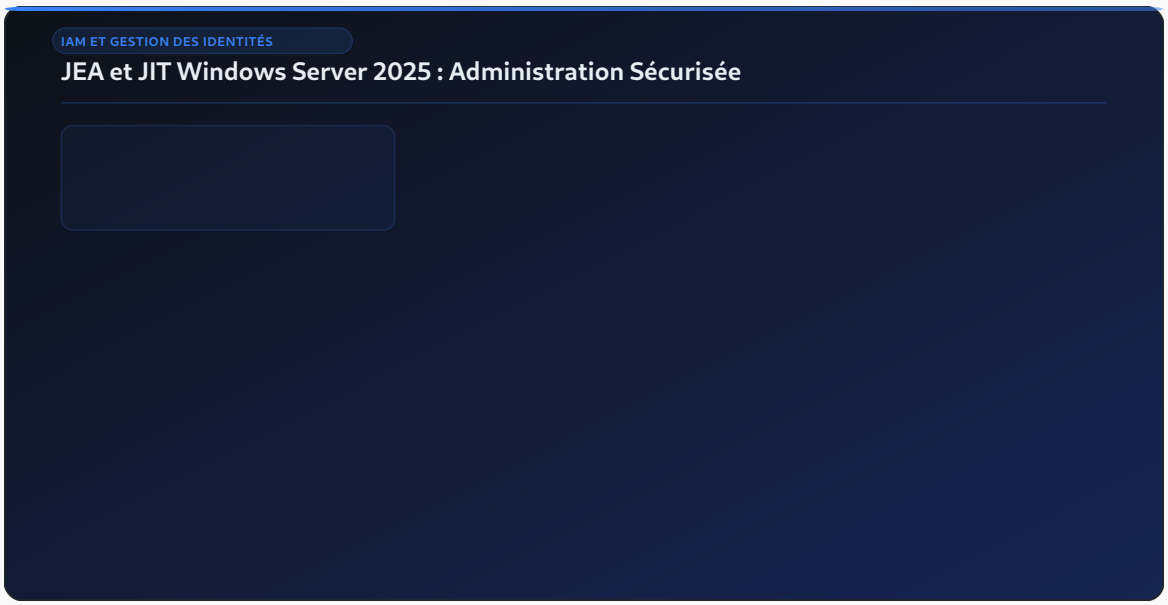
JEA et JIT : Administration Sécurisée Microsoft Windows Server 2025

Déployez JEA et JIT sur Windows Server 2025 pour une administration sécurisée Zero Trust : Role Capability Files, Entra PIM, audit PowerShell. Guide expert.

📋 EN BREF



- **Zero Trust natif** : la combinaison JEA+JIT élimine les comptes administrateurs permanents et réduit la surface d'attaque liée aux mouvements latéraux
- **Conformité NIS 2** : ces mécanismes répondent directement aux exigences de gestion des accès à privilèges imposées aux entités françaises essentielles et importantes
- **Audit complet** : ScriptBlock logging, transcription PowerShell et Event IDs dédiés permettent une traçabilité forensique exhaustive de chaque session d'administration

La compromission d'un compte administrateur représente aujourd'hui le vecteur d'attaque le plus dévastateur dans les infrastructures Microsoft. En France, les incidents de sécurité majeurs analysés par l'ANSSI en 2024 et 2025 révèlent que plus de 70 % des intrusions réussies impliquent une élévation de privilèges non contrôlée, souvent via des comptes disposant de droits administrateurs permanents et non surveillés. Face à cette réalité, Microsoft a considérablement renforcé ses outils d'administration sécurisée dans Windows Server 2025, en plaçant au cœur de sa stratégie deux mécanismes complémentaires : Just Enough Administration (JEA) et Just-in-Time (JIT). Ces approches, ancrées dans le paradigme Zero Trust, transforment radicalement la posture de sécurité des organisations. Pour les entreprises françaises soumises à la directive NIS 2, désormais transposée en droit national, la mise en œuvre de JEA et JIT n'est plus un choix architectural optionnel : c'est une exigence de conformité. Les RSSI des entreprises de la région parisienne et d'Île-de-France, ainsi que dans toute la France, font face à des obligations de traçabilité et de contrôle des accès à privilèges que ces outils permettent de satisfaire concrètement. Ce guide expert détaille l'architecture, le déploiement et la supervision de JEA et JIT dans un contexte Windows Server 2025, avec des exemples PowerShell prêts à l'emploi et une analyse de leur intégration dans une stratégie Zero Trust complète.

Pourquoi JEA et JIT sont devenus incontournables en 2025

Le modèle d'administration traditionnel repose sur un principe dangereux : pour administrer un serveur Windows, un compte doit être membre du groupe Domain Admins ou Administrators. Ces memberships permanents créent une fenêtre d'exposition continue. Si ce compte est compromis, l'attaquant dispose immédiatement de droits illimités sur l'ensemble de l'infrastructure Active Directory. Les attaques par Pass-the-Hash, Kerberoasting et DCSync exploitent précisément cette réalité.

> ⚡ **Retour terrain**
>
> Dans les projets IAM, le cas le plus fréquent que je rencontre est celui des comptes de service sans propriétaire identifié — créés pour un projet terminé ou un prestataire parti, et jamais supprimés. Pour une banque régionale, l'inventaire a identifié 134 comptes de service sans propriétaire actif, dont 23 avaient des droits administrateurs locaux sur des serveurs de production. La désactivation progressive de ces comptes sur 6 semaines n'a déclenché aucune anomalie applicative.

Windows Server 2025 intègre nativement des mécanismes pour éliminer ce paradigme. JEA permet de déléguer à un technicien de support uniquement les cmdlets PowerShell dont il a besoin — redémarrer un service IIS, consulter les logs d'événements, gérer les certificats — sans jamais lui accorder de droits administrateurs réels. JIT va plus loin en garantissant que même lorsqu'une élévation est nécessaire, elle est bornée dans le temps : l'accès expire automatiquement après 1 heure, 4 heures ou toute durée configurée. Cette approche s'inscrit dans la continuité des Privileged Access Workstations (PAW), recommandées par Microsoft pour isoler physiquement et logiquement les postes d'administration.

La menace est réelle et documentée. Les groupes ransomware comme LockBit 3.0, ALPHV/BlackCat et leurs successeurs ciblent systématiquement les comptes à privilèges dans les

premières 48 heures suivant leur intrusion initiale. Limiter la durée et le périmètre des droits administrateurs réduit mécaniquement le rayon d'explosion d'une compromission.

Just Enough Administration (JEA) : Principes et Architecture

JEA est un framework PowerShell intégré à Windows Server 2025 qui permet de créer des endpoints d'administration restreints. Son fonctionnement repose sur deux composants fondamentaux :

Role Capability Files (.psrc) : Ces fichiers YAML-like définissent exactement ce qu'un rôle peut faire. Ils listent les cmdlets visibles (`VisibleCmdlets`), les fonctions (`VisibleFunctions`), les alias (`VisibleAliases`) et les providers PowerShell (`VisibleProviders`) accessibles dans la session JEA. Un fichier .psrc pour un opérateur IIS ne contiendra que les cmdlets liées à la gestion de IIS, interdisant toute autre commande.

Session Configuration Files (.pssc) : Ces fichiers définissent les paramètres de la session PowerShell restreinte — quel compte exécute réellement les commandes (RunAsVirtualAccount ou gMSA), quels groupes Active Directory ont accès à cet endpoint, et quelle langue PowerShell est autorisée (généralement ConstrainedLanguage pour maximiser la sécurité).

Le mécanisme clé de JEA est le **Virtual Account** ou le **Group Managed Service Account** : lorsqu'un technicien se connecte à un endpoint JEA, ses commandes s'exécutent sous l'identité d'un compte virtuel temporaire avec les droits nécessaires, jamais sous son propre compte. Le technicien n'a donc jamais de droits permanents sur le serveur — il emprunte temporairement une identité fonctionnelle contrôlée.

Cette architecture s'articule naturellement avec les mécanismes de Credential Guard sur Windows Server 2025, qui protège les credentials en mémoire contre les attaques de type LSASS dump, rendant la chaîne de sécurité cohérente de bout en bout.

Déploiement JEA sur Windows Server 2025

Voici un exemple complet de déploiement JEA pour un rôle d'opérateur IIS. Ce scénario illustre la création d'un endpoint permettant de gérer IIS sans droits administrateurs directs.

Étape 1 : Créer le Role Capability File

```
# Créer le répertoire JEA si nécessaire
$modulePath = "C:\Program Files\WindowsPowerShell\Modules\JEA_IIS"
New-Item -ItemType Directory -Path "$modulePath\RoleCapabilities" -Force

# Créer le Role Capability File pour l'opérateur IIS
New-PSRoleCapabilityFile -Path "$modulePath\RoleCapabilities\IISOperator.psrc" `
    -Description "Opérateur IIS - gestion des sites et pools d'applications" `
    -Author "Equipe Sécurité - AyiNedjimi Consultants" `
    -CompanyName "AyiNedjimi Consultants" `
    -VisibleCmdlets @(
        # Gestion IIS via WebAdministration
        'WebAdministration\Get-Website',
        'WebAdministration\Start-Website',
        'WebAdministration\Stop-Website',
        'WebAdministration\Restart-WebItem',
        'WebAdministration\Get-WebAppPoolState',
        'WebAdministration\Start-WebAppPool',
        'WebAdministration\Stop-WebAppPool',
        # Gestion des services Windows ciblés
        @{Name='Restart-Service'; Parameters=@{Name='Name'; ValidateSet='W3SVC','WAS','WMSVC'}},
        @{Name='Stop-Service'; Parameters=@{Name='Name'; ValidateSet='W3SVC','WAS'}},
        @{Name='Start-Service'; Parameters=@{Name='Name'; ValidateSet='W3SVC','WAS'}},
        # Lecture des logs d'événements
        @{Name='Get-EventLog'; Parameters=@{Name='LogName'; ValidateSet='Application','System'}},
        'Get-WinEvent',
        # Cmdlets de base autorisées
        'Get-Date', 'Get-Process', 'Get-Service', 'Measure-Object', 'Select-Object',
        'Where-Object', 'Format-Table', 'Format-List', 'Out-Default', 'Get-Help'
    ) `
    -VisibleProviders 'FileSystem' `
    -VisibleFunctions 'TabExpansion2' `
    -ModulesToImport 'WebAdministration'
```

Étape 2 : Créer le Session Configuration File

```
# Créer le Session Configuration File
New-PSSessionConfigurationFile -Path "C:\JEA\IISOperator.pssc" `
    -SessionType RestrictedRemoteServer `
    -Description "Endpoint JEA pour opérateurs IIS" `
    -Author "Equipe Sécurité" `
    -RunAsVirtualAccount `
    -RunAsVirtualAccountGroups 'IIS_JEA_RunAs' `
    -RoleDefinitions @{
        'CONTOSO\GRP_JEA_IIS_Operators' = @{RoleCapabilities = 'IISOperator'}
        'CONTOSO\GRP_JEA_IIS_Admins'   = @{RoleCapabilities = 'IISOperator', 'IISAdministrator'}
    } `
    -TranscriptDirectory 'C:\JEA\Transcripts' `
    -LanguageMode ConstrainedLanguage

# Vérifier la validité du fichier avant d'enregistrer
Test-PSSessionConfigurationFile -Path "C:\JEA\IISOperator.pssc"
```

Étape 3 : Enregistrer et activer l'endpoint JEA

```
# Enregistrer l'endpoint JEA (nécessite des droits administrateur)
Register-PSSessionConfiguration `
    -Name "JEA_IISOperator" `
    -Path "C:\JEA\IISOperator.pssc" `
    -Force `
    -NoServiceRestart

# Redémarrer le service WinRM pour activer l'endpoint
Restart-Service WinRM

# Vérifier que l'endpoint est bien enregistré
Get-PSSessionConfiguration -Name "JEA_IISOperator" | Format-List Name, Permission, Enabled

# Tester les capacités d'un utilisateur spécifique
Get-PSSessionCapability -ConfigurationName "JEA_IISOperator" -Username "CONTOSO\jdupont"

# Se connecter à l'endpoint JEA (côté technicien)
Enter-PSSession -ComputerName "SRV-WEB-01" -ConfigurationName "JEA_IISOperator"
# Ou pour exécuter une commande à distance
Invoke-Command -ComputerName "SRV-WEB-01" -ConfigurationName "JEA_IISOperator" `
    -ScriptBlock { Get-Website | Select-Object Name, State, PhysicalPath }
```

Déploiement à grande échelle via DSC ou GPO

```
# Déploiement JEA via Desired State Configuration (DSC)
Configuration JEADeployment {
    param([string[]]$ComputerName)
    Import-DscResource -ModuleName PSDesiredStateConfiguration

    Node $ComputerName {
        File JEAModuleDir {
            DestinationPath = "C:\Program Files\WindowsPowerShell\Modules\JEA_IIS"
            Type = "Directory"
            Ensure = "Present"
        }

        Script RegisterJEA {
            GetScript  = { @{Result = (Get-PSSessionConfiguration -Name 'JEA_IISOperator' -EA Sil
            TestScript = { (Get-PSSessionConfiguration -Name 'JEA_IISOperator' -EA SilentlyContin
            SetScript  = {
                Register-PSSessionConfiguration -Name "JEA_IISOperator" -Path "C:\JEA\IISOperator
                Restart-Service WinRM
            }
            DependsOn = "[File]JEAModuleDir"
        }
    }
}
```

Just-in-Time (JIT) avec Microsoft PAM et Privileged Identity Management

Le JIT administration repose sur un principe fondamental : aucun compte ne doit être membre permanent d'un groupe à privilèges. L'appartenance à ces groupes est octroyée temporairement, pour la durée strictement nécessaire à l'exécution d'une tâche, puis révoquée automatiquement.

Microsoft PAM (Privileged Access Management) — Solution on-premises

Microsoft PAM, disponible dans les forêts Active Directory fonctionnant au niveau Windows Server 2016 ou supérieur, introduit le concept de **shadow principals** et de **bastion forest**. Une

forêt bastion dédiée, isolée de la forêt de production, héberge des comptes PAM associés aux comptes de production via des shadow principals. Lorsqu'un administrateur demande un accès élevé, son compte de production est temporairement ajouté au groupe cible avec un `msDS-MemberTimeToLive` — un attribut Active Directory qui provoque la suppression automatique de la membership après expiration du TTL.

```
# Exemple : Activer une membership temporaire via Microsoft PAM
# (exécuté depuis la forêt bastion)

# Importer le module MIM PAM (Microsoft Identity Manager)
Import-Module MIMPAM

# Demander un accès JIT à un groupe à privilèges
$pamRequest = New-PAMRequest `
    -Justification "Maintenance urgente serveur SRV-SQL-01 - Ticket INC-20250612-001" `
    -Role (Get-PAMRole -DisplayName "SQL Server Administrators") `
    -RequestedTTL (New-TimeSpan -Hours 2)

# Surveiller le statut de la demande
Get-PAMRequest -Filter { State -eq "Active" } |
    Select-Object RequestorName, RoleName, ExpirationTime, Justification

# Lister les memberships JIT actifs sur un groupe cible
Get-ADGroup "SQL_Administrators" |
    Get-ADGroupMember |
    ForEach-Object {
        $ttl = (Get-ADUser $_ -Properties "msDS-MembershipTimeToLive")."msDS-MembershipTimeToLive
        [PSCustomObject]@{
            Member     = $_.SamAccountName
            ExpiresIn  = if ($ttl) { "$([Math]::Round($ttl/60)) minutes" } else { "Permanent" }
        }
    }
```

Entra Privileged Identity Management (PIM) — Solution hybride/cloud

Pour les environnements hybrides ou full-cloud, Entra PIM (anciennement Azure AD PIM) offre une interface unifiée pour le JIT. Les administrateurs ne sont plus membres permanents des rôles Entra ID (Global Administrator, Security Administrator, etc.) ni des groupes Entra ID associés à des ressources Azure. Ils sont **éligibles** à ces rôles et doivent activer leur accès explicitement, avec justification, pour une durée limitée.

```
# Gestion Entra PIM via Microsoft Graph PowerShell
Connect-MgGraph -Scopes "RoleManagement.ReadWrite.Directory", "PrivilegedAccess.ReadWrite.AzureAD

# Rendre un utilisateur éligible au rôle Security Administrator (JIT)
$userId = (Get-MgUser -Filter "userPrincipalName eq 'jdupont@contoso.fr'").Id
$roleId = (Get-MgRoleManagementDirectoryRoleDefinition -Filter "displayName eq 'Security Administ

# Créer une assignation éligible (pas permanente)
New-MgRoleManagementDirectoryRoleEligibilityScheduleRequest -BodyParameter @{
    action          = "adminAssign"
    justification   = "Assignation JIT pour l'équipe SOC"
    roleDefinitionId = $roleId
    directoryScopeId = "/"
    principalId     = $userId
    scheduleInfo    = @{
        startDateTime = (Get-Date)
        expiration    = @{ type = "afterDuration"; duration = "P180D" }
    }
}

# Activer le rôle (côté utilisateur, durée 4h max)
New-MgRoleManagementDirectoryRoleAssignmentScheduleRequest -BodyParameter @{
    action           = "selfActivate"
    justification    = "Audit de sécurité planifié - Ticket SEC-20250612-042"
    roleDefinitionId = $roleId
    directoryScopeId = "/"
    principalId      = $userId
    scheduleInfo     = @{
        startDateTime = (Get-Date)
        expiration    = @{ type = "afterDuration"; duration = "PT4H" }
    }
}
```

Entra PIM s'intègre également avec les Authentication Policy Silos de Windows Server 2025 pour garantir que les activations de rôles JIT ne peuvent provenir que de workstations de confiance, ajoutant une couche de contrôle contextuel.

Combinaison JEA + JIT : Architecture Zero Trust

La puissance réelle de ces mécanismes apparaît dans leur combinaison. Une architecture Zero Trust mature pour l'administration Microsoft repose sur quatre couches :

1. Isolation des postes d'administration : les administrateurs utilisent des PAW dédiées depuis lesquelles ils initient leurs sessions JEA. Ces postes sont isolés du réseau bureautique standard et soumis à des GPO restrictives.

2. Activation JIT préalable : avant toute session d'administration, l'opérateur doit activer son accès via Entra PIM ou Microsoft PAM, avec justification et pour une durée définie. Cette activation déclenche une notification aux responsables sécurité et crée une entrée d'audit dans le SIEM.

3. Connexion via endpoint JEA : une fois le JIT actif, l'opérateur se connecte à l'endpoint JEA approprié. Ses commandes s'exécutent dans un environnement PowerShell restreint, sous une identité virtuelle, sans jamais exposer ses credentials réels au serveur cible.

4. Expiration automatique : à la fin de la fenêtre JIT, l'appartenance au groupe est révoquée. Si l'opérateur a oublié de fermer sa session JEA, celle-ci perd ses droits effectifs au niveau système (le Virtual Account ne peut plus effectuer d'actions privilégiées).

Cette architecture s'appuie également sur les comptes membres du groupe Protected Users pour les identités d'administration, empêchant l'utilisation de NTLM, la délégation Kerberos et la mise en cache des credentials — des protections essentielles contre les attaques pass-the-hash et overpass-the-hash.

```
# Script d'administration Zero Trust : workflow JIT + JEA complet
function Invoke-SecureAdminTask {
    param(
        [string]$TargetServer,
        [string]$JEAEndpoint,
        [string]$JITRoleId,
        [string]$Justification,
        [int]$DurationHours = 1,
        [scriptblock]$AdminBlock
    )

    # Étape 1 : Activer l'accès JIT
    Write-Host "[JIT] Activation de l'accès élevé pour $DurationHours heure(s)..." -ForegroundCol
    $jitRequest = New-MgRoleManagementDirectoryRoleAssignmentScheduleRequest -BodyParameter @{
        action           = "selfActivate"
        justification    = $Justification
        roleDefinitionId = $JITRoleId
        directoryScopeId = "/"
        principalId      = (Get-MgContext).Account
        scheduleInfo     = @{
            startDateTime = (Get-Date)
            expiration    = @{ type = "afterDuration"; duration = "PT${DurationHours}H" }
        }
    }
    Start-Sleep -Seconds 5  # Laisser le temps à la propagation AD

    try {
        # Étape 2 : Exécuter la tâche via JEA
        Write-Host "[JEA] Connexion sécurisée à $TargetServer via $JEAEndpoint..." -ForegroundCol
        $result = Invoke-Command -ComputerName $TargetServer `
            -ConfigurationName $JEAEndpoint `
            -ScriptBlock $AdminBlock

        return $result
    }
    finally {
        # Étape 3 : Révoquer l'accès JIT immédiatement après la tâche
        Write-Host "[JIT] Révocation de l'accès élevé..." -ForegroundColor Green
        # La révocation anticipée est une bonne pratique Zero Trust
        Remove-MgRoleManagementDirectoryRoleAssignmentScheduleRequest -UnifiedRoleAssignmentSched
    }
}
```

Audit et Monitoring des sessions JEA/JIT

La traçabilité est un impératif légal sous NIS 2 et une nécessité opérationnelle pour les équipes SOC. JEA et JIT génèrent naturellement une piste d'audit riche, à condition de configurer correctement les mécanismes de logging PowerShell.

Configuration du logging PowerShell via GPO

```
# Activer le logging PowerShell via le registre (équivalent GPO)
# Applicable sur Windows Server 2025 avec PowerShell 5.1 et 7+

$psPolicies = "HKLM:\SOFTWARE\Policies\Microsoft\Windows\PowerShell"

# 1. Module Logging (enregistre les noms de modules et fonctions appelés)
Set-ItemProperty "$psPolicies\ModuleLogging" -Name "EnableModuleLogging" -Value 1 -Type DWord
Set-ItemProperty "$psPolicies\ModuleLogging\ModuleNames" -Name "*" -Value "*" -Type String

# 2. Script Block Logging (capture le code PowerShell exécuté, y compris déobfusqué)
Set-ItemProperty "$psPolicies\ScriptBlockLogging" -Name "EnableScriptBlockLogging" -Value 1 -Type
Set-ItemProperty "$psPolicies\ScriptBlockLogging" -Name "EnableScriptBlockInvocationLogging" -Val

# 3. Transcription (fichier texte complet de chaque session)
Set-ItemProperty "$psPolicies\Transcription" -Name "EnableTranscripting" -Value 1 -Type DWord
Set-ItemProperty "$psPolicies\Transcription" -Name "OutputDirectory" -Value "\SRV-LOG-01\JEATrans
Set-ItemProperty "$psPolicies\Transcription" -Name "EnableInvocationHeader" -Value 1 -Type DWord

Write-Host "Logging PowerShell activé. Events: 4103 (module), 4104 (scriptblock), 800 (session)"
```

Event IDs clés pour la surveillance JEA/JIT

```
# Requêtes de surveillance des événements JEA/JIT critiques

# Event 4672 : Ouverture de session avec privilèges spéciaux (détecte les élévations JIT)
Get-WinEvent -FilterHashtable @{
    LogName = 'Security'
    Id      = 4672
    StartTime = (Get-Date).AddHours(-24)
} | Where-Object { $_.Message -match 'SeDebugPrivilege|SeTakeOwnershipPrivilege' } |
    Select-Object TimeCreated, @{N='User';E={$_.Properties[1].Value}}, Message

# Event 4624 : Ouverture de session réseau (Type 3) vers endpoint JEA
Get-WinEvent -FilterHashtable @{
    LogName = 'Security'
    Id      = 4624
    StartTime = (Get-Date).AddHours(-24)
} | Where-Object { $_.Properties[8].Value -eq 3 } |  # LogonType 3 = Network
    Select-Object TimeCreated,
        @{N='User';E={$_.Properties[5].Value}},
        @{N='WorkstationName';E={$_.Properties[11].Value}},
        @{N='ProcessName';E={$_.Properties[17].Value}}

# Event 4104 : Script Block Logging (capture les commandes JEA exécutées)
Get-WinEvent -FilterHashtable @{
    LogName      = 'Microsoft-Windows-PowerShell/Operational'
    Id           = 4104
    StartTime    = (Get-Date).AddHours(-8)
} | Select-Object TimeCreated, @{N='ScriptBlock';E={$_.Properties[2].Value}} |
    Where-Object { $_.ScriptBlock -notlike '*TabExpansion*' }  # Filtrer le bruit

# Rapport JIT : memberships temporaires actifs dans AD
Get-ADGroup -Filter { Name -like "*Admin*" -or Name -like "*Privileged*" } |
    ForEach-Object {
        $group = $_
        Get-ADGroupMember $group | ForEach-Object {
            $user = Get-ADUser $_ -Properties "msDS-MembershipTimeToLive", "Description"
            [PSCustomObject]@{
                Group      = $group.Name
                Member     = $user.SamAccountName
                TTL        = $user."msDS-MembershipTimeToLive"
                Expires    = if ($user."msDS-MembershipTimeToLive") {
                                 (Get-Date).AddSeconds($user."msDS-MembershipTimeToLive")
                             } else { "Permanent - ALERT" }
```

```
            }
        }
    } | Sort-Object Group, Expires
```

Pour une intégration dans un SIEM, ces événements doivent être transférés via Windows Event Forwarding (WEF) vers un collecteur centralisé. Microsoft Sentinel dispose de règles analytiques préconstruites pour détecter les anomalies dans les sessions JEA (commandes hors périmètre tentées, connexions depuis des postes non-PAW) et les élévations JIT suspectes (fréquence anormale, horaires inhabituels). Les équipes SOC des entreprises françaises utilisant cet écosystème bénéficient d'une couverture de détection significativement améliorée sans développement de règles personnalisées.

Il est également recommandé de coupler cet audit avec le monitoring des RODC (Read-Only Domain Controllers), notamment dans les sites distants où JEA peut être déployé pour réduire le besoin d'accès directs aux contrôleurs de domaine principaux.

JEA/JIT et conformité NIS 2 en France

La directive NIS 2 (Network and Information Security Directive 2), transposée en France par la loi n° 2024-1077 du 7 novembre 2024, impose aux entités essentielles et importantes des mesures de cybersécurité proportionnées aux risques. L'article 21 de la directive, relatif aux mesures de gestion des risques de cybersécurité, couvre explicitement la gestion des accès à privilèges.

Mapping NIS 2 → JEA/JIT

L'article 21.2.i impose des *politiques et procédures relatives à l'utilisation de la cryptographie et, le cas échéant, du chiffrement*, ainsi que des *contrôles d'accès*. JEA répond à l'exigence de contrôle d'accès en implémentant le principe du moindre privilège au niveau PowerShell. JIT répond à l'exigence de limitation de la durée des accès élevés, réduisant la fenêtre d'exposition en cas de compromission de compte.

Pour les entités du secteur de la santé (hôpitaux, groupements hospitaliers de territoire), de l'énergie (distributeurs électricité/gaz) et de l'administration publique soumises à NIS 2 en Île-de-

France et en France, le déploiement de JEA/JIT permet de documenter concrètement les contrôles techniques mis en place pour satisfaire aux exigences de l'ANSSI. Les plans de contrôle NIS 2 soumis à l'ANSSI doivent inclure une description des mécanismes de PAM (Privileged Access Management), et JEA/JIT en constituent l'implémentation technique native Microsoft.

Documentation recommandée pour l'audit NIS 2

Inventaire des endpoints JEA déployés (ConfigurationName, serveurs cibles, groupes autorisés)

Matrice des Role Capability Files et cmdlets autorisées par rôle métier

Politique JIT : durées maximales d'élévation par type d'opération, workflow de validation

Preuves de logging : activation ScriptBlock/Transcription, centralisation vers SIEM

Rapport mensuel des élévations JIT : qui, quand, durée, justification, ressource ciblée

Les RSSI français peuvent s'appuyer sur le référentiel PGSSI-S de l'ANSSI et sur le guide Microsoft Security Compliance Toolkit pour aligner leur implémentation JEA/JIT avec les exigences réglementaires. La combinaison JEA/JIT/PAW constitue aujourd'hui la réponse technique la plus complète aux exigences de contrôle des accès à privilèges dans les environnements Microsoft.

Questions fréquentes sur JEA et JIT

Quelle est la différence entre JEA et JIT dans Windows Server 2025 ?

JEA (Just Enough Administration) limite ce qu'un administrateur peut faire en restreignant les commandes PowerShell disponibles via des Role Capability Files. JIT (Just-in-Time) limite la durée pendant laquelle un compte dispose de privilèges élevés. Ces deux approches sont complémentaires : JEA contrôle le périmètre fonctionnel, JIT contrôle la fenêtre temporelle. Ensemble, ils forment le socle d'une administration Zero Trust en réduisant drastiquement la surface d'attaque liée aux comptes à privilèges permanents.

JEA nécessite-t-il des licences spécifiques dans un environnement Microsoft ?

JEA est intégré nativement dans PowerShell 5.1 et PowerShell 7+ sur Windows Server 2025, sans coût de licence supplémentaire. En revanche, Entra Privileged Identity Management (PIM) pour le JIT cloud requiert une licence Microsoft Entra ID P2 (anciennement Azure AD Premium P2). La fonctionnalité Microsoft PAM pour le JIT on-premises est incluse dans Windows Server et ne nécessite que les droits Active Directory appropriés. Les entreprises françaises utilisant déjà Microsoft 365 E3/E5 disposent souvent déjà des licences nécessaires pour Entra PIM.

Comment auditer efficacement les sessions JEA sur Windows Server 2025 ?

L'audit des sessions JEA repose sur trois mécanismes PowerShell : la transcription (journalisation de chaque commande dans des fichiers texte), le module logging (enregistrement de tous les modules appelés) et le ScriptBlock logging (capture du code PowerShell exécuté). Ces logs génèrent les événements Windows 4103, 4104 et 800 dans le journal PowerShell/Operational. Pour une visibilité complète, associez ces mécanismes à un SIEM (Microsoft Sentinel, Splunk) et surveillez également les événements 4624, 4625 et 4672 du Security log pour détecter les tentatives d'élévation non autorisées.

Peut-on utiliser JEA avec des comptes de service et des pipelines CI/CD ?

Oui, JEA est parfaitement adapté aux comptes de service et aux pipelines CI/CD. Vous créez un endpoint JEA dédié avec un RunAsVirtualAccount ou un Group Managed Service Account (gMSA) pour l'exécution, et vous assignez uniquement les cmdlets nécessaires au pipeline (déploiement, redémarrage de services spécifiques, etc.). Le compte de service du pipeline se connecte via Enter-PSSession ou Invoke-Command avec ConfigurationName, sans jamais avoir de droits administrateur directs. Cette approche réduit considérablement le risque lié à la compromission d'un token CI/CD dans vos chaînes DevSecOps.

Comment intégrer JEA et JIT avec la politique NIS 2 pour une entreprise française ?

La directive NIS 2, transposée en droit français via la loi du 7 novembre 2024, impose aux entités essentielles et importantes des mesures de gestion des accès à privilèges. JEA répond à

l'exigence de contrôle des accès en appliquant le principe du moindre privilège. JIT répond à l'exigence de supervision des accès à privilèges en limitant leur durée à la stricte nécessité. L'audit combiné JEA/JIT génère la traçabilité requise par NIS 2. Les RSSI français des secteurs santé, énergie et administration publique doivent documenter ces contrôles dans leur politique PAM soumise à l'ANSSI.

Pour les environnements hybrides AD/Entra ID, Microsoft Entra Connect v3 est le composant critique qui synchronise les identités — son durcissement est aussi prioritaire que le déploiement de JEA/JIT.

À RETENIR

Points clés à retenir

JEA élimine les droits permanents : les techniciens n'ont jamais de droits administrateurs directs — ils accèdent à un environnement PowerShell restreint qui exécute les commandes sous un Virtual Account contrôlé

JIT borne la fenêtre d'exposition : même les vraies élévations administrateur sont limitées dans le temps (1h, 4h, etc.) via Entra PIM ou Microsoft PAM, avec expiration automatique

L'architecture Zero Trust complète combine PAW + JEA + JIT + Protected Users + Credential Guard pour une défense en profondeur cohérente contre les mouvements latéraux

Le logging est obligatoire : activez ScriptBlock Logging, Module Logging et Transcription PowerShell dès le déploiement — ces logs sont indispensables pour la détection forensique et la conformité NIS 2

Commencez par un pilote : déployez JEA sur un rôle fonctionnel limité (IIS, DNS, monitoring) avant de généraliser — testez avec Get-PSSessionCapability pour valider les permissions avant la mise en production

Sources

ANSSI CERT-FR — Publications et alertes

MITRE ATT&CK — Framework des techniques d'attaque

CISA — Advisories de cybersécurité