Report: Privileged Access Management (PAM)

1. Introduction

Privileged Access Management (PAM) is a fundamental component of a robust cybersecurity strategy, focusing on controlling and securing administrative privileges within an organization's IT environment. Administrative accounts inherently possess elevated permissions that, if misused or compromised, can lead to severe security breaches including unauthorized access, data theft, or disruption of critical services. To address these risks, organizations adopt a multi-layered approach to PAM that incorporates technologies and processes designed to minimize the scope and duration of privileged access while ensuring administrative tasks can still be effectively performed.

Just Enough Administration (JEA) is a Microsoft security technology that enforces the principle of least privilege by allowing helpdesk and support staff to perform specific administrative functions through constrained PowerShell endpoints. This minimizes the need to grant full administrative rights, reducing the risk of privilege abuse or accidental misconfiguration. Complementing this, time-limited administrative accounts provide temporary elevated access, automatically expiring after a set period, ensuring that privileged credentials are not active longer than necessary. This reduces the risk window for credential compromise and limits opportunities for unauthorized activity.

Further strengthening the security posture, Privileged Access Workstations (PAWs) are dedicated, hardened workstations or virtual machines used exclusively for sensitive administrative tasks. By isolating administrative activities from general user environments, PAWs protect privileged credentials and sessions from malware, phishing attacks, and lateral movement threats that often target admin accounts on standard desktops. Together, these PAM controls help create a secure, manageable, and auditable administrative environment that balances operational needs with security imperatives.

2. Objective

The primary objective of implementing Just Enough Administration (JEA), time-limited administrative accounts, and Privileged Access Workstations (PAWs) is to strengthen the security and management of privileged access within an organization. These controls collectively enforce the principle of least privilege by limiting administrative users to only the necessary permissions and reducing the time window during which elevated privileges are active. Additionally, by isolating sensitive administrative tasks on dedicated, hardened workstations, the risk of credential theft and compromise is significantly lowered. This approach not only reduces the attack surface and prevents misuse or abuse of administrative rights but also improves accountability and auditing capabilities, ensuring that privileged activities are closely monitored and controlled. Ultimately, these measures help protect critical systems and data from unauthorized access, insider threats, and external attacks targeting privileged accounts.

Key goals include:

- **Enforce Least Privilege Access**
  Limit users' administrative permissions strictly to the tasks they need to perform, preventing unnecessary or excessive rights.

- **Reduce Privileged Access Exposure**
  Minimize the duration that administrative accounts remain active by using time-limited accounts that automatically expire.

- **Enhance Security of Admin Environments**
  Isolate sensitive administrative operations on dedicated, hardened Privileged Access Workstations to protect credentials from malware and phishing attacks.

- **Improve Accountability and Auditing**
  Ensure all privileged actions are logged and auditable to track administrative activities and detect potential misuse.

- **Mitigate Risk of Credential Theft and Abuse**
  Lower the likelihood of attackers gaining persistent privileged access by combining just-in-time access controls and secure administration environments.

3. Methodology

Modern Active Directory (AD) environments are prime targets for attackers seeking to escalate privileges or persist in a compromised network. To mitigate risks such as credential theft and privilege misuse, organizations must implement strong identity and access controls. The following tasks—Just Enough Administration (JEA), Time-Limited Admin Accounts, and Privileged Access Workstations (PAWs)—represent a layered defense strategy that enforces the principle of least privilege and operational containment.

3.1 JEA Implementation and Configuration

This phase involves implementing Just Enough Administration (JEA) to restrict helpdesk staff privileges by creating a controlled PowerShell environment that limits their access to only the necessary cmdlets and functions required for their tasks. By defining role capabilities within a PowerShell module, configuring a session configuration file, and registering a dedicated JEA endpoint, helpdesk users are granted minimal administrative rights following the principle of least privilege, thereby enhancing security and reducing the risk of unauthorized actions.

- **Prepare Role Capability**
  Create a JEA role capability file (`Helpdesk.psrc`) defining permitted cmdlets and functions.
  Place this .psrc file inside a RoleCapabilities folder within a PowerShell module directory (C:\ProgramFiles\WindowsPowerShell\Modules\HelpdeskRole\1.0.0\RoleCapabilities\Helpdesk.psrc).

```
PS C:\Users\Administrator> $modulePath = "C:\Program Files\WindowsPowerShell\Modules\HelpdeskRole\1.0.0"
PS C:\Users\Administrator>
PS C:\Users\Administrator> New-Item -Path "$modulePath\RoleCapabilities" -ItemType Directory -Force


    Directory: C:\Program Files\WindowsPowerShell\Modules\HelpdeskRole\1.0.0

Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         5/16/2025   5:41 AM                RoleCapabilities


PS C:\Users\Administrator>
PS C:\Users\Administrator>
PS C:\Users\Administrator>
PS C:\Users\Administrator> Move-Item -Path "$modulePath\Helpdesk.psrc" -Destination "$modulePath\RoleCapabilities\Helpdesk.ps
rc" -Force
PS C:\Users\Administrator>
```

- **Create PowerShell Module Manifest**

  Generate a module manifest (HelpdeskRole.psd1) in the module folder to make PowerShell recognize the module and role capability.

```
Administrator: Windows PowerShell
PS C:\Users\Administrator> New-ModuleManifest -Path "C:\Program Files\WindowsPowerShell\Modules\HelpdeskRole\1.0.0\HelpdeskRo
le.psd1" `
>>   -RootModule "" `
>>   -Guid ([guid]::NewGuid()) `
>>   -Author "JEA Setup" `
>> -Description "Helpdesk JEA Role Module" `
>> -ModuleVersion "1.0.0"
PS C:\Users\Administrator>
```

- **Create JEA Session Configuration**

  Use New-PSSessionConfigurationFile to create a session configuration file referencing the role capability.

  Assign the helpdesk Active Directory group (e.g., `YASH\IT_Helpdesk`) permissions to use this role capability.

```
PS C:\Users\Administrator> New-PSSessionConfigurationFile -Path "C:\JEA\HelpdeskSession.pssc" `
>>   -SessionType RestrictedRemoteServer `
>>   -RunAsVirtualAccount `
>> -RoleDefinitions @{ 'YASH\IT_Helpdesk' = @{ RoleCapabilities = 'Helpdesk' } }
PS C:\Users\Administrator>
```

- **Register JEA Endpoint**

  Register the session configuration as a JEA endpoint (e.g., HelpdeskEndpoint) on the target server (Domain Controller or others).

```
PS C:\Users\Administrator> Register-PSSessionConfiguration -Name HelpdeskEndpoint -Path "C:\JEA\HelpdeskSession.pssc" -Force

   WSManConfig: Microsoft.WSMan.Management\WSMan::localhost\Plugin

Type            Keys                                Name
----            ----                                ----
Container       {Name=HelpdeskEndpoint}             HelpdeskEndpoint
```

- **Validate and Test**

  Connect to the JEA endpoint remotely using Enter-PSSession and credentials of a helpdesk user.

```
PS C:\Users\Anjali.Patel> Enter-PSSession -ComputerName DC1 -ConfigurationName HelpdeskEndpoint -Credential (G
et-Credential)

cmdlet Get-Credential at command pipeline position 1
Supply values for the following parameters:
Credential
[DC1]: PS C:\Users\Anjali.Patel\Documents>
```


3.2 Create Time-Limited Admin Accounts

Provision administrative accounts that automatically expire after a predefined time period (e.g., 4 hours) to reduce standing privileged access and minimize attack surface.

- Identify admin accounts that require temporary elevated access.
- Use Active Directory (AD) or PowerShell to create or modify accounts with an expiration timestamp set to 4 hours from creation.
- Implement automation scripts or workflows to generate these time-limited accounts on demand.
- Monitor and audit the usage and expiry of these accounts to ensure compliance.

(Created a user.)

(Add details of the user.)

```
PS C:\Users\Administrator>
PS C:\Users\Administrator> New-LocalUser -Name $Username -Password $Password -FullName "Temporary Administrator" -Description
 "4-hour admin account"

Name      Enabled Description
----      ------- -----------
TempAdmin True    4-hour admin account
```

(Add in the Administrator group.)

```
PS C:\Users\Administrator>
PS C:\Users\Administrator> net localgroup Administrators YASH\TempAdmin /add
The command completed successfully.

PS C:\Users\Administrator> _
```

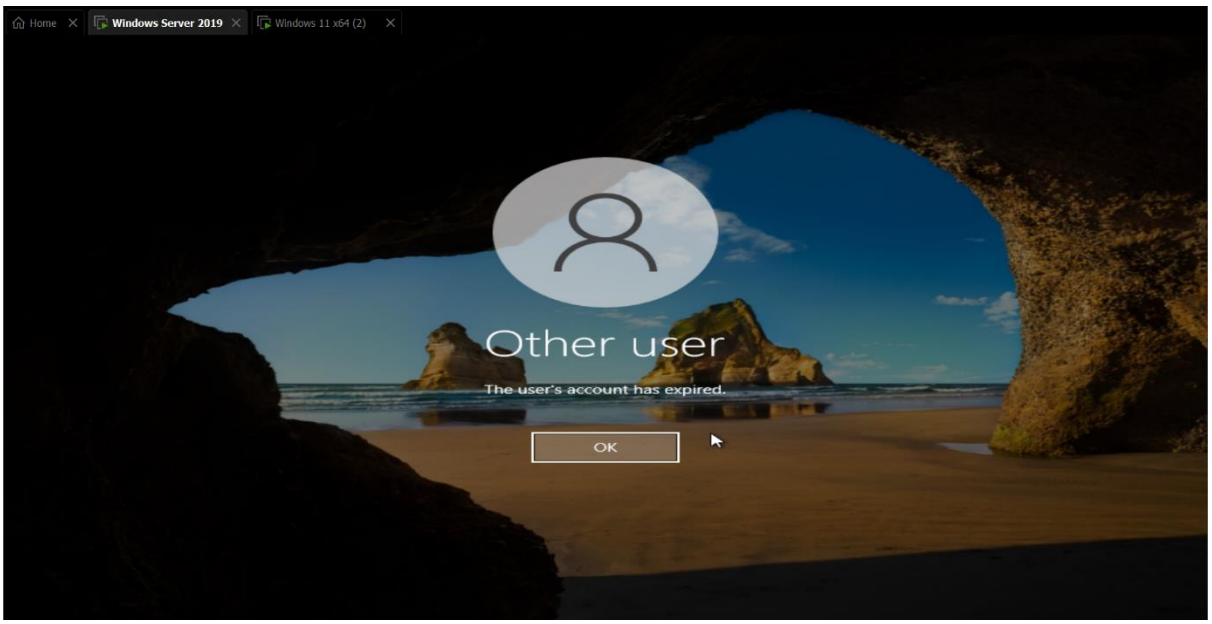
(Set the expiration time.)

```
Administrator: Windows PowerShell
PS C:\Users\Administrator> $expireTime = (Get-Date).AddHours(4)
PS C:\Users\Administrator>
PS C:\Users\Administrator> Set-LocalUser -Name $Username -AccountExpires $expireTime
PS C:\Users\Administrator>
PS C:\Users\Administrator> _
```

(Verify it.)

```
PS C:\Users\Administrator>
PS C:\Users\Administrator> Get-LocalUser -Name $Username | Select-Object Name, Enabled, AccountExpires

Name      Enabled AccountExpires
----      ------- --------------
TempAdmin    True 5/15/2025 7:46:41 AM


PS C:\Users\Administrator>
PS C:\Users\Administrator> _
```

(Again verified it by trying to login.)



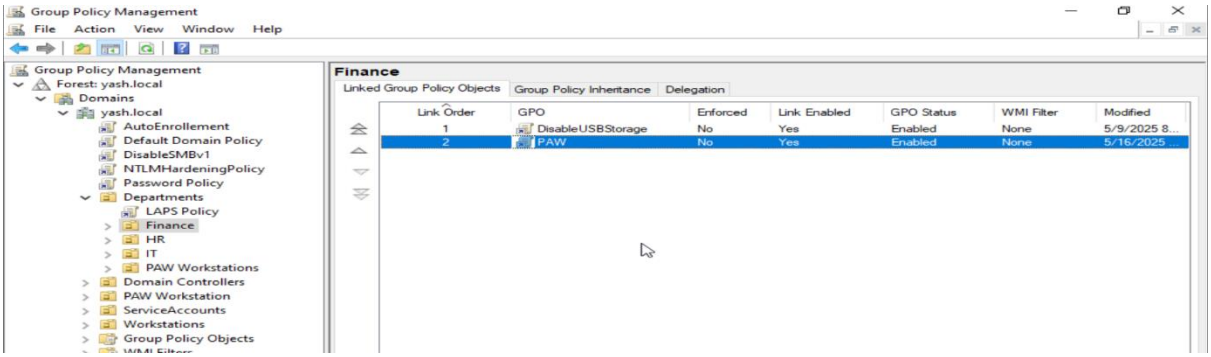
3.3 Set Up Privileged Access Workstations (PAWs)

A second domain-joined workstation was prepared to function as a PAW. A new security group called PAW was created in Active Directory, and the Domain Administrator account was added to this group.

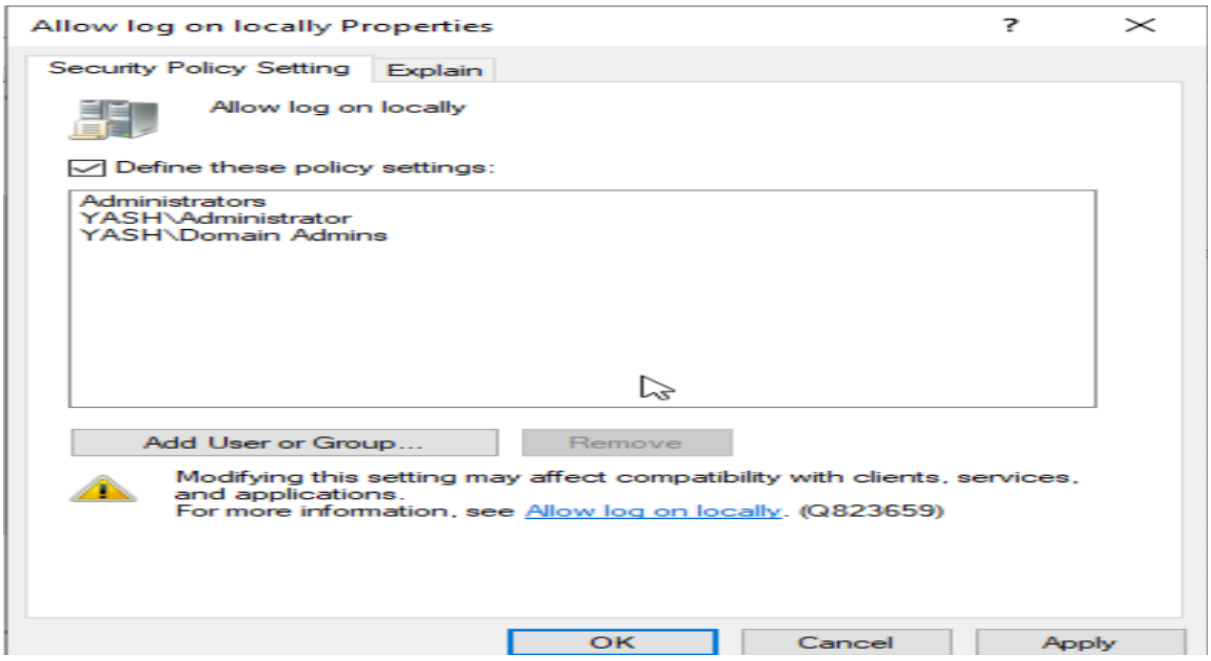
Group Policy was then configured to:

- Allow log on locally only to Administrator and Domain Users .
- Deny log on locally to Domain Users.
- This configuration ensured that only authorized admin accounts could use YASHWIN11. A login attempt using the Domain Admin account was successful while general users were denied access.

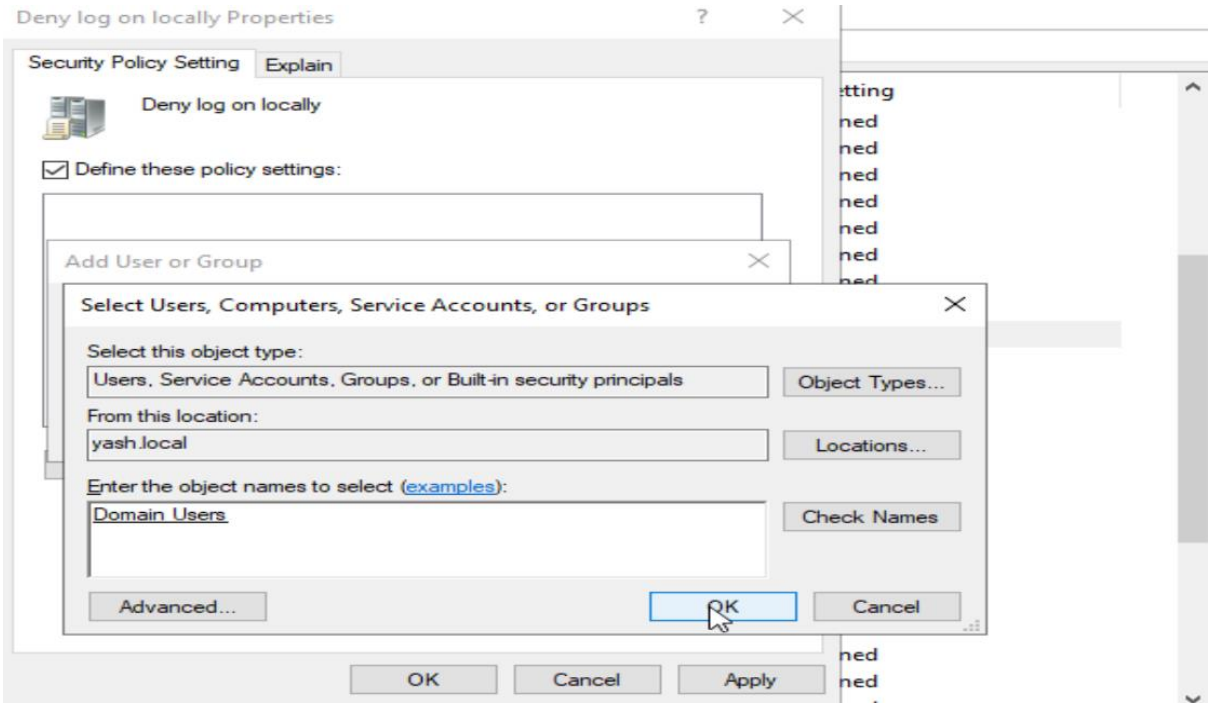
(Created a new Group policy.)



(Users that are allowed to login locally.)



(Users that are not allowed to log on locally.)



(Updated the policy.)

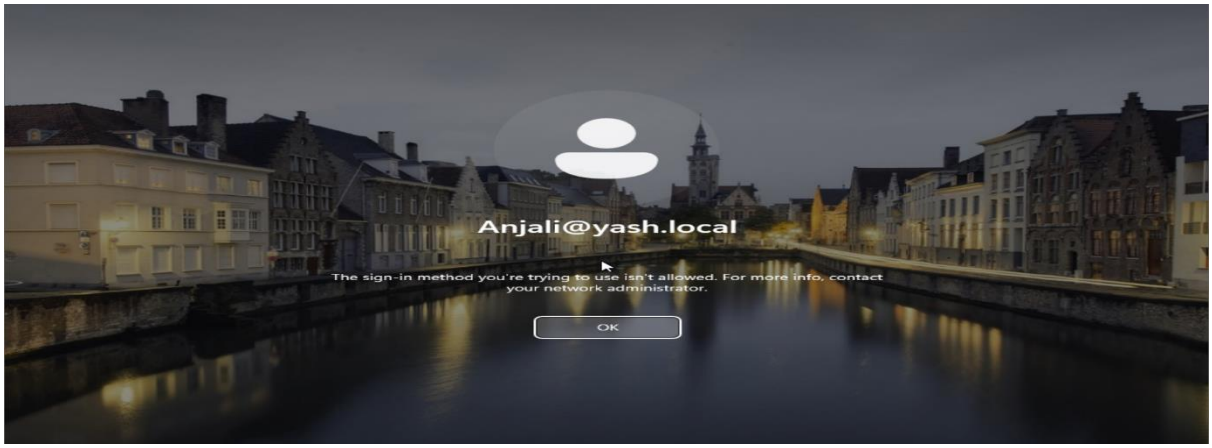
```
PS C:\WINDOWS\system32> gpupdate /force
Updating policy...

Computer Policy update has completed successfully.
User Policy update has completed successfully.
```

(We can see the policy name here.)

```
PS C:\WINDOWS\system32> gpresult /r

Microsoft (R) Windows (R) Operating System Group Policy Result tool v2.0
© Microsoft Corporation. All rights reserved.

Created on  5/ 16/ 2025 at 7:42:16 PM


RSOP data for YASHWIN11\yashv on YASHWIN11 : Logging Mode
------------------------------------------------------------

OS Configuration:            Member Workstation
OS Version:                  10.0.26100
Site Name:                   N/A
Roaming Profile:             N/A
Local Profile:               C:\Users\yashv
Connected over a slow link?: No


COMPUTER SETTINGS
------------------

    Last time Group Policy was applied: 5/16/2025 at 7:41:50 PM
    Group Policy was applied from:      DC1.yash.local
    Group Policy slow link threshold:   500 kbps
    Domain Name:                        YASH
    Domain Type:                        Windows 2008 or later

    Applied Group Policy Objects
    -----------------------------
        DisableUSBStorage
        PAW
        LAPS Policy
        Default Domain Policy
        Password Policy
        DisableSMBv1
        NTLMHardeningPolicy
        AutoEnrollement
```

(Try to login the user account which is member of Domain Users.)

(As we see log on failed that means our group policy is working.)



(Log in using the allowed accounts.)

(As we see log in successfully.)

4. Results And Findings

This section highlights the actual results observed after completing the tasks. It includes practical evidence of whether the configurations worked as intended — such as successful restrictions, system behavior, log outputs, and access limitations. It provides technical confirmation that each security control (JEA, temporary accounts, PAWs) functioned as expected in your environment.

4.1 JEA Restricted Helpdesk Permissions

Just Enough Administration (JEA) successfully allowed helpdesk staff to perform only specific PowerShell commands. Unauthorized or sensitive operations were automatically blocked.

4.2 Time-Limited Admin Accounts Expired as Planned

Temporary admin accounts were configured to expire after 4 hours. This controlled elevated privileges and minimized the window of potential misuse.

4.3 PAW Was Configured for Admin-Only Tasks

The server was set up to function as a Privileged Access Workstation (PAW), restricted to administrative users only. Regular user access and internet-facing features were avoided.

4.4 Admin Activity Was Logged and Traceable

All administrative activity through JEA and account usage was logged. This ensured traceability and improved auditing capabilities.

4.5 No Unnecessary Tools Were Present

Only essential tools like RSAT and PowerShell were available. No general-use or non-admin software was installed, reducing the system’s attack surface.

5. Recommendations

This section gives actionable suggestions for strengthening and scaling the setup further. While the initial configuration works, recommendations help apply it in larger or production environments with stronger security practices. It includes ideas like automating account cleanups, applying GPOs, and using dedicated devices for PAWs.

5.1 Extend JEA to More Admin Roles

Apply JEA to other support roles, customizing role capabilities for each job. This enforces least privilege across more users.

5.2 Monitor and Clean Up Expired Accounts

Regularly review and remove expired or disabled admin accounts. Automate cleanup using scripts to prevent account buildup.

5.3 Avoid Using Domain Controllers as PAWs in Production

In production environments, use dedicated workstations or virtual machines as PAWs. Avoid using DCs directly to reduce risk.

5.4 Enforce PAW Hardening via Group Policy

Use GPOs to disable unnecessary features (USB, internet access, etc.) on PAWs. This standardizes security across all privileged workstations.

5.5 Audit Admin Sessions Regularly

Review JEA logs and admin session logs frequently. This helps detect suspicious activity and improves incident response readiness.

6. Conclusion

In conclusion, the combined implementation of Just Enough Administration (JEA), time-limited administrative accounts, and Privileged Access Workstations (PAWs) provided a robust foundation for securing privileged access in the Windows Server environment. The JEA configuration successfully enforced fine-grained delegation by allowing helpdesk users to execute only specific, approved PowerShell commands — effectively reducing the attack surface and mitigating the risk of privilege escalation. The creation of time-bound admin accounts, designed to automatically expire after four hours, ensured that elevated access was strictly temporary, minimizing the potential for misuse or persistence by threat actors. Meanwhile, configuring a PAW on the domain controller allowed for secure administrative operations by restricting local logon rights, limiting installed tools to essential administrative utilities, and avoiding general-purpose software and internet access. Collectively, these tasks advanced the principle of least privilege, improved auditability through logging and access tracking, and enforced isolation of sensitive activities. These outcomes align with modern cybersecurity best practices and demonstrate a proactive approach to protecting high-value domain assets from internal and external threats.