) and a mounted network share (e.g., smb://MyFileServer/MyShare/). Create a new folder in each location and place a test text file inside.

Expected: The system successfully creates folders and files in both the local directory and the network share. No permission errors or delays occur during file creation, access, or deletion.

9.8.9.5 Text File and Office Document Editing

Open a plain text file (e.g., /home/sysadmin/test.txt) using a text editor such as Xed or Gedit. Make a small edit, save the file, and verify the changes persist. Then open an office document (e.g., .docx or .xlsx) using LibreOffice Writer or Calc, make a minor edit, save, and reopen to confirm the changes.

Expected: The system successfully opens, edits, and saves both text and office document files. Changes persist after closing and reopening the files, and no errors or unexpected behavior occur during the editing or saving process.

9.8.9.6 Web Browsing

Open the default web browser (such as Firefox or Chrome), navigate to a well-known external website (e.g., https://www.google.com), and verify that the page loads fully. Next, navigate to an internal company web resource, such as the intranet homepage or internal documentation site, and verify successful loading and access.

Expected: Both external and internal websites load completely without errors, and basic browsing functions such as clicking links and navigating pages work as intended.

9.8.9.7 Audio and Video Usage

Play a video with sound (for example, a YouTube video) using the default media player or web browser and verify that audio plays clearly through the speakers. Next, use a simple sound recording application to record a short audio clip via the built-in microphone and then play back the recording to verify audio capture and playback. Finally, launch a video conferencing application (such as Zoom or Teams) and verify that both video and audio are functioning during a test call or meeting.

Expected: Audio playback is clear and audible through local speakers; audio recording captures sound correctly and playback functions properly; video conferencing shows both audio and video without issues.

9.8.9.8 Printing

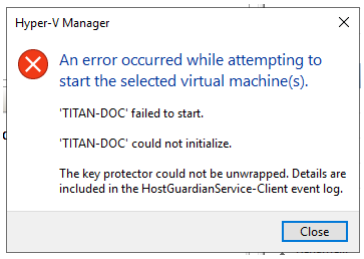
Open a document and initiate a print job using the system's default print dialog. Select a known working printer from the list and send the job. Confirm that the printer receives the job and produces a physical copy with correct formatting and content.

Expected: The system successfully connects to the selected printer, sends the print job without error, and produces a correctly formatted physical document. The print dialog displays available printers, and no driver or connectivity issues occur during the process.

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 oop

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.

nless 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.

9.9.3 Problems Encountered with Licensing Transfer and Upgrade Path

The “Saturn” physical laptop upgrade required a nonstandard multi-step approach due to licensing and media compatibility issues. Saturn's valid Windows 10 Enterprise retail key was reassigned to a VM on the hypervisor “Sirius” after repurposing the physical device for Linux Mint. The original installation media (version 1511) was outdated and incompatible with direct Windows 11 upgrades—even with the unsupported hardware bypass registry entry in place.

To resolve this, the upgrade followed four critical steps:

1. Bypass Windows 10 Enterprise v1511 entirely.
2. Install Windows 10 Enterprise v22H2 via a retail-compatible ISO created using the Media Creation Tool and license key.
3. Apply the `AllowUpgradesWithUnsupportedTPMOrCPU` registry key to bypass hardware checks.
4. Perform in-place upgrade to Windows 11 Enterprise using a multi-edition ISO.

Post-upgrade activation initially failed but was resolved by manually re-entering the original key. Evaluation ISOs from Microsoft’s Evaluation Center were avoided, as they cannot be activated with retail keys or upgraded to Windows 11.

This case illustrates how legacy media and mismatched editions can block upgrade paths and highlights the need for edition-aware install media when reusing retail licenses.

9.9.4 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.

9.9.5 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.

9.9.6 Linux Troubleshooting Tools

Terminal Equivalents to Windows Tools

Windows Command	**Linux Mint Equivalent**
Ipconfig	ip a or ifconfig (ifconfig may require net-tools)
ping	ping <hostname/IP> (same syntax)
tracert	traceroute <hostname/IP> (may need install)
nslookup	nslookup <hostname> or dig
netstat	ss -tuln or netstat -tuln (install net-tools)
sysdm.cpl	hostnamectl or System Info (GUI)

9.9.7 Linux Mint Outgoing RDP Missing

To make outgoing RDP connections from Linux Mint, you need to install Remmina.

```
sudo apt update
sudo apt install remmina remmina-plugin-rdp -y
```

Launch Remmina from the Start menu or by running remmina in the terminal.

Create a new RDP connection

- Click the + (New Connection)
- Set the following:
    - Protocol: RDP – Remote Desktop Protocol
    - Server: sirius.d1.internal or its IP
    - Username: your Windows username (e.g., d1\robidas or .\Administrator)
    - Password: (optional – can prompt each time)
- Save and double-click to connect.

9.9.8 Resolving a Failed Remote RDP to Linux Mint by Forced Remote Reboot

Error: RDP from a Windows computer to a Linux Mint machine fails silently, returning immediately to the local desktop with no error message.

Problem: XRDP cannot launch a remote GUI session when a local console session is active on the Linux Mint computer. The RDP client disconnects immediately without feedback.

Root Cause: Active console sessions on Linux Mint block incoming XRDP connections. If physical access to log out is unavailable, remote intervention is required.

Solution Steps:

On the Linux Mint system, install and start the SSH service:

```
sudo apt install openssh-server -y
sudo systemctl start ssh
```

From any Windows computer, open CMD or PowerShell and run:

```
ssh -t <linuxmint-username>@<linuxmint-ip-address> sudo reboot
```

Replace placeholders with the actual username and IP. Enter the password when prompted. The Linux Mint system will reboot, closing the active console session and allowing XRDP to accept future remote connections.

Security Note: SSH listens on port 22. When connecting a Linux Mint system to an untrusted or external network (e.g., Wi-Fi), configure a firewall to restrict SSH and XRDP access to trusted IP ranges. See Section 13.9 for configuration details.

9.9.9 TPM Key Protector Cannot be Unwrapped

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.

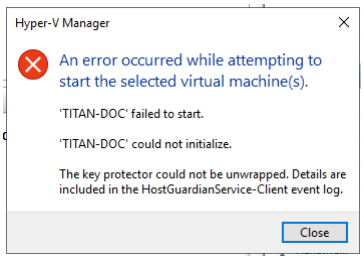
1. Export the VM on the Source Hypervisor

- 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

9.9.10 Internet Access Lost When Moving Hypervisor to Protected Wifi Subnet

Error 1: Network icon shows that there is no internet access. Network troubleshooter reports

- There might be a problem with the driver for the Wi-Fi adapter Not fixed
- Windows couldn't automatically bind the IP protocol stack to the network adapter.

Resolution:

Open Command Prompt as Administrator and run these commands one by one

```
netsh winsock reset
netsh int ip reset
ipconfig /release
ipconfig /renew
ipconfig /flushdns
```

Then restart the computer.

Error 2: unable to find router

Resolution: Add the computer's MAC address to the router's whitelist in the advanced > network filter list.

9.9.11 Loss of Local Audio After RDP Connection to Linux Mint Laptop

Error: local sound stops working after connecting via RDP from a Windows 11 computer

After remotely connecting to a Linux Mint laptop (e.g., jupiter) via RDP from a Windows 11 Pro system (e.g., galaxy), users may discover that local sound playback is no longer functional. The issue does not appear immediately after the remote session ends, but becomes evident after the Linux system is restarted and logged into locally.

Cause: The issue is caused by a session startup script that launches a virtual sound redirection service for XRDP. Specifically, the file `/etc/xdg/autostart/pipewire-xrdp.desktop` is responsible for starting pipewire-xrdp, which creates a virtual playback device named xrdp-sync. This device is intended to route audio from the Linux session back to the remote RDP client. However, when the session is restarted locally, xrdp-sync can remain active or be reinitialized, overriding the default sound output device and preventing audio from reaching the local speakers.

This behavior is problematic in environments where remote sound playback is not required, as it interferes with normal desktop audio even after the remote session has ended.

Resolution: To prevent xrdp-sync from launching during session startup, the autostart script can be safely disabled by renaming it

```
sudo mv /etc/xdg/autostart/pipewire-xrdp.desktop /etc/xdg/autostart/pipewire-xrdp.desktop.disabled
```

Renaming the file disables its execution without deleting it, allowing easy reactivation if needed in the future. This change is harmless in our environment, as we do not rely on XRDP sound redirection. Disabling the script restores normal audio routing to the local playback device and does not affect other system functions.

Testing & Verification: To confirm the fix, we reproduced the issue and verified the resolution through the following steps:

1. Initiated an RDP session from galaxy to jupiter and logged in remotely.
2. Logged out and restarted jupiter, then logged in locally and confirmed that sound playback was no longer working.
3. Renamed the autostart script using the command shown above.
4. Restarted jupiter and logged in locally again — sound playback was restored.
5. Repeated the RDP session from galaxy and verified that local sound remained functional after the session ended.

Note: This workaround disables remote sound playback during XRDP sessions, which is acceptable for our Linux Mint laptops. If remote audio becomes necessary in the future, the script can be re-enabled by renaming it back to its original filename.

Optional Tool for Diagnosing Audio Issues: pavucontrol (PulseAudio Volume Control), a graphical mixer tool for the PulseAudio sound server. It allows users to:

- View and adjust volume levels for individual applications
- Select and switch between output devices (speakers, headphones, etc.)
- Manage input devices like microphones
- Redirect playback streams to different devices mid-session It's especially useful for identifying when xrdp-sync has hijacked the default output device. You can install it via terminal:

9.9.12 Linux Nemo File Manager Authentication Failure

Error

When attempting to access a Windows network file share through Nemo, the following message appears

Could not display "smb://file-01/temporary/".

Error: Failed to mount Windows share: Invalid argument

Please select another viewer and try again.

[OK]

This error means that Nemo (through GVFS, the background service that handles network mounts) was unable to mount the Windows share. The phrase *“Invalid argument”* does not point to a specific problem—it simply indicates that something in Nemo’s request to the SMB client was rejected. The “Please select another viewer” part suggests Nemo has no working handler available to display the share.

Cause

This issue occurs when cached metadata or state information inside GVFS becomes corrupted or outdated, typically after a system sleep, network disruption, or configuration change. Although the share itself is reachable and DNS resolution works, Nemo fails to negotiate the mount because it is reusing stale or invalid information. In other words, the problem is not with the file server but with Nemo’s background mounting process.

Resolution

To resolve the problem, bypass Nemo's cached state and re-establish the connection manually. This can be done with the `gio mount` command, which directly tells GVFS to mount the share using fresh parameters. For example

gio mount smb://file-01.d1.internal/temporary/

You will be prompted for credentials (domain, username, password). Once successful, the share appears again in Nemo's sidebar and functions normally. This clears the cached or corrupt state that caused the error.

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 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.
- **Windows 11 upgrades deployed across eligible machines**: All Windows 10 systems meeting minimum requirements were upgraded to Windows 11, improving security posture, boot performance, and compatibility with modern management tools. In-place upgrades were validated to preserve user profiles and application configurations, minimizing disruption.
- **Linux Mint adopted for legacy hardware**: Physical laptops with Windows 10 unable to support Windows 11 were transitioned to Linux Mint, extending device usability while reducing licensing costs. Core productivity functions were preserved, and domain integration was bypassed in favor of local account management.

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.
- **Hardware limitations shaped OS deployment decisions**. Early compatibility checks identified systems unsuitable for Windows 11 upgrades, leading to Linux Mint replacements as an effective alternative.

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 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.

13.2 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).

13.3 Implement Speech Recognition on Linux Mint Workstations

To improve accessibility and reduce time spent on manual text input, evaluate and implement speech-to-text and screen-reading solutions for Linux Mint laptops. Candidate solutions should be tested for accuracy, integration with existing applications, and minimal performance impact.

13.4 Linux Mint Software and Peripheral Validation

To ensure Linux Mint workstations fully meet user needs, additional validation of optional software packages and peripheral hardware is recommended. This includes testing webcams, printers, and other accessories that may be required for conferencing or remote access. Evaluating compatibility and usability will help standardize configurations and reduce support overhead.

13.5 Moving the Canon Printer to Linux

As part of the broader infrastructure realignment, the Canon TS5300 printer currently hosted on TITAN should be migrated to SATURN, a Linux Mint computer. This change will reduce reliance on TITAN, which will be reserved for other roles.

SATURN will be configured to host the printer using the Common Unix Printing System (CUPS), with the printer physically connected via USB and shared over the network using Samba. Linux printer drivers will be installed on SATURN to ensure proper operation and compatibility with CUPS.

When printing from Windows clients to a printer hosted on a Linux system, the client machines do not require Linux drivers. Instead, Windows uses its own Canon TS5300 driver to generate print jobs, which are then transmitted to the CUPS server on SATURN. CUPS interprets and processes these jobs using its own Linux-based drivers. As long as the printer is correctly shared and advertised over the network, and the drivers on both ends are compatible with the printer's capabilities, cross-platform printing should function reliably without requiring Linux drivers on Windows clients.

This migration will simplify cross-platform support, reduce administrative overhead, and improve service reliability.

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
- **Download the latest Windows 11 ISO from the official Microsoft page:**
  https://www.microsoft.com/en-ca/software-download/windows11
- **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