# Configuration du Windows supervisé pour le pack windows-by-WinRM__shinken

**Sommaire**


- Contexte
- Versions Windows supportées
- Procédure de configuration
- Configuration de WinRM
  - Configuration par script
    - Le script de configuration d'un serveur windows à superviser
    - Comment déployer le script de configuration
    - Exécuter le script de configuration
      - Pré-requis avant d'éxecuter le script
      - Configuration par défaut
      - Configuration personnalisée
      - Configuration des services à superviser
      - Activer l'exécution temporaire des scripts
    - Tester le pare-feu et WinRM
      - Vérifier le pare-feu Windows
      - Vérifier le service WinRM
  - Configuration étape par étape
    - Configuration minimale
    - Authentification
      - ntlm
      - basic
    - Configuration de l'utilisateur
      - Création de l'utilisateur
      - Installation de la langue
      - Configuration de la langue
      - Configuration des groupes
    - Configuration des permissions
      - Permissions WinRM pour l'utilisateur
      - Autorisation aux objets CIM
      - Autorisation au service W32Time
      - Autorisation au journal de sécurité ( Security Event Log )
      - Appliquer les permissions
    - Tester le pare-feu et WinRM
      - Vérifier le pare-feu Windows :
      - Vérifier le service WinRM :


## Contexte

Cette page a pour but de décrire la mise en place d'une configuration minimale nécessaire pour un Windows supervisé par le pack **windows-by-WinRM__shinken.**

Afin de configurer des **postes Windows d'un domaine** ( *Active Directory* ), voir la page Configuration du Windows supervisé dans un Domaine ( Active Directory ) pour le pack windows-by-WinRM__shinken

## Versions Windows supportées

Voici la liste des versions Windows que la sonde peut superviser à distance :

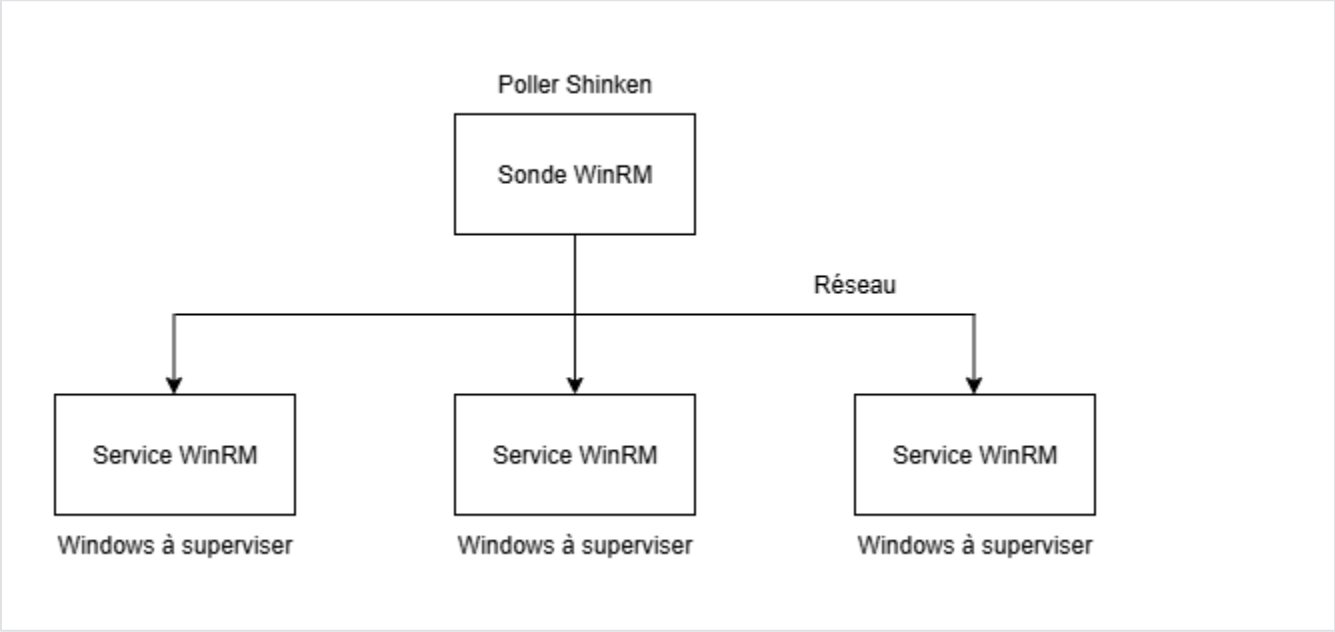
- Windows Server 2025
- Windows Server 2022
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012 R2
- Windows 11
- Windows 10

## Procédure de configuration

La supervision d'un Windows par un **poller Shinken** se fait par requête WinRM.

- Sur le **poller Shinken**, la sonde WinRM installée est responsable d'envoyer les requêtes et de traiter les réponses.

- Sur les Windows à superviser, le **service Windows WinRM** est responsable de répondre aux requêtes.




Le service Windows **WinRM** sur un Windows supervisé permet l'exécution de commandes à distance, et notamment la récupération d'informations sur la machine mises à disposition par **WMI** ( *Windows Management Instrumentation* ).

Il est donc nécessaire de configurer :

- le service **WinRM** sur chaque Windows à superviser,
- un utilisateur qui exécutera des commandes via **WinRM** et accèdera aux informations nécessaires aux checks.

⚠ Les configurations suivantes sont destinées à des serveurs **Windows configurés en "groupe de travail"** ( *Workgroup* ).

La configuration pour un serveur **Windows configuré par "domaine"** ( *Active Directory* ) diffère partiellement ( *Voir la page Configuration du Windows supervisé dans un Domaine ( Active Directory ) pour le pack windows-by-WinRM__shinken* ).

## Configuration de WinRM

Le service **WinRM** est installé par défaut sur les systèmes d'exploitations compatibles avec le pack. Voici comment le configurer :

Toutes les commandes ci-dessous doivent être exécutées dans un **terminal PowerShell** lancé en mode **administrateur** ( *sauf mention contraire* )

Voici la commande à exécuter pour vérifier si le service est bien démarré :

```
Get-Service WinRM
```

Exemple :

```
C:\Users\Administrateur> Get-Service WinRM

Status   Name               DisplayName
------   ----               -----------
Running  WinRM              Gestion à distance de Windows (Gest...
```

Sur Windows 11 et Windows 10, il est nécessaire de configurer le démarrage automatique de WinRM :

```
Set-Service -Name WinRM -StartupType Automatic
```

## Configuration par script

Voici un script permettant de configurer rapidement par ligne de commandes un poste Windows.

Il doit être exécuté sur chaque poste Windows à superviser.

Son usage **n'est PAS obligatoire**, il reprend l'entièreté des commandes décrites dans la procédure de configuration étape par étape détaillée plus bas dans la documentation.

⚠ Avant d'utiliser ce script, lire attentivement les points recommandés suivants :

- **Lecture préalable recommandée**
  - Le script modifie la configuration système ( WinRM, WMI, services, permissions ).
- **Environnement de test recommandé**
  - Exécutez d'abord le script dans un environnement de test ou sur un serveur non critique.
  - Ne déployez en production qu'après validation de son bon fonctionnement.
- **Privilèges administratifs requis**
  - Le script doit être lancé **en tant qu'administrateur**.
  - Un manque de droits entraînera des erreurs ou une configuration partielle.
- **Impact sécurité**
  - L'activation de WinRM et l'usage éventuel de l'authentification Basic/AllowUnencrypted peuvent exposer des risques de sécurité.
  - Vérifiez vos politiques de sécurité internes avant de déployer ce type de configuration.
- **Adaptabilité du script**
  - Certaines parties sont spécifiques à **Windows 10, Windows 11, Windows Server 2025**.
  - Pour d'autres versions de Windows, l'installation des langues devra être fait manuellement.
- **Responsabilité**
  - L'exécution reste **sous la responsabilité de l'administrateur**.

Le script va :

- **Détecter** les langues disponibles.
- **Créer un utilisateur**, et l'ajouter dans les groupes nécessaires.
- **Configurer la langue** du nouvel utilisateur de supervision.
- Configurer WinRM pour l'**authentification Basic ou Negotiate**, puis HTTP.
- Configurer l'utilisateur de supervision pour exécuter des **commandes WinRM.**
- Configurer l'utilisateur de supervision pour récupérer **les informations WMI/CIM root\cimv2**.
- Configurer l'utilisateur de supervision pour récupérer **les informations WMI/CIM root\standardcimv2**.
- Configurer l'utilisateur de supervision pour récupérer **les informations du journal de sécurité (Security Event Log) de Windows**.
- Configurer l'utilisateur de supervision pour récupérer **les informations de W32Time, le service NTP de Windows**.
- Configurer l'utilisateur de supervision pour récupérer **le status d'une liste configurable de services**.

Rappel : le script nécessite les droits administrateurs pour l'exécution

### Le script de configuration d'un serveur windows à superviser

**Télécharger le script ICI**

Configure-Host.ps1

### Comment déployer le script de configuration

Ensuite, il est nécessaire de **déployer le script** sur chaque machine Windows à superviser.

Les méthodes de déploiement du script vont dépendre de votre environnement et de vos outils ( *SSH, FTP, cloud, docker, Ansible, Terraform ...* ) .

Avant de déployer le script de configuration sur plusieurs serveurs Windows à superviser, il est fortement conseillé de tester le script de configuration sur UN serveur Windows, pour valider son comportement dans votre environnement.

Nous vous recommandons d'**utiliser vos propres solutions de déploiement** afin de correspondre à vos enjeux de sécurités. Néanmoins, voici un exemple de déploiement via SSH.

**Déploiement du script de configuration Windows via SSH :**

Si le poste Windows à superviser dispose déjà d'**un serveur SSH**, il est possible d'y transférer directement le script de configuration.

Le script fourni se trouve dans le pack, sous le répertoire "**supervised-host**". Il est identique à celui présenté précédemment dans la documentation.

Exemple :

```
scp supervised-host/Configure-Host.ps1 Administrateur@<IP_HOST>:C:/Users/Administrateur/
```

### Exécuter le script de configuration

#### Pré-requis avant d'éxecuter le script

Il est nécessaire pour les hôtes supervisés d'avoir d'installé au moins une des langues suivantes, pour le bon fonctionnement de la sonde :

- Français ( **fr-FR** )
- Anglais ( **en-US** )

La commande suivante permet d'afficher les langues installées sur votre machine :

```
Get-InstalledLanguage
```

Un résultat similaire à celui-la sera obtenu :

```
Language Language Packs  Language Features
-------- --------------  -----------------
en-US    LpCab, LXP      BasicTyping, Handwriting, Speech, TextToSpeech, OCR
fr-FR    LpCab, LXP      BasicTyping, Handwriting, Speech, TextToSpeech, OCR
```

Alors, si vos Windows à superviser **ont au moins une des langues nécessaires, passer à l'étape suivante**.

Sinon, voici quelques outils pour installer de nouvelles langues :

Ici, la langue configurée n'a **pas d'impact sur la traduction et l'affichage** des résultats générés par la sonde, mais a un **impact sur le bon fonctionnement**.

Pour les Windows ci-dessous, il est possible d'exécuter la commande suivante afin d'installer une langue :

- Windows 11
- Windows 10
- Windows Server 2025

```
# Installer la langue en Anglais
Install-Language -Language en-US -ExcludeFeatures

# Installer la langue en Francais
Install-Language -Language fr-FR -ExcludeFeatures
```

Pour les autres versions de Windows Server, d'autres solutions existent :

- Windows Server 2022
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012 R2

Une façon de déployer un nouveau langage sur ses machines supervisées est d'utiliser Windows *Feature On Demand* ( FoD ) afin d'installer un pack de langue.
Il est possible de télécharger et d'installer le pack de langue avec la commande suivante ( *non disponible sur Windows Server 2012 R2* ):

```
DISM.exe /Online /add-capability /CapabilityName:Language.Basic~~~en-US~0.0.1.0
```

Pour les installations hors-ligne, il est possible de demander à son support Microsoft le pack de langue livré dans un fichier *.iso* ou *.cab.* Il faudra ensuite le déployer et l'installer avec l'outil *DISM.EXE*. Plus d'informations ici : https://learn.microsoft.com/en-us/windows-hardware/manufacture/desktop/features-on-demand-v2--capabilities?view=windows-11.

Sinon, il est possible d'installer les langues par interface graphique.

Il est possible d'afficher l'aide du script avec la commande suivante :

```
.\Configure-Host.ps1 -Help
```

### Configuration par défaut

La commande ci-dessous permet de configurer votre poste Windows avec les paramètres par défaut livrés avec les modèles hôtes.

```
.\Configure-Host.ps1 -Default
```

> La configuration par défaut est une bonne façon de tester le pack, mais il est recommandé de changer les paramètres d'authentification pour des environnements exposés.

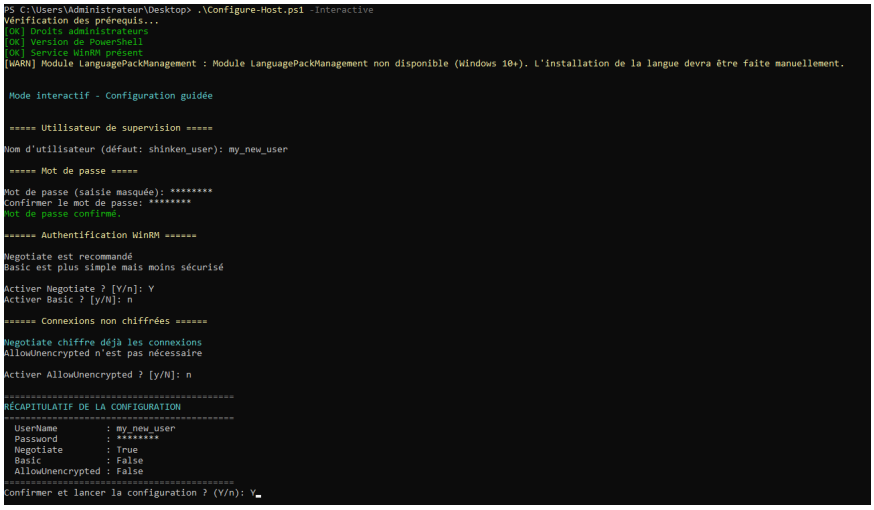
Ensuite, **redémarrez le poste Windows à superviser** afin d'appliquer les permissions, et passer à l'étape "Tester le pare-feu et WinRM"

### Configuration personnalisée

La commande ci-dessous permet de configurer votre poste Windows pas à pas.

```
.\Configure-Host.ps1 -Interactive
```

Il faudra ensuite entrer les paramètres demandés. Voici un example de configuration pas à pas :




Il est également possible de passer dans la ligne de commande les paramètres configurables :

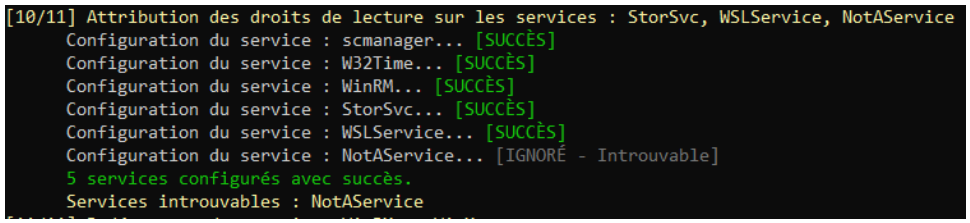
```
.\Configure-Host.ps1 -UserName MyNewUser -Password MyNewPassword -Negotiate true -Basic false -
AllowUnencrypted false
```

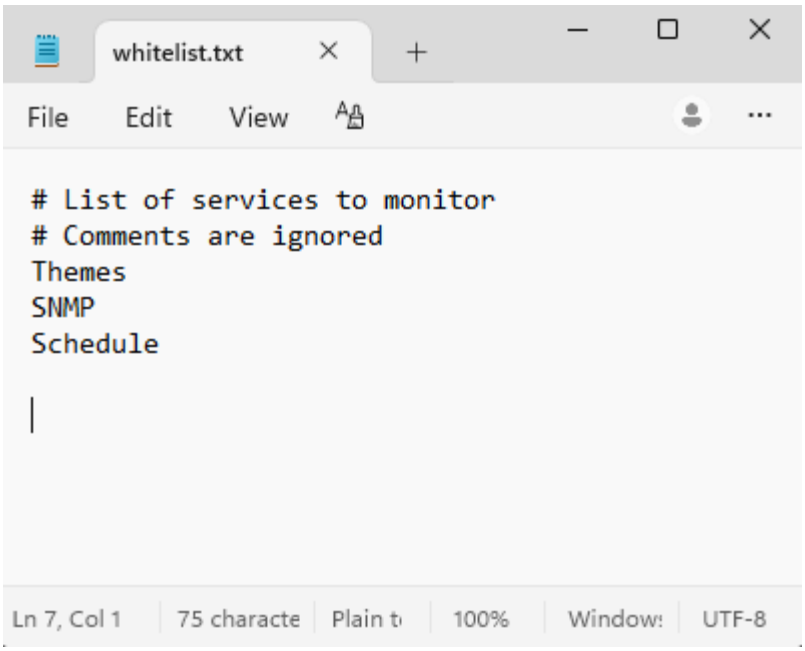
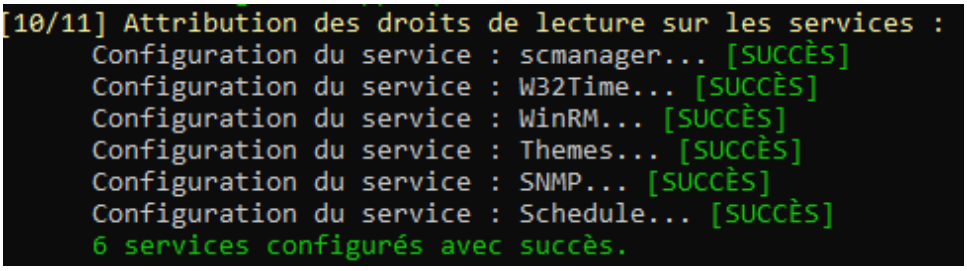
Ensuite, **redémarrez votre ordinateur** afin d'appliquer les permissions, et passer à l'étape "Tester le pare-feu et WinRM"

### Configuration des services à superviser

Superviser un service via le check Service $KEY$ State by WinRM necessite que l'utilisateur de supervision ai un accès en lecture au status de chaque Service à superviser.
Alors, il est **NECESSAIRE** de configurer les permissions de l'utilisateur de supervision pour acceder en lecture à ces status.

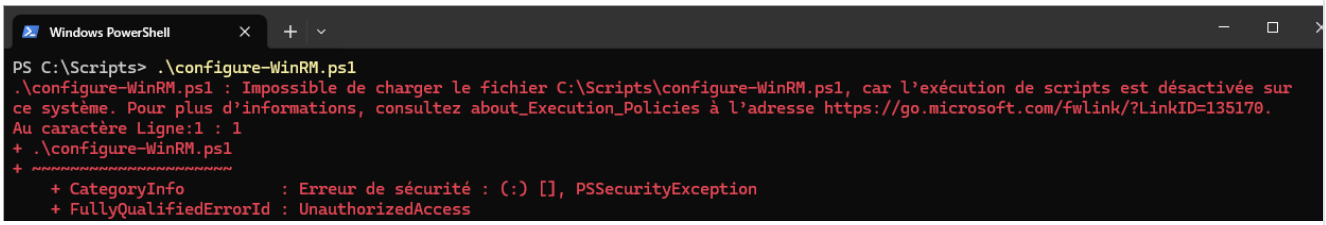
Voici les paramètres qui permettent la configuration des services :

| Paramètre | Description | Example |
|---|---|---|
| **-ServicesName** <String[]> | Liste des noms de services (tableaux de string) à autoriser en lecture.<br>• Associable avec **ServiceWhiteListPath** | `.\Configure-Host.ps1 -ServicesName StorSvc,WSLService,NotAService`<br>  |

| | | |
|---|---|---|
| **-ServiceWhiteListPath** <String> | Chemin vers un fichier texte contenant une liste de services à autoriser (un par ligne).<br><br>▪ Associable avec **ServicesName** |  <br><br><br>`.\Configure-Host.ps1 -ServiceWhiteListPath ./whitelist.txt`<br><br>  <br> |
| **-GrantAll** | Accorde les droits de lecture à TOUS les services présents sur la machine. | Liste l'ensemble des services sur la machine, et attribue les droits de lectures à l'utilisateur de supervision paramétré.<br>Ce paramètre n'est **pas recommandé** pour un environnement de production. |
| **-RevokeUnlisted** | Révoque les droits de lecture pour les services qui ne sont PAS dans la liste (ServicesName ou WhiteList).<br><br>Utile pour nettoyer une configuration précédente. | |
| **-SkipServicePermissions** | Ignore la configuration des services. | Les paramètres **ServicesName, ServiceWhiteListPath, GrantAll, RevokeUnlisted** seront ignorés |

## Activer l'exécution temporaire des scripts

Il est possible que vous obteniez l'erreur suivante.  Par défaut, PowerShell peut empêcher l'exécution de scripts pour des raisons de sécurité. Cette restriction est contrôlée par votre stratégie d'exécution ( *Execution Policy* ) .




La solution suivante est de **modifier temporairement** la stratégie d'exécution :

```
Set-ExecutionPolicy Unrestricted -Scope Process
```

- ⚠
  - **-Scope Process** : Ici, on agit uniquement sur **la session PowerShell en cours**, donc aucun impact persistant sur la machine.
  - **Unrestricted** : Permet l'exécution de scripts non-signés.

Il est maintenant possible de **relancer le script** dans le même terminal PowerShell.

### Tester le pare-feu et WinRM

> Les tests suivants sont à effectuer sur le poste Windows à superviser.

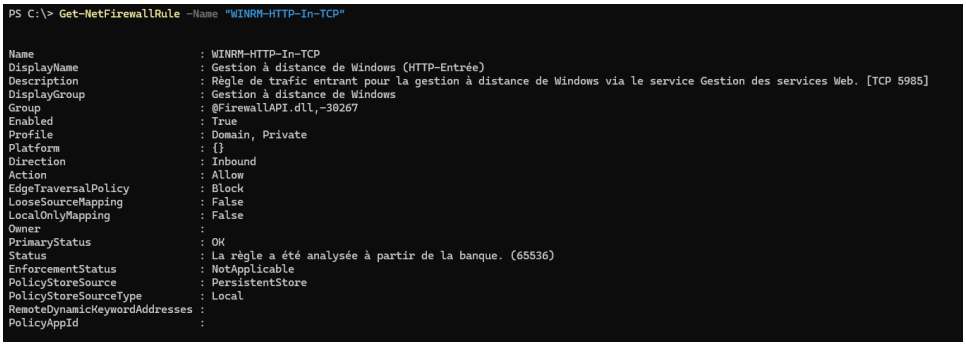
#### Vérifier le pare-feu Windows

L'objectif de ce test est de vérifier si le pare-feu possède une règle qui autorise la connexion WinRM.

- Exécuter la commande (en **administrateur**) :

  ```
  Get-NetFirewallRule -Name "WINRM-HTTP-In-TCP"
  ```

  - Résultat attendu :




    - Il faut s'assurer que :
      - ***"Enable"*** vaut ***"True"***.
      - ***"Action"*** vaut "***Allow***".

- Si ce n'est pas le cas, activer la règle ( **toujours avec les droits administrateurs** ):

```
Enable-NetFirewallRule -Name "WINRM-HTTP-In-TCP"
```

### Vérifier le service WinRM

L'objectif de ce test est de réaliser en local une connexion WinRM, avec les outils fournis par Windows.

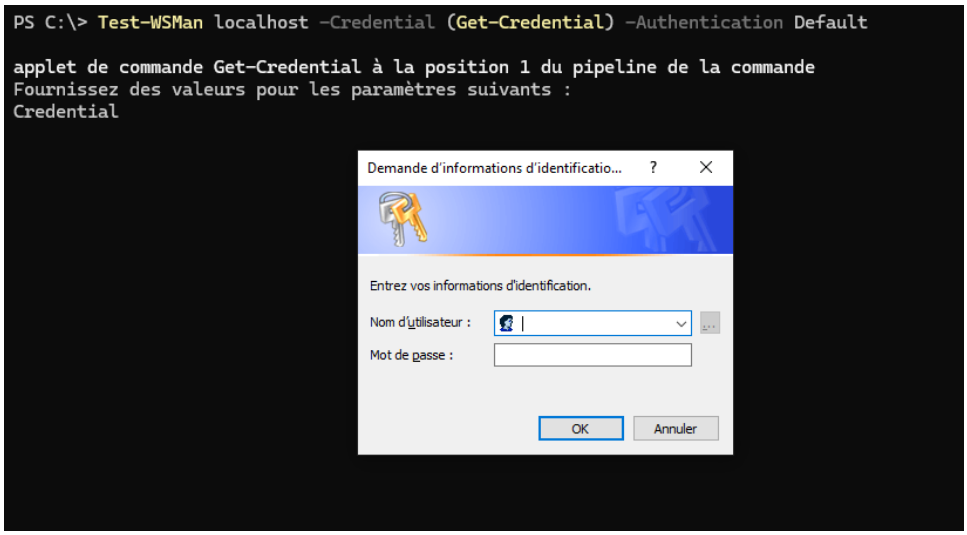
- Avec la commande **"Test-WSMan"** :

  - Si vous avez configuré **Negotiate** :

```
Test-WSMan localhost -Credential (Get-Credential) -Authentication Negotiate
```

  - Si vous avez configuré **Basic** :

```
Test-WSMan localhost -Credential (Get-Credential) -Authentication Basic
```

  - Rentrer le nom d'utilisateur ainsi que le mot de passe de l'utilisateur de supervision :
  - 


  - Résultat attendu :

```
PS C:\> Test-WSMan localhost -Credential (Get-Credential) -Authentication Default

applet de commande Get-Credential à la position 1 du pipeline de la commande
Fournissez des valeurs pour les paramètres suivants :
Credential


wsmid           : http://schemas.dmtf.org/wbem/wsman/identity/1/wsmanidentity.xsd
ProtocolVersion : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
ProductVendor   : Microsoft Corporation
ProductVersion  : OS: 10.0.19045 SP: 0.0 Stack: 3.0
```

Cela confirme que le service **WinRM** et l'**utilisateur de supervision** sont actifs et opérationnels sur la machine.

Une fois ces deux tests validés, la configuration du poste Windows **est terminée** et il est prêt à être supervisé.

L'étape suivante est de choisir, d'accrocher et de paramétrer les modèles d'hôtes fournis dans le pack ( *Voir la page Modèles d'hôtes du pack windows-by-WinRM__shinken* ).

## Configuration étape par étape

### Configuration minimale

WinRM dispose d'une commande de configuration intégrée qui permet entre autres de :

- Démarrer le service WinRM
- Mettre en place l'écouteur (HTTP) pour recevoir les requêtes des sondes Shinken
- Configurer le Firewall Windows pour autoriser les accès vers l'écouteur

Pour effectuer les actions de configuration, il suffit d'exécuter la commande suivante :

```
winrm quickconfig
```

Exemple :

```
C:\Users\Administrateur> winrm quickconfig

WinRM n'est pas configuré pour la gestion à distance de cet ordinateur.
Les modifications suivantes doivent être effectuées :

Créez un écouteur WinRM sur HTTP://* pour accepter les demandes de la gestion des services Web sur toutes
les adresses IP de cet ordinateur.

Activez l'exception de pare-feu WinRM.
Configurez LocalAccountTokenFilterPolicy pour attribuer des droits d'administration à distance à des
utilisateurs locaux.

Effectuer ces modifications [y/n] ? y

WinRM a été mis à jour pour la gestion à distance.

Écouteur WinRM créé sur HTTP://* pour accepter les demandes de la gestion des services Web sur toutes les
adresses IP de cet ordinateur.

Exception de pare-feu WinRM activée.
LocalAccountTokenFilterPolicy configuré pour attribuer des droits d'administration à distance à des
utilisateurs locaux.
```

Attention, Sur **Windows 11** et **Windows 10** l'hôte a besoin d'être en réseau privé et non public qui est par défaut. Sinon la configuration sera bloquée et non appliquée :

```
C:\Users\Administrateur> winrm quickconfig

Le service WinRM est déjà en cours d'exécution sur cet ordinateur.
WSManFault
    Message
        ProviderFault
            WSManFault
                Message = L'exception de pare-feu WinRM ne fonctionnera pas car l'un des types de
connexion réseau de cet ordinateur est défini à Public. Changez le type de connexion réseau en Domaine
ou Privé, puis recommencez.

Numéro d'erreur :  -2144108183 0x80338169
L'exception de pare-feu WinRM ne fonctionnera pas car l'un des types de connexion réseau de cet
ordinateur est défini à Public. Changez le type de connexion réseau en Domaine ou Privé, puis
recommencez.
```

Sur **Windows 11** et **Windows 10**, il est necessaire de configurer le démarrage automatique de WinRM :

```
Set-Service -Name WinRM -StartupType Automatic
```

### Authentification

Selon le choix du mode d'authentification, il faut configurer l'authentification à WinRM.

Plus de détails sur l'authentification ici ( Modèles d'hôtes du pack windows-by-WinRM__shinken ).

Par défaut, le service WinRM est configuré pour autoriser l'authentification "**Negotiate**" et désactive par défaut "**Basic**".

"**Negotiate**" est le protocole de négociation nécessaire pour communiquer via **ntlm.**

#### ntlm

L'authentification **ntlm** peut être activé en configurant "**Negotiate**" avec la commande suivante :

```
winrm set winrm/config/service/auth '@{Negotiate="true"}'
```

#### basic

L'utilisation du protocole « Basic » n'est pas recommandée lorsque des modes d'authentification plus sécurisés sont disponibles sur le système.

```
winrm set winrm/config/service/auth '@{Basic="true"}'
```

L'authentification **basic** n'ajoute pas d'encryption, alors il est necessaire de configurer le Windows pour accepter les communications non-chiffrées :

```
winrm set winrm/config/service '@{AllowUnencrypted="true"}'
```

### Configuration de l'utilisateur

La configuration nécessite la création d'un utilisateur spécifiquement utilisé pour la supervision via ce pack. Cet utilisateur bénéficiera de permissions nécessaires à la supervision et permettra à la sonde du pack de se connecter à distance afin d'exécuter des commandes.

L'utilisation du compte administrateur du poste Windows permettrait d'obtenir toutes les informations du système, car celui-ci possède par défaut tous les droits d'accès ( *WMI, WinRM, etc.* ).

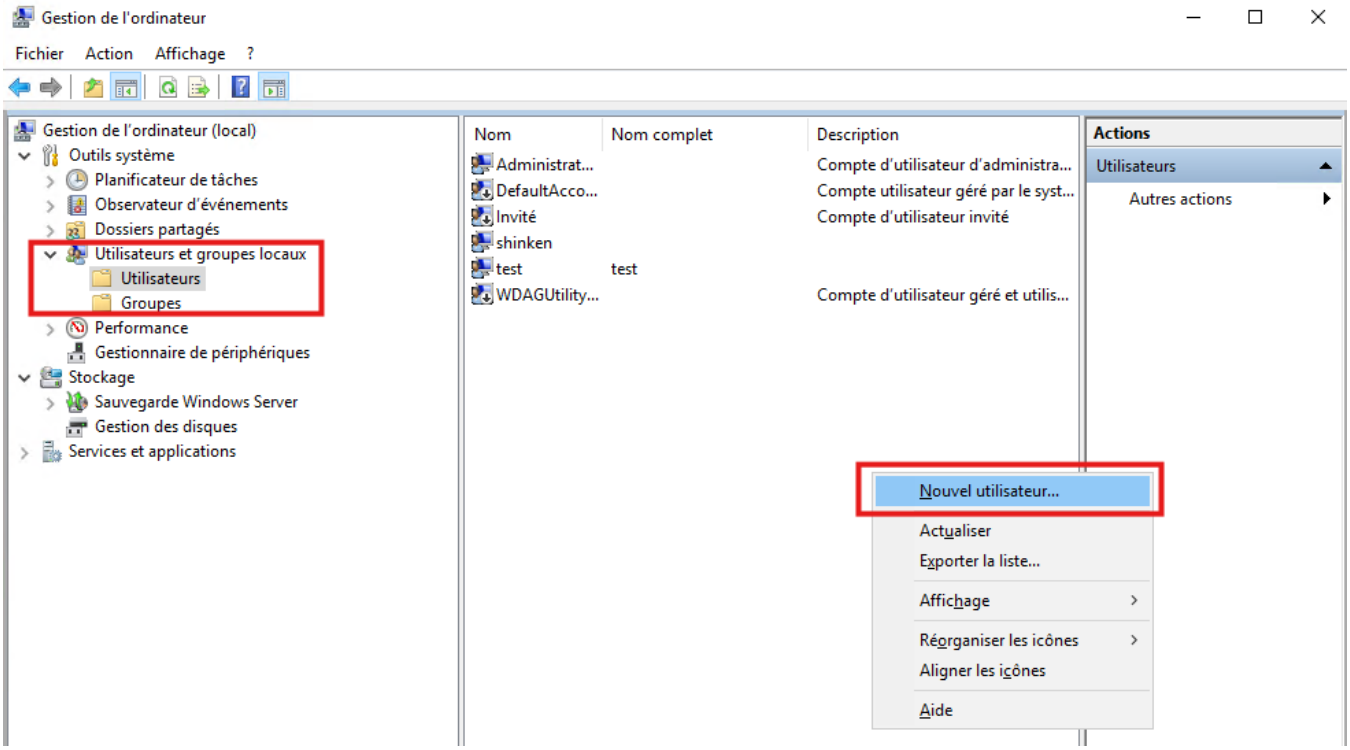
Cependant, pour des raisons de sécurité, il n'est pas conseillé de l'utiliser pour effectuer la supervision. Il faut alors créer un utilisateur qui exécutera les commandes demandées par les sondes, avec uniquement les droits nécessaires.

Voici la procédure pour créer un nouvel utilisateur dédié à la supervision avec les droits suffisant pour collecter les informations nécessaires au fonctionnement des sondes fournies par Shinken.

## Création de l'utilisateur

### Par interface

Sur le poste Windows à superviser, ouvrir "**Gestion de l'ordinateur**" ( *compmgmt.msc* ) puis dans **Utilisateurs et groupes locaux > Utilisateurs,** clic droit et **Nouvel utilisateur...**




### Par ligne de commande

Dans un PowerShell en Administrateur sur le poste Windows supervisé :

```
& {
param(
    [Parameter(Mandatory=$true)]
    [string]$UserName,

    [Parameter(Mandatory=$true)]
    [string]$Password
)
net user $Username $Password /ADD

}
```

## Installation de la langue

Pour assurer l'interprétation des commandes Windows par la sonde, il est nécessaire d'avoir une des langues suivantes, et de la configurer pour son utilisateur de supervision :

- Anglais ( en-US )
- Français ( fr-FR )

La commande suivante permet d'afficher les langues installées sur la machine :

```
Get-WinUserLanguageList
```

Voici un exemple de résultat de la commande.

```
LanguageTag     : en-US
Autonym         : English (United States)
EnglishName     : English
LocalizedName   : English (United States)
ScriptName      : Latin
InputMethodTips : {0409:0001040C}
Spellchecking   : True
Handwriting     : False

LanguageTag     : fr-FR
Autonym         : Français (France)
EnglishName     : French
LocalizedName   : French (France)
ScriptName      : Latin
InputMethodTips : {040C:0000040C}
Spellchecking   : True
Handwriting     : False
```

Si au moins une des langues nécessaires est installée, alors il faut la configurer pour l'utilisateur de supervision, passez à l'étape suivante.

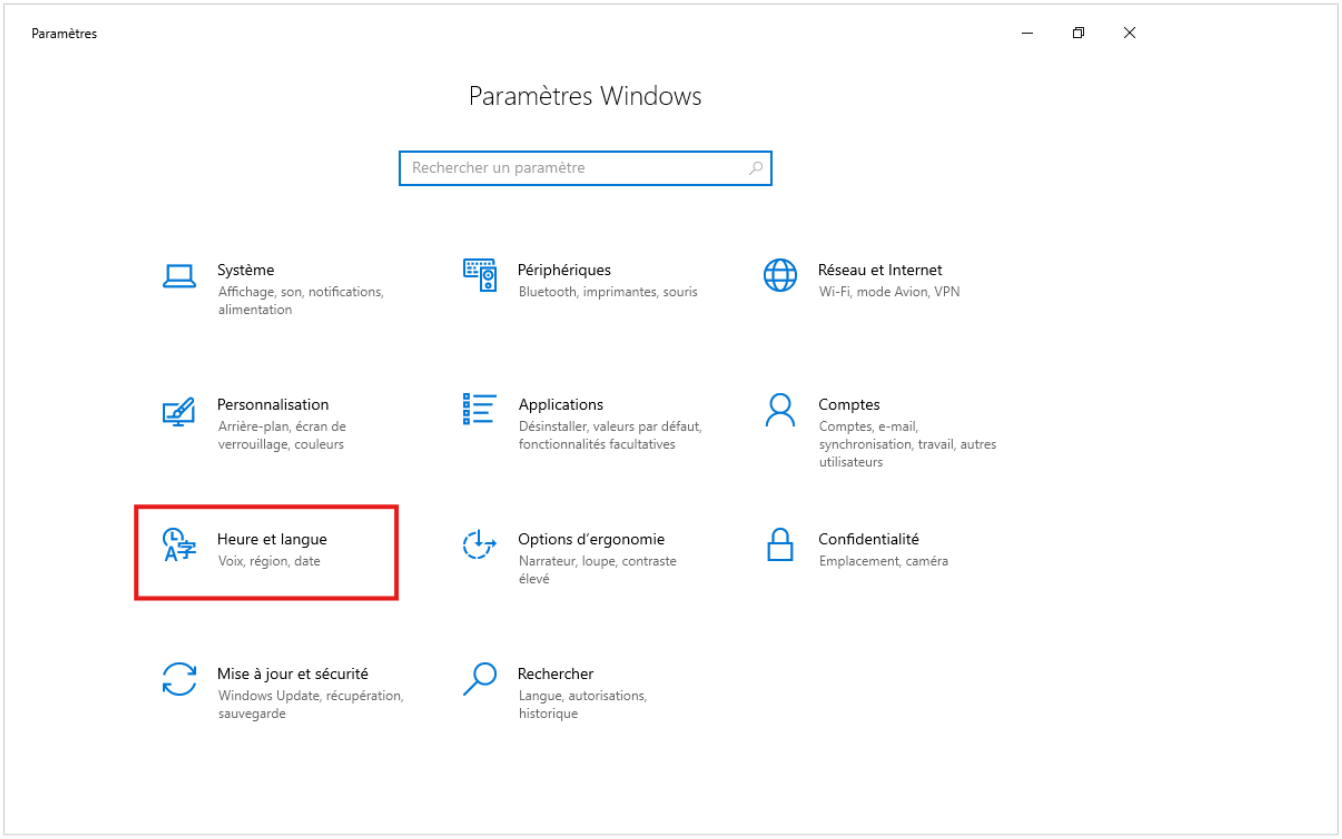
Sinon, installer une des langues requises.

### Par interface

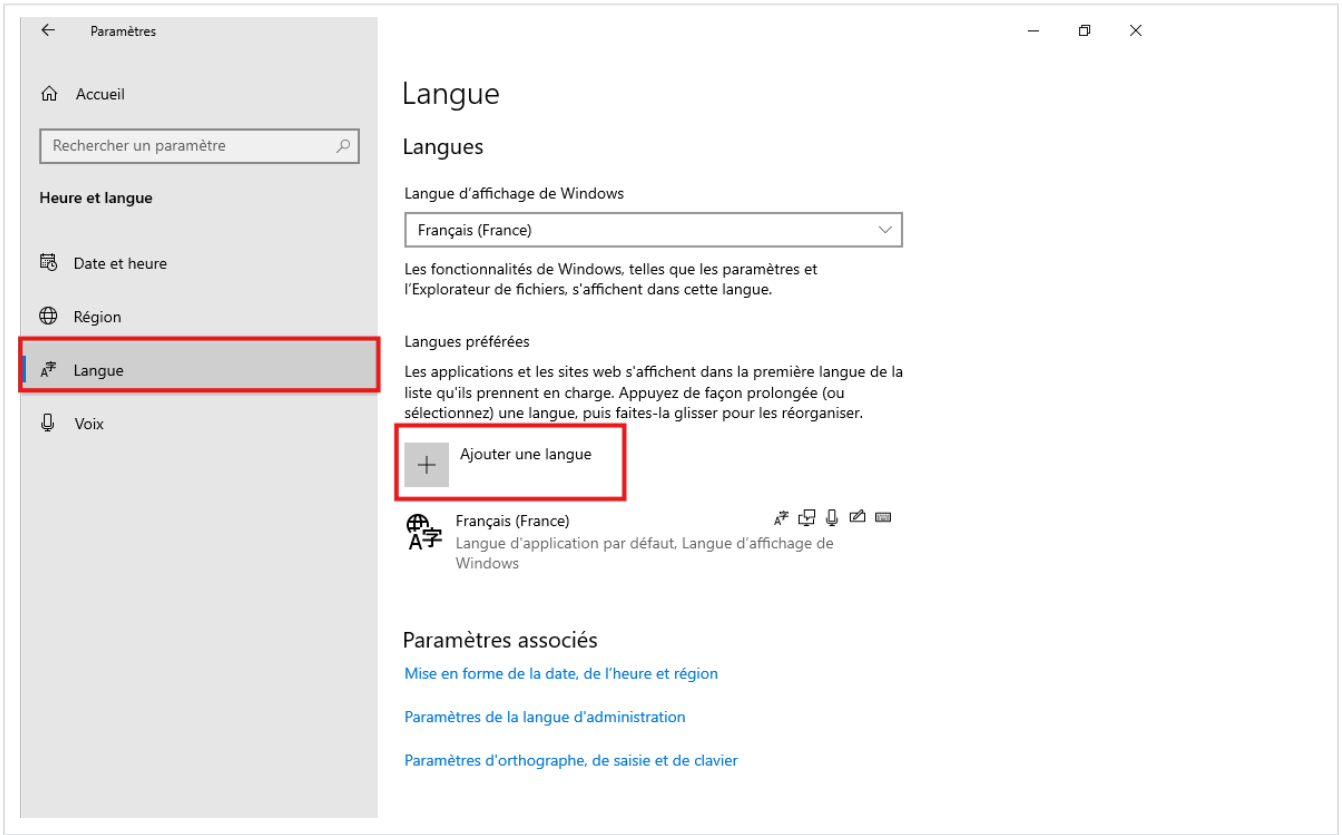
- **Installer la langue ( depuis le compte administrateur )**

La première étape est de télécharger la langue "**English (United-States)**" depuis le compte de l'administrateur.

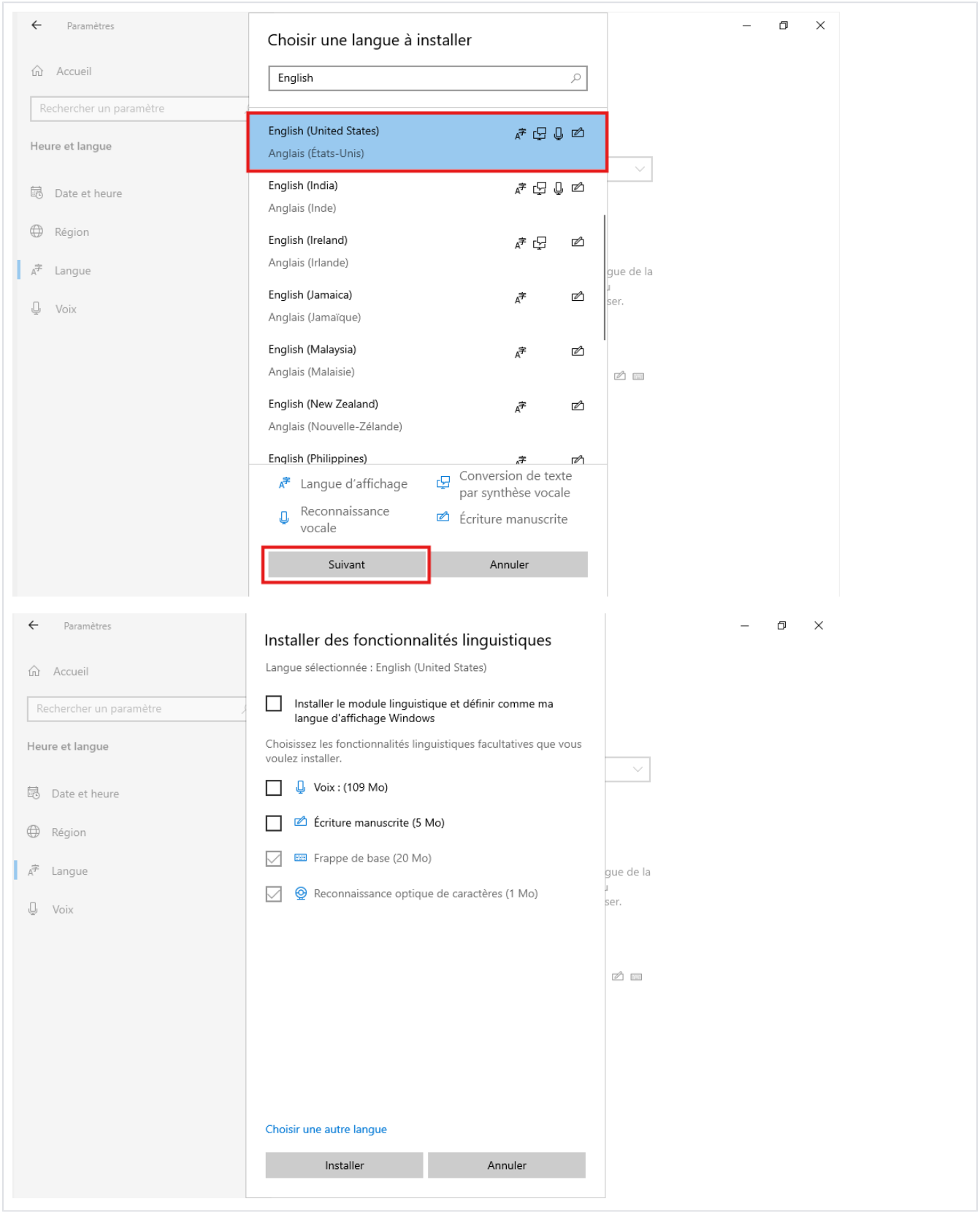
Lancer les paramètres Windows puis accéder à la catégorie "**Heure et Langue**".



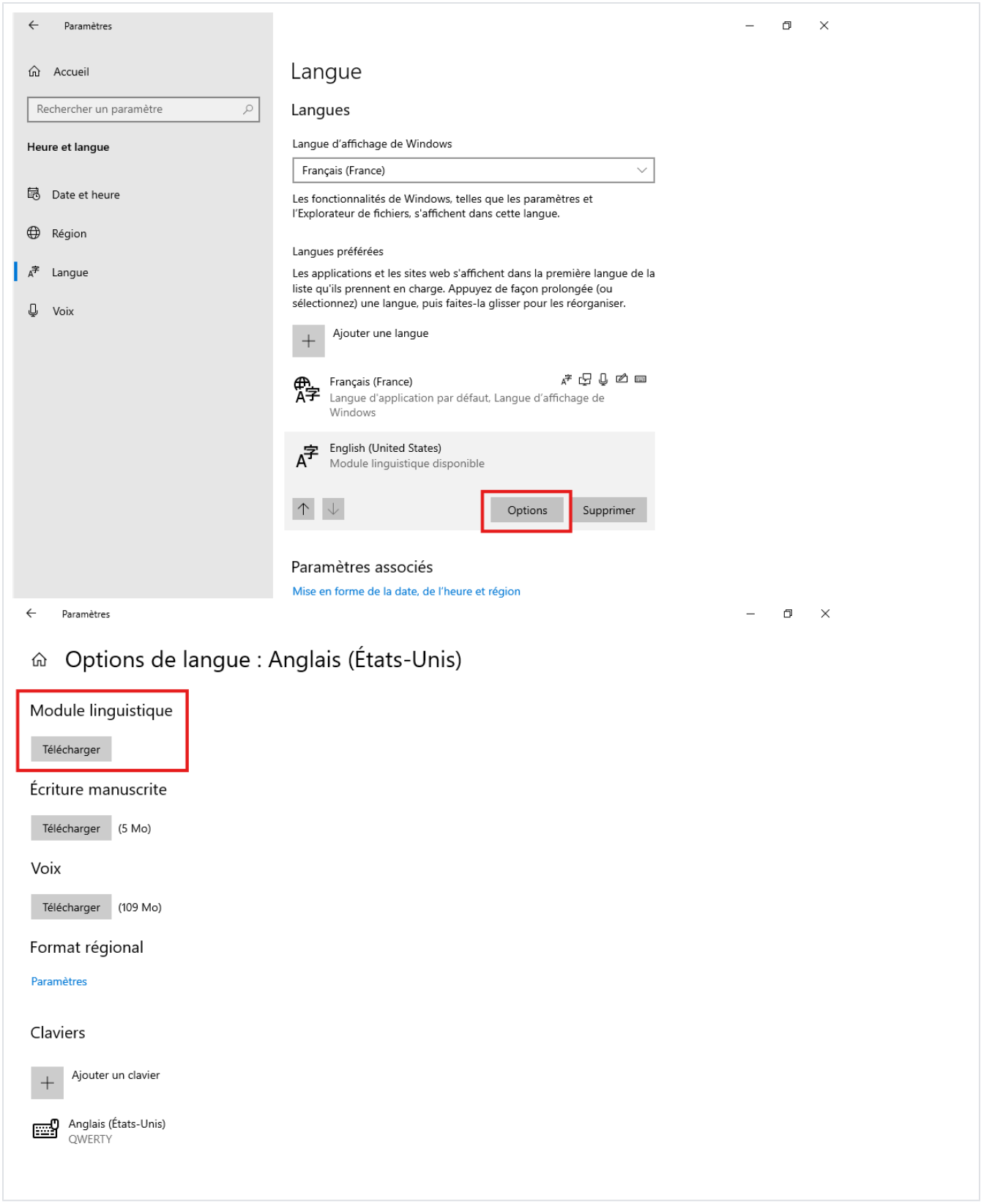

Ensuite, accéder à la sous-catégorie "**Langue**" puis sélectionner "**Ajouter une langue**"



Dans la nouvelle fenêtre, choisir "**English (United-States)**", et installer la langue.




La langue sera désormais listée. Cliquer dessus et sélectionner "**Option**", puis s'assurer que le **module linguistique** est installé.




La langue "**English (United-States)**" est désormais disponible sur la machine et il est **nécessaire de l'ajouter depuis le compte de supervision**.

Par ligne de commande

Après avoir ouvert un PowerShell en mode administrateur ( *sur le compte administrateur de la machine* ) :

La commande ci-dessous va installer le pack de langage nécessaire à la sonde.

> Cette commande n'est disponible que sur Windows 10, Windows 11 et Windows Server 2025.
> Dans les autres cas, la langue devra être téléchargée manuellement par interface.

Cette opération peut prendre 5 à 30 minutes en fonction de votre machine et de la charge des serveurs de téléchargement Windows.

```
# Installer la langue en Anglais
Install-Language -Language en-US -ExcludeFeatures

# Installer la langue en Francais
Install-Language -Language fr-FR -ExcludeFeatures
```

## Configuration de la langue

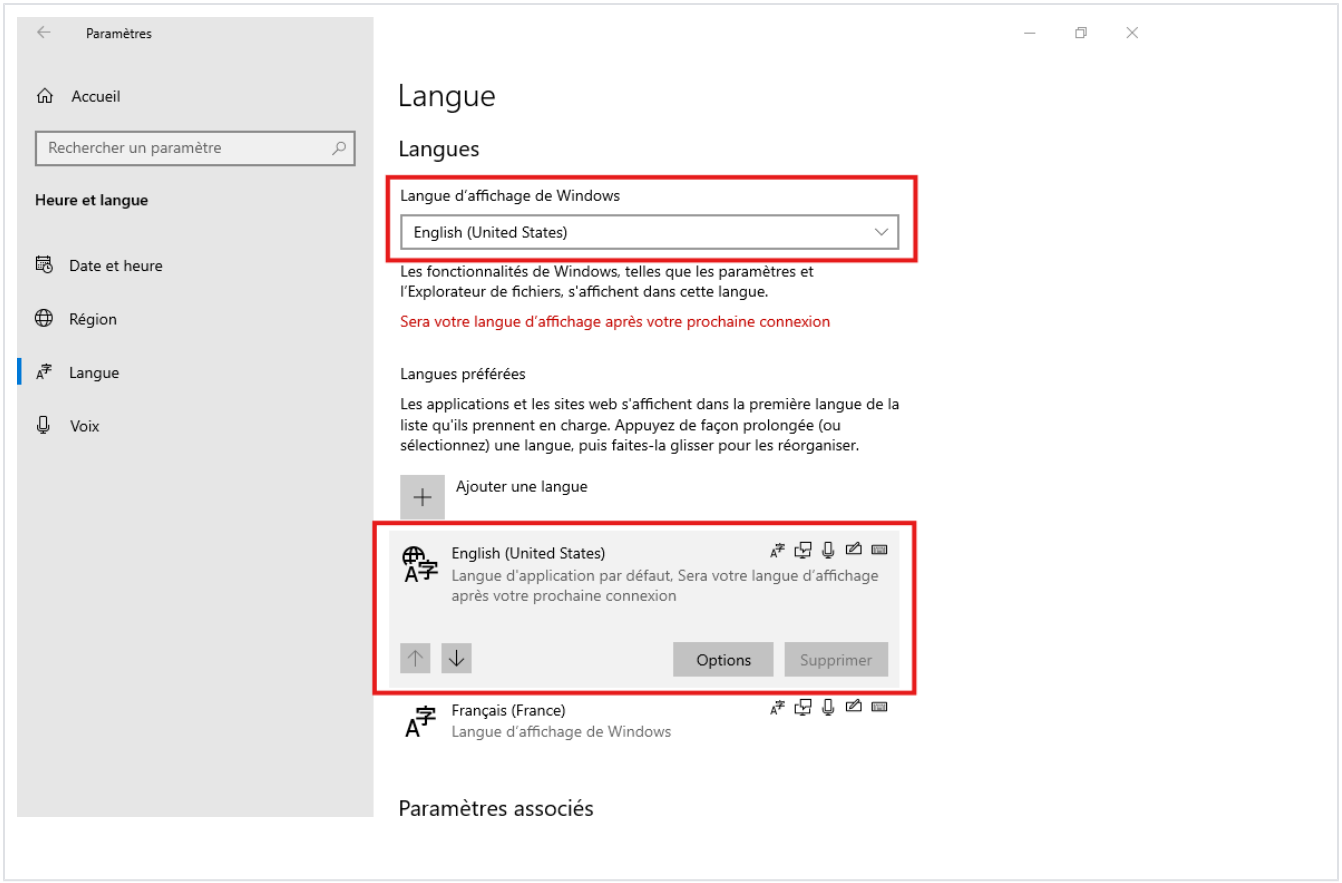
> Il est nécessaire de **se connecter au nouveau compte de supervision** créé pour changer la langue de l'utilisateur.

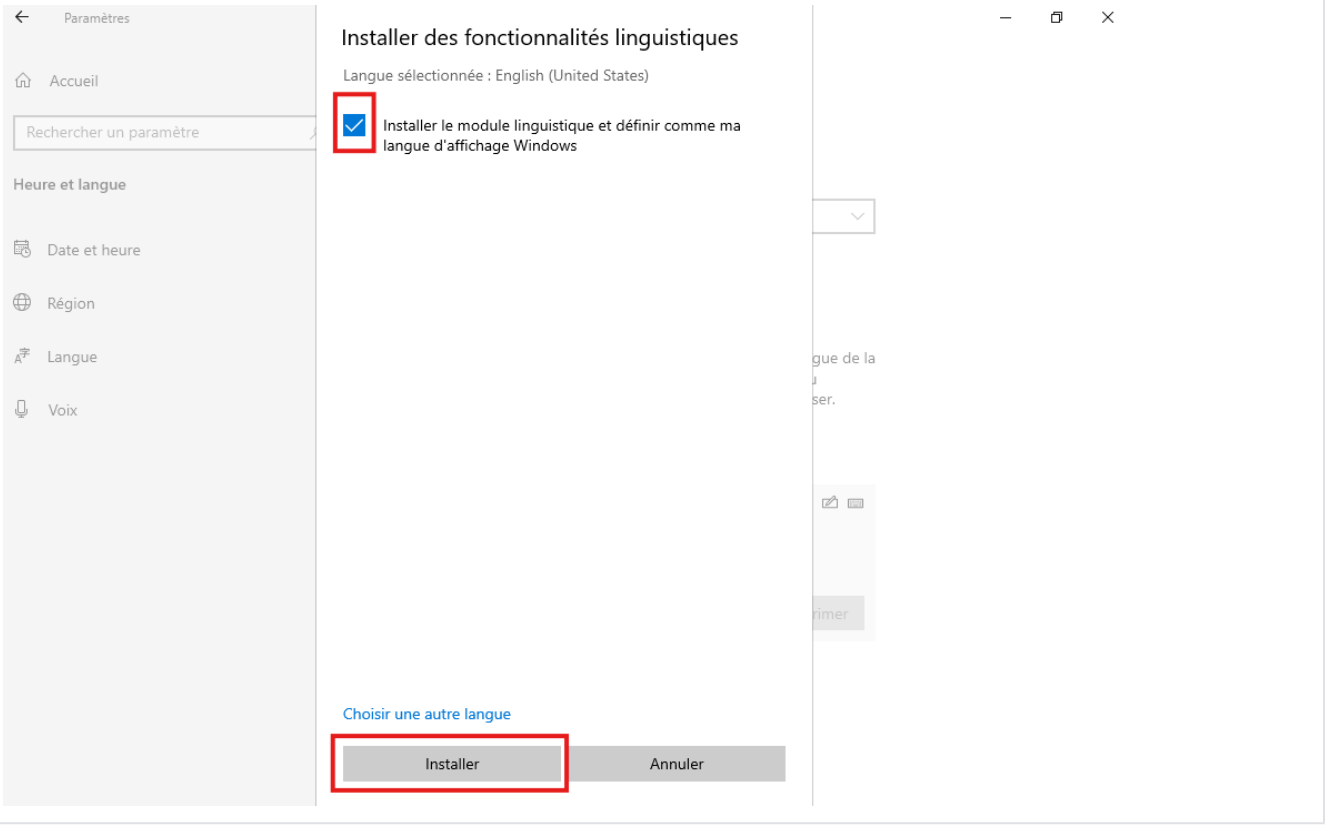
Répéter les mêmes étapes que pour l'installation de la langue par l'administrateur.

Lors de l'installation, sélectionner "**Installer le module linguistique et définir comme ma langue d'affichage Windows**".

Une fois installé, s'assurer que :

- La langue "**English (United-States)**" en première position de la liste des langues préférée.
- La langue "**English (United-States)**" dans la section "**Langue d'affichage de Windows**".






Par ligne de commande

Une fois le pack de langue téléchargé, il est possible de l'appliquer au nouvel utilisateur de supervision créé :

```
& {

param(
    [Parameter(Mandatory=$true)]
    [string]$UserName,

    [Parameter(Mandatory=$true)]
    [string]$Password
)

$LangCode = "en-US", "fr-FR"

$Computer = $env:COMPUTERNAME
$SID = (Get-WmiObject Win32_UserAccount -Filter "Name='$UserName' AND Domain='$Computer'").SID
if (-not $SID) { throw "Impossible de récupérer le SID pour $UserName" }

# Recupération du profil
$UserProfile = Get-CimInstance Win32_UserProfile | Where-Object SID -eq $SID

# Le profil peut ne pas exister si l'utilisateur a été créé mais jamais été connecté.
# Tentative de connection à l'utilisateur ...
if (-not $UserProfile) {
        $SecurePassword = ConvertTo-SecureString $Password -AsPlainText -Force
        $Credential = New-Object System.Management.Automation.PSCredential ($UserName, $SecurePassword)
    try {
        Start-Process -FilePath "whoami.exe" -Credential $Credential -WindowStyle Hidden -Wait
    } catch {
```

```
        Write-Host "Impossible de lancer un logon test pour $UserName ($_)" -ForegroundColor Red
    }

    # Récupération à nouveau du profil
    $UserProfile = Get-CimInstance Win32_UserProfile | Where-Object SID -eq $SID
}

if (-not $UserProfile) { throw "Profil introuvable pour $UserName" }

$HivePath = Join-Path $UserProfile.LocalPath "NTUSER.DAT"
if (-not (Test-Path $HivePath)) { throw "Profil non initialisé (NTUSER.DAT introuvable) : $HivePath" }

$regRelative = "Control Panel\Desktop"
$valueName = "PreferredUILanguages"

$IsLoaded = Test-Path "Registry::HKEY_USERS\$SID"
if ($IsLoaded) {
    $fullPath = "Registry::HKEY_USERS\$SID\$regRelative"
    if (-not (Test-Path $fullPath)) { New-Item -Path $fullPath -Force | Out-Null }
    if (-not (Get-ItemProperty -Path $fullPath -Name $valueName -ErrorAction SilentlyContinue)) {
        New-ItemProperty -Path $fullPath -Name $valueName -Value $LangCode -PropertyType MultiString -Force
| Out-Null
    } else {
        Set-ItemProperty -Path $fullPath -Name $valueName -Value $LangCode
    }
} else {
    Write-Host "Utilisateur non connecté. Chargement de la ruche..." -ForegroundColor Gray
    reg load "HKU\TempHive" $HivePath | Out-Null
    try {
        $tempPath = "Registry::HKEY_USERS\TempHive\$regRelative"
        if (-not (Test-Path $tempPath)) { New-Item -Path $tempPath -Force | Out-Null }
        if (-not (Get-ItemProperty -Path $tempPath -Name $valueName -ErrorAction SilentlyContinue)) {
            New-ItemProperty -Path $tempPath -Name $valueName -Value $LangCode -PropertyType MultiString -
Force | Out-Null
        } else {
            Set-ItemProperty -Path $tempPath -Name $valueName -Value $LangCode
        }
    } finally {
        reg unload "HKU\TempHive" | Out-Null
        Write-Host "Ruche déchargée." -ForegroundColor Gray
    }
}

Write-Host "Langue préférée utilisateur définie sur $LangCode (REG_MULTI_SZ)." -ForegroundColor Green
}
```

Ensuite, il est nécessaire de **se déconnecter puis se reconnecter** au nouvel utilisateur afin de correctement appliquer le changement de langue.

Une fois l'opération réalisée, il est possible de se reconnecter au compte administrateur Windows et de poursuivre la configuration.

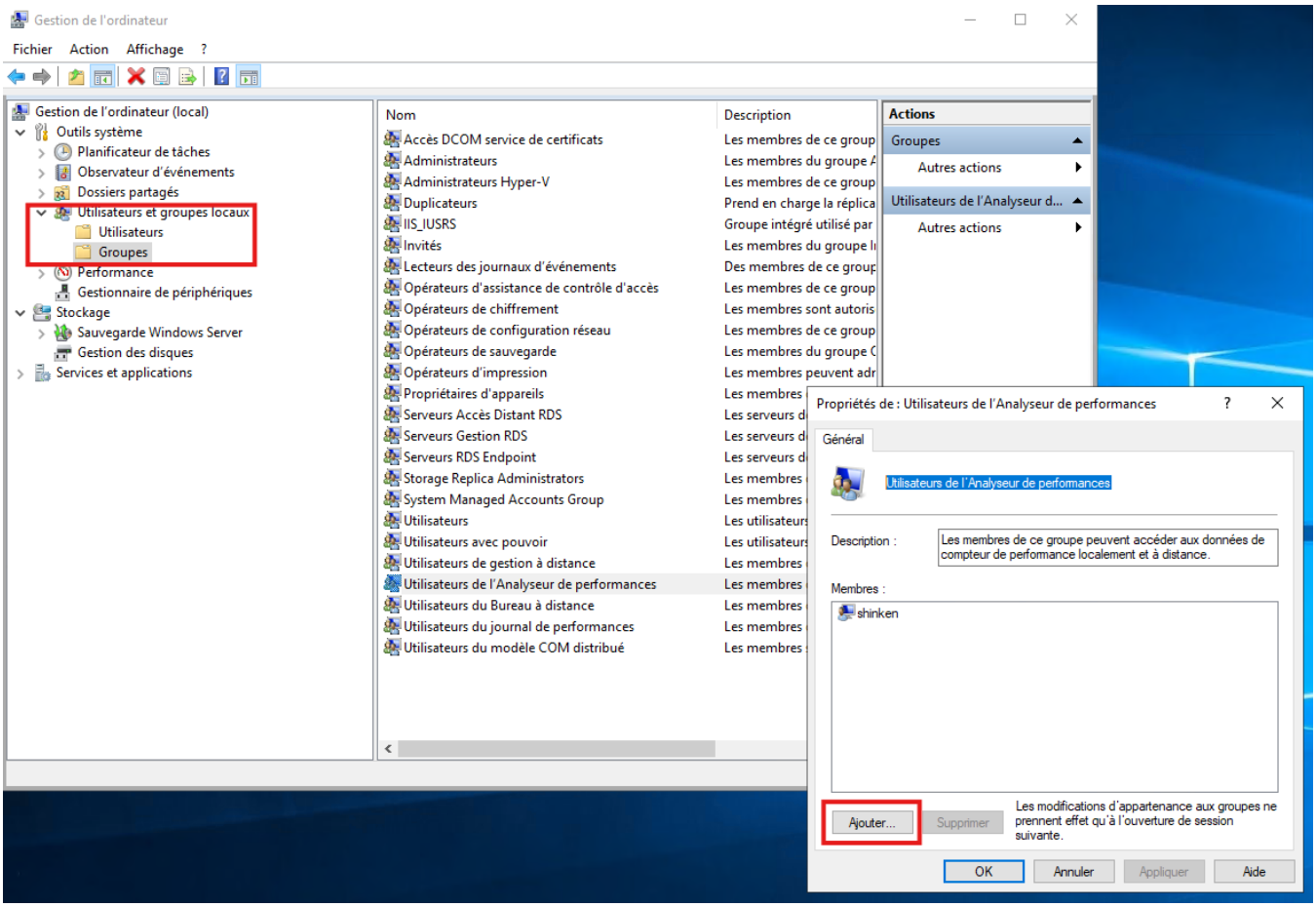
## Configuration des groupes

### Par interface

Une fois le nouvel utilisateur créé sur le poste client, et sa langue configurée, il faut ajouter sur le poste Windows concerné l'utilisateur dans les groupes suivants :

- **Utilisateurs de gestion à distance**
- **Utilisateurs de l'Analyseur de performances**

Cela s'effectue dans la console de "**Gestion de l'ordinateur**" ( *compmgmt.msc* ) puis dans **Utilisateurs et groupes locaux > Groupes,** clic droit sur le groupe puis **Propriétés**. Une nouvelle fenêtre s'ouvre, cliquer sur **Ajouter…** :




Par ligne de commande

Dans un PowerShell :

- Si la langue de l'utilisateur administrateur est français :

```
& {
param([Parameter(Mandatory=$true)][string]$UserName)

net localgroup "Utilisateurs de gestion à distance" $UserName /ADD
net localgroup "Utilisateurs de l'Analyseur de performances" $UserName /ADD

}
```

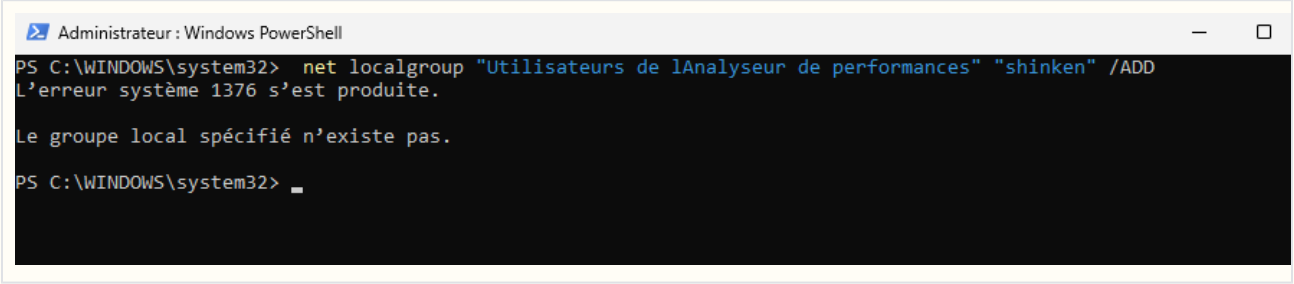
- Si la langue de l'utilisateur administrateur est anglais :

```
& {
param([Parameter(Mandatory=$true)][string]$UserName)

net localgroup "Remote Management Users" $UserName /ADD
net localgroup "Performance Monitor Users" $UserName /ADD

}
```

Il est possible que la commande "net localgroup "Utilisateurs de l'Analyseur de performances" $UserName /ADD" ne fonctionne pas lors d'un copier-coller.

Il y a un bug avec le terminal PowerShell sur le copier-coller qui ne retranscrira pas le symbole apostrophe ' lorsque l'action "coller" est déclenché avec un clic droit dans le terminal.
**Il faut utiliser le raccourci "CTRL+V"** afin de coller correctement la commande
Sinon, il est possible de générer le caractère spécial apostrophe ' avec le raccourci "ALT + 0146". ( *Différent de l'apostrophe de la touche 4* ).

Exemple du bug du copier-coller dans un terminal PowerShell Administrateur où l'apostrophe n'est pas retranscrite :




.

## Configuration des permissions

### Permissions WinRM pour l'utilisateur

Il est nécessaire d'ajouter les droits de lecture et d'exécution aux commandes WinRM au nouvel utilisateur de supervision :

#### Par interface

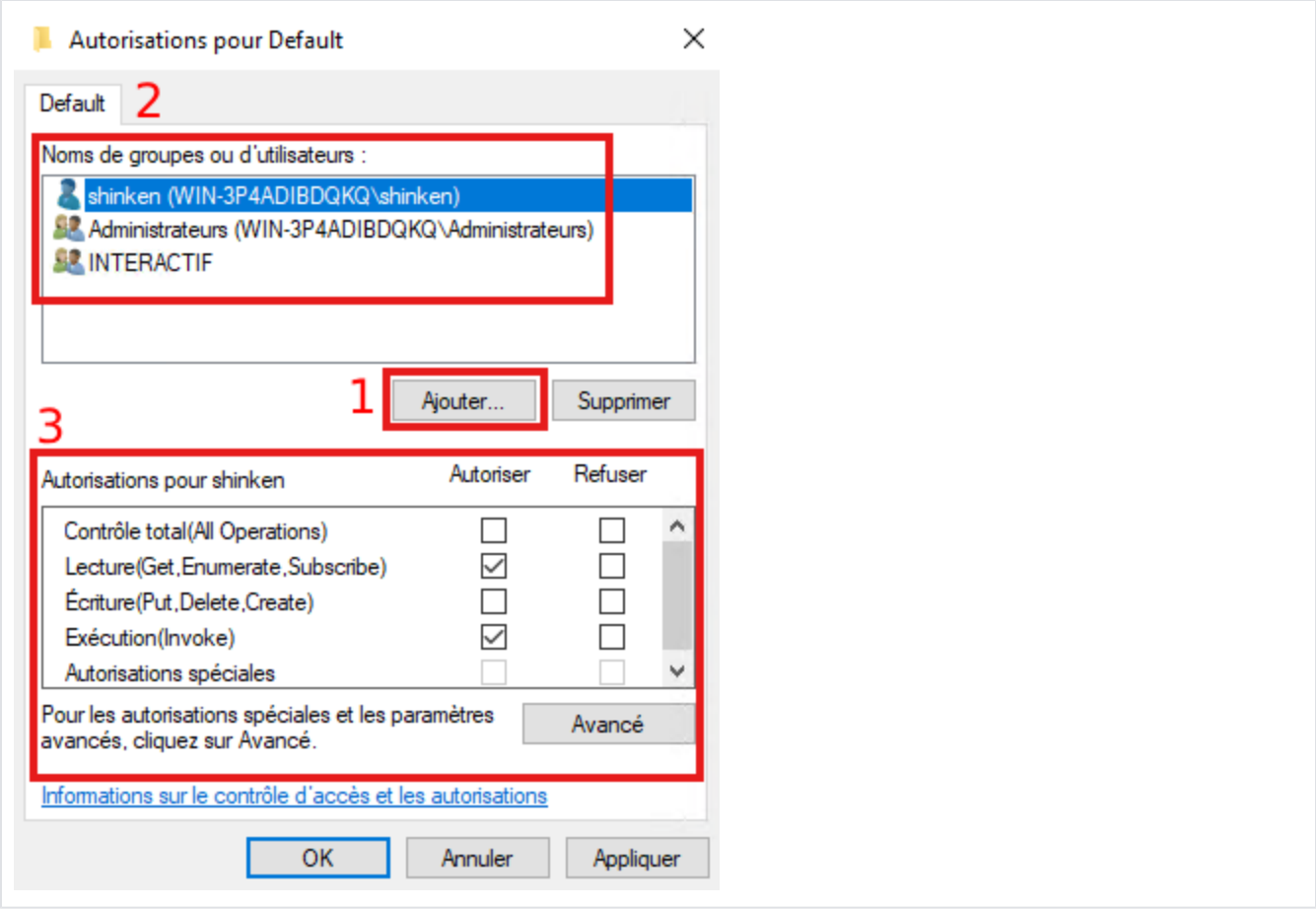
Dans une console PowerShell en Administrateur, exécuter la commande suivante :

```
winrm configSDDL default
```

Une nouvelle fenêtre s'ouvre. Dans celle-ci, ajouter le nouvel utilisateur et cliquer sur **Ajouter...** et attribuer les droits :

- Lecture(Get,Enumerate,Subscribe)
- Exécution(Invoke)

En cochant les cases correspondantes dans le tableau des droits situé en dessous de la liste des utilisateurs ( *cliquer sur l'utilisateur au préalable* ).




Par ligne de commande

Dans un PowerShell :

```
& {

param(
    [Parameter(Mandatory = $true)][string]$UserName
)

$GENERIC_READ = 0x80000000
$GENERIC_EXECUTE = 0x20000000

$user_sid = (New-Object -TypeName System.Security.Principal.NTAccount -ArgumentList $UserName).Translate
([System.Security.Principal.SecurityIdentifier])

# get the existing SDDL of the WinRM listener
$sddl = (Get-Item -Path WSMan:\localhost\Service\RootSDDL).Value

# convert the SDDL string to a SecurityDescriptor object
$sd = New-Object -TypeName System.Security.AccessControl.CommonSecurityDescriptor -ArgumentList $false,
$false, $sddl

# apply a new DACL to the SecurityDescriptor object
$sd.DiscretionaryAcl.AddAccess(
    [System.Security.AccessControl.AccessControlType]::Allow,
    $user_sid,
    ($GENERIC_READ -bor $GENERIC_EXECUTE),
    [System.Security.AccessControl.InheritanceFlags]::None,
    [System.Security.AccessControl.PropagationFlags]::None
)

# get the SDDL string from the changed SecurityDescriptor object
$new_sddl = $sd.GetSddlForm([System.Security.AccessControl.AccessControlSections]::All)

# apply the new SDDL to the WinRM listener
Set-Item -Path WSMan:\localhost\Service\RootSDDL -Value $new_sddl -Force
Write-Host "Permissions de 'Lecture' et 'Execution' WinRM ajoutées pour l'utilisateur $UserName" -
ForegroundColor Green
}
```

## Autorisation aux objets CIM

### Par interface

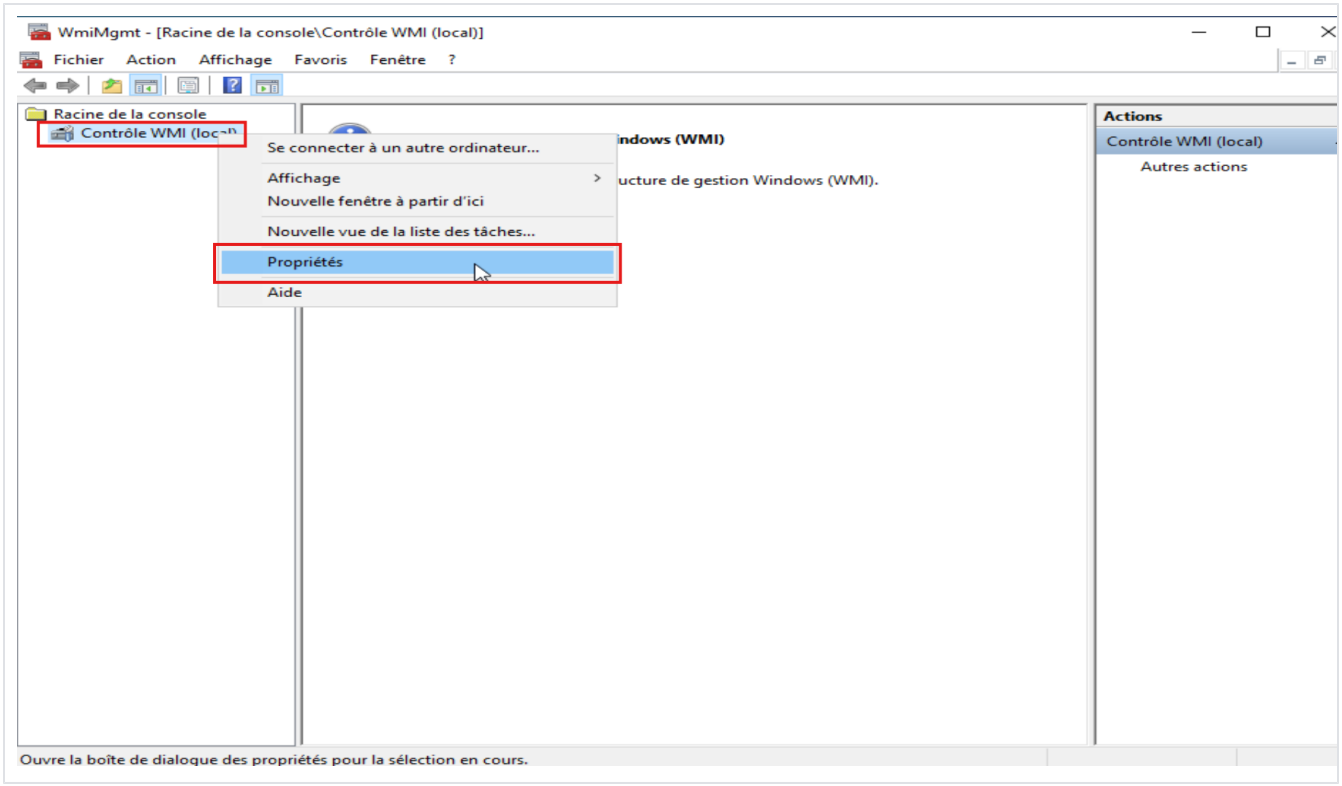
La sonde va demander les informations systèmes via les objets CIM, il est nécessaire d'ajouter les droits à l'utilisateur.

> Cette section n'est pas nécessaire si les machines supervisées sont configurées avec un Active Directory.

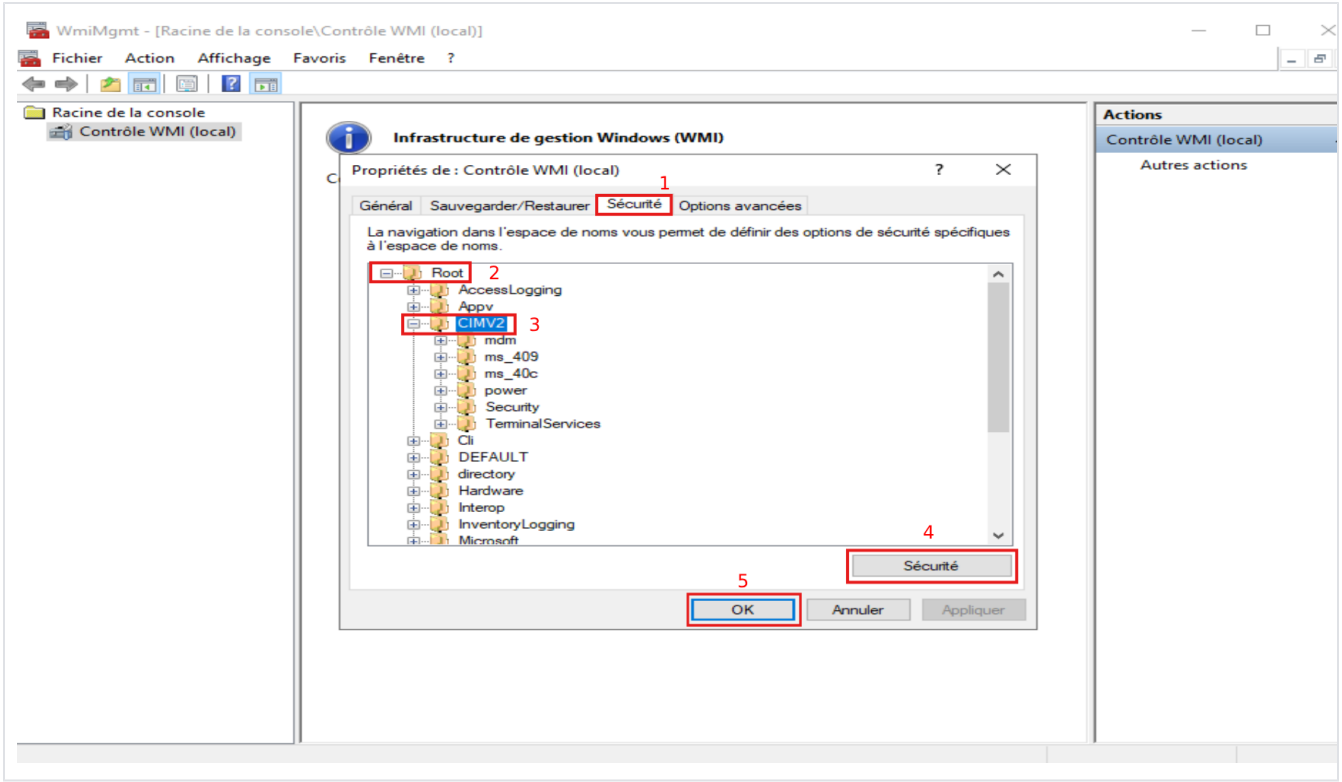
Il faut en premier temps lancer la fenêtre de contrôle WMI avec la commande :

```
wmimgmt.msc
```

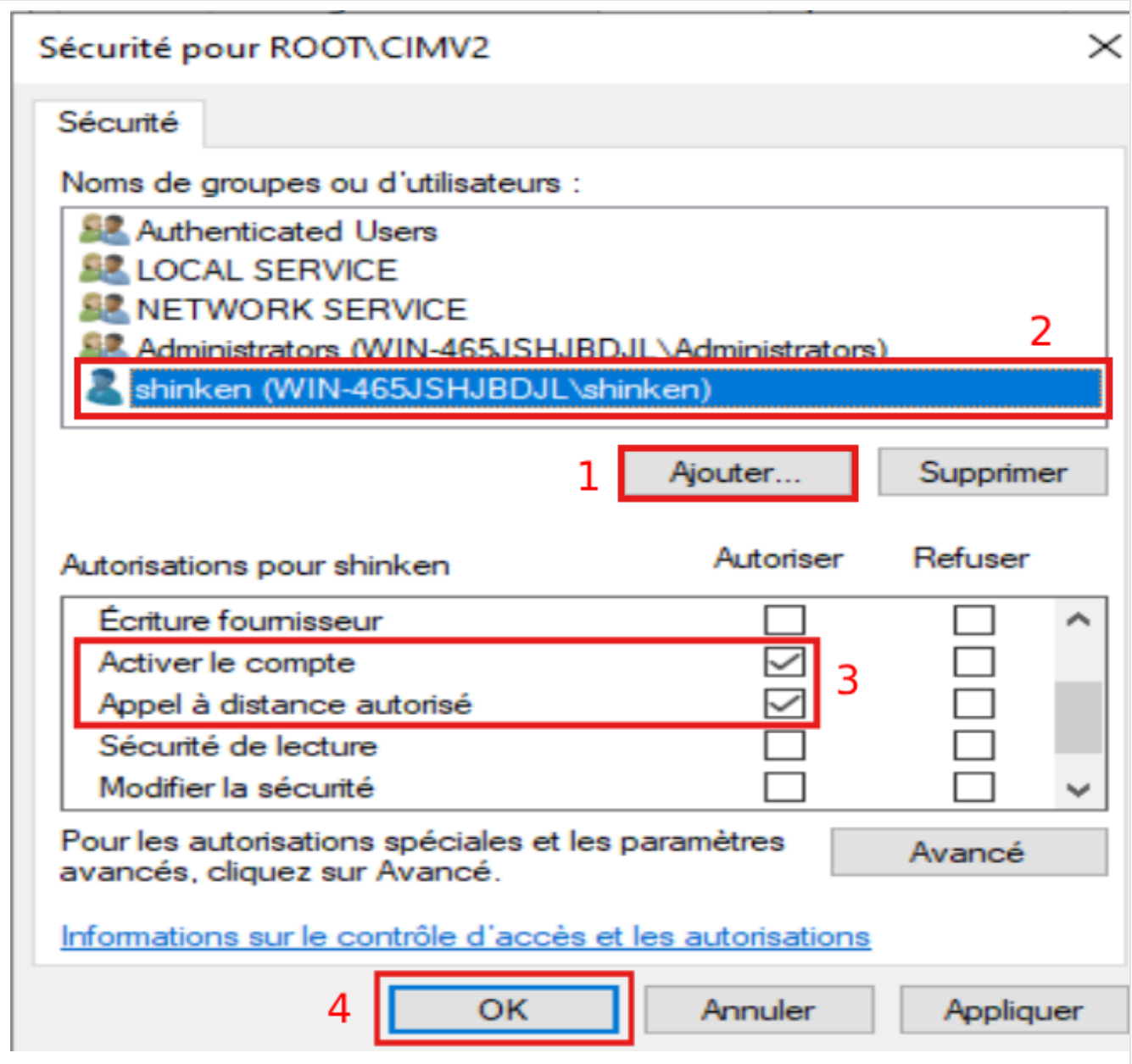
Une fois lancé, clic droit sur **Contrôle WMI (local)** puis sélectionner **Propriétés**.



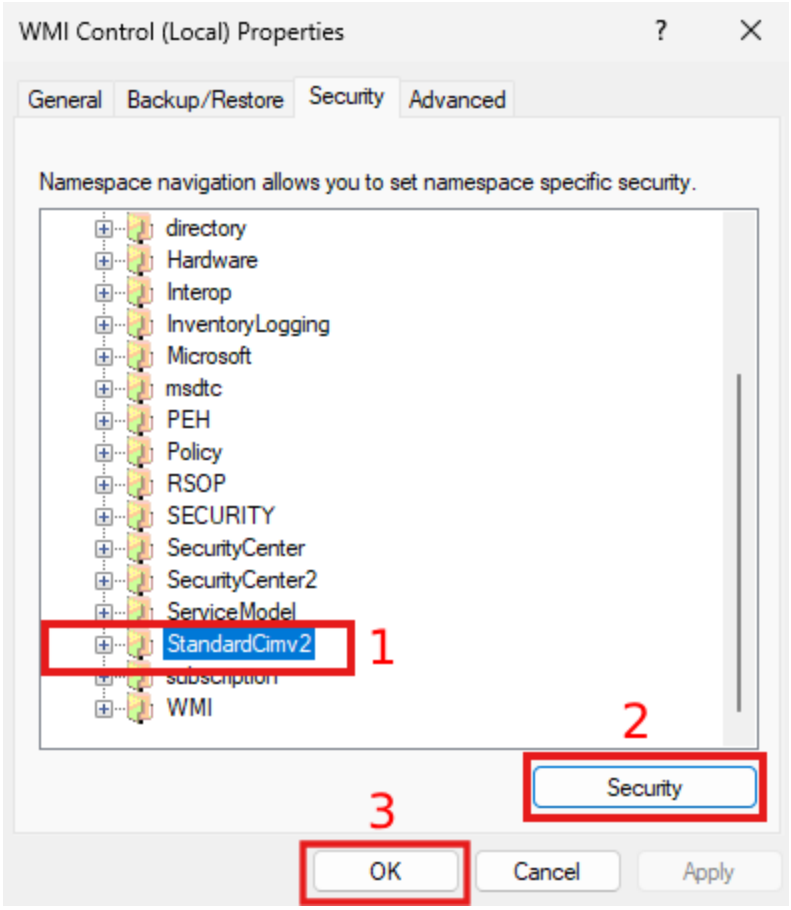

Accéder à la section Sécurité, puis dans l'arborescence ci-dessous, sélectionner **Root > CIMV2** puis cliquer sur le bouton **Sécurité** situé en bas à droite.




Enfin, ajouter l'utilisateur afin de lui appliquer de nouveaux droits, puis cocher **Activer le compte** et **Appel à distance autorisé**.




Ensuite, répétez l'opération pour **StandardCimV2**




Par ligne de commande

Dans un PowerShell en Administrateur. Ce script permet d'ajouter les droits nécessaires à la lecture des objets CIM.

```
&{

Param (
    [parameter(Mandatory=$true)][string] $username,
    [bool] $allowInherit = $false,
    [bool] $deny = $false,
    [string] $computerName = ".",
    [System.Management.Automation.PSCredential] $credential = $null
)

$namespaces = @("root\cimv2", "root\standardcimv2")
$permissions = @("Enable", "RemoteAccess")

foreach ($namespace in $namespaces) {
    Write-Host "Configuration des permissions pour le namespace: $namespace"
    $ErrorActionPreference = "Stop"

    Function Get-AccessMaskFromPermission($permissions) {
        $WBEM_ENABLE = 1
        $WBEM_METHOD_EXECUTE = 2
        $WBEM_FULL_WRITE_REP = 4
        $WBEM_PARTIAL_WRITE_REP = 8
        $WBEM_WRITE_PROVIDER = 0x10
        $WBEM_REMOTE_ACCESS = 0x20
        $READ_CONTROL = 0x20000
        $WRITE_DAC = 0x40000

        $WBEM_RIGHTS_FLAGS = $WBEM_ENABLE,$WBEM_METHOD_EXECUTE,$WBEM_FULL_WRITE_REP,`
```

```
        $WBEM_PARTIAL_WRITE_REP,$WBEM_WRITE_PROVIDER,$WBEM_REMOTE_ACCESS,`
        $READ_CONTROL,$WRITE_DAC

    $WBEM_RIGHTS_STRINGS = "Enable","MethodExecute","FullWrite","PartialWrite",`
        "ProviderWrite","RemoteAccess","ReadSecurity","WriteSecurity"

    $permissionTable = @{}
    for ($i = 0; $i -lt $WBEM_RIGHTS_FLAGS.Length; $i++) {
        $permissionTable.Add($WBEM_RIGHTS_STRINGS[$i].ToLower(), $WBEM_RIGHTS_FLAGS[$i])
    }

    $accessMask = 0
    foreach ($permission in $permissions) {
        if (-not $permissionTable.ContainsKey($permission.ToLower())) {
            throw "Unknown permission: $permission`nValid permissions: $($permissionTable.Keys)"
        }
        $accessMask += $permissionTable[$permission.ToLower()]
    }
    $accessMask
}

if ($PSBoundParameters.ContainsKey("Credential")) {
    $remoteparams = @{ComputerName=$computerName;Credential=$credential}
} else {
    $remoteparams = @{}
}

$invokeparams = @{Namespace=$namespace;Path="__systemsecurity=@"} + $remoteParams
$output = Invoke-WmiMethod @invokeparams -Name GetSecurityDescriptor
if ($output.ReturnValue -ne 0) {
    throw "GetSecurityDescriptor failed: $($output.ReturnValue)"
}

$acl = $output.Descriptor
$OBJECT_INHERIT_ACE_FLAG = 0x1
$CONTAINER_INHERIT_ACE_FLAG = 0x2
$computerName = (Get-WmiObject @remoteparams Win32_ComputerSystem).Name

if ($username.Contains('\')) {
    $domainaccount = $username.Split('\')
    $domain = $domainaccount[0]
    if (($domain -eq ".") -or ($domain -eq "BUILTIN")) {
        $domain = $computerName
    }
    $accountname = $domainaccount[1]
} elseif ($username.Contains('@')) {
    $domainaccount = $username.Split('@')
    $domain = $domainaccount[1].Split('.')[0]
    $accountname = $domainaccount[0]
} else {
    $domain = $computerName
    $accountname = $username
}

$getparams = @{Class="Win32_Account";Filter="Domain='$domain' and Name='$accountname'"} + $remoteParams
$win32account = Get-WmiObject @getparams
if ($win32account -eq $null) {
    throw "Account was not found: $username"
}

$accessMask = Get-AccessMaskFromPermission($permissions)
$ace = (New-Object System.Management.ManagementClass("win32_Ace")).CreateInstance()
        $ace.AccessMask = $accessMask

        if ($allowInherit) {
            $ace.AceFlags = $OBJECT_INHERIT_ACE_FLAG + $CONTAINER_INHERIT_ACE_FLAG
        } else {
            $ace.AceFlags = 0
        }

        $trustee = (New-Object System.Management.ManagementClass("win32_Trustee")).CreateInstance()
```

```
            $trustee.SidString = $win32account.Sid
            $ace.Trustee = $trustee

            $ACCESS_ALLOWED_ACE_TYPE = 0x0
            $ACCESS_DENIED_ACE_TYPE = 0x1

            if ($deny) {
                $ace.AceType = $ACCESS_DENIED_ACE_TYPE
            } else {
                $ace.AceType = $ACCESS_ALLOWED_ACE_TYPE
            }
            $acl.DACL += $ace.psobject.immediateBaseObject

    $setparams = @{Name="SetSecurityDescriptor";ArgumentList=$acl.psobject.immediateBaseObject} +
$invokeParams
    $output = Invoke-WmiMethod @setparams
    if ($output.ReturnValue -ne 0) {
        throw "SetSecurityDescriptor failed: $($output.ReturnValue)"
    }

    Write-Host "Permissions configurées avec succès pour $username sur $namespace"
}

Write-Host "Configuration terminée pour tous les namespaces"

}
```

### Autorisation au service W32Time

Afin de récupérer les informations de temps, il est nécessaire au nouvel utilisateur d'avoir certains accès en lecture au service de temps de Windows W32Time.

#### Par ligne de commande

Dans un PowerShell :

```
&{

Param (
    [parameter(Mandatory=$true)][string] $UserName
)

$sid = (New-Object System.Security.Principal.NTAccount($UserName)).Translate([System.Security.Principal.
SecurityIdentifier]).Value
$currentSddl = & sc.exe sdshow w32time
# Ajoute les droits suivants à l'utilisateur
# CC – SERVICE_QUERY_CONFIG
# LC – SERVICE_QUERY_STATUS
# LO – SERVICE_INTERROGATE
$newAce = "(A;;CCLCLO;;;${sid})"
$newSddl = $currentSddl + $newAce

sc.exe sdset w32time "$newSddl"

Restart-Service w32time

}
```

### Autorisation au journal de sécurité ( Security Event Log )

Afin de récupérer les tentatives de connexions échouées, il est nécessaire au nouvel utilisateur d'avoir certains accès en lecture au journal d'événements de sécurité.

Dans un PowerShell :

```
&{

Param (
    [parameter(Mandatory=$true)][string] $UserName
)
        try {
        $Path = "HKLM:\SYSTEM\CurrentControlSet\Services\EventLog\Security"
        $acl = Get-Acl $Path
        $rule = New-Object System.Security.AccessControl.RegistryAccessRule(
            $UserName,
            "ReadKey",
            "Allow"
        )
        $acl.AddAccessRule($rule)
        Set-Acl $Path $acl
        Write-Host "      Droits Registre appliqués." -ForegroundColor Green
    } catch {
        Write-Host "      Erreur permissions EventLog Security: $_" -ForegroundColor Red
        throw
    }
}
```

### Appliquer les permissions

Les nouvelles permissions attribuées au nouvel utilisateur seront appliquées lorsque de nouveaux tokens de connections seront générés. Il est possible de :

- Attendre quelques minutes que les anciens tokens expirent et que de nouveaux, ayant de nouvelles permissions, soient générés
- Redémarrer la machine

## Tester le pare-feu et WinRM

> Les tests suivants sont à effectuer sur le poste Windows à superviser.

### Vérifier le pare-feu Windows :

L'objectif de ce test est de vérifier si le pare-feu possède une règle qui autorise la connection WinRM.

- Exécuter la commande (en **administrateur**) :

  ```
  Get-NetFirewallRule -Name "WINRM-HTTP-In-TCP"
  ```

  - Résultat attendu :

    ```
    PS C:\> Get-NetFirewallRule -Name "WINRM-HTTP-In-TCP"


    Name                            : WINRM-HTTP-In-TCP
    DisplayName                     : Gestion à distance de Windows (HTTP-Entrée)
    Description                     : Règle de trafic entrant pour la gestion à distance de Windows via le service Gestion des services Web. [TCP 5985]
    DisplayGroup                    : Gestion à distance de Windows
    Group                           : @FirewallAPI.dll,-30267
    Enabled                         : True
    Profile                         : Domain, Private
    Platform                        : {}
    Direction                       : Inbound
    Action                          : Allow
    EdgeTraversalPolicy             : Block
    LooseSourceMapping              : False
    LocalOnlyMapping                : False
    Owner                           :
    PrimaryStatus                   : OK
    Status                          : La règle a été analysée à partir de la banque. (65536)
    EnforcementStatus               : NotApplicable
    PolicyStoreSource               : PersistentStore
    PolicyStoreSourceType           : Local
    RemoteDynamicKeywordAddresses :
    PolicyAppId                     :
    ```

    - Il faut s'assurer que :
      - ***"Enable"*** vaut ***"True"***.
      - ***"Action"*** vaut "***Allow***".
    - Si ce n'est pas le cas, activer la règle ( **toujours avec les droits administrateurs** ):

```
Enable-NetFirewallRule -Name "WINRM-HTTP-In-TCP"
```

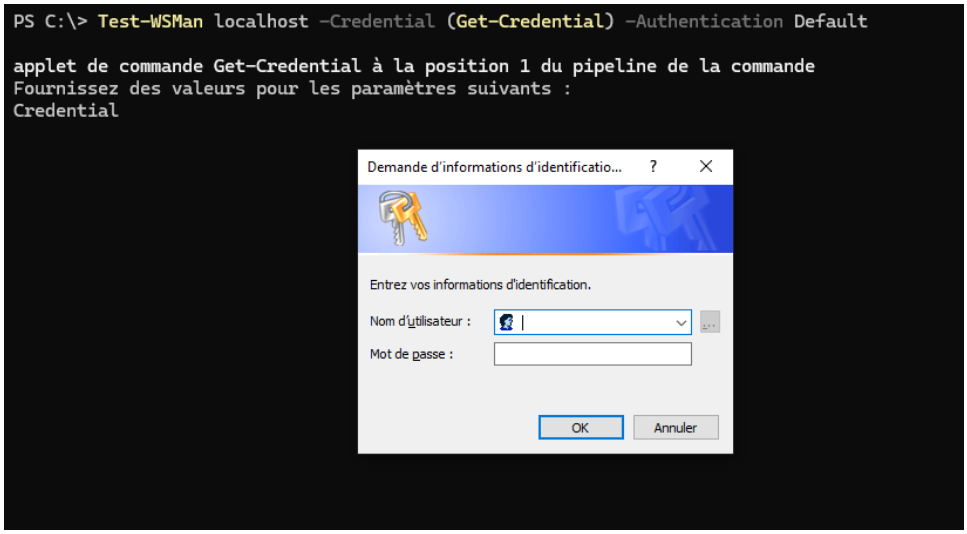
### Vérifier le service WinRM :

L'objectif de ce test est de réaliser en local une connexion WinRM, avec les outils fournis par Windows.

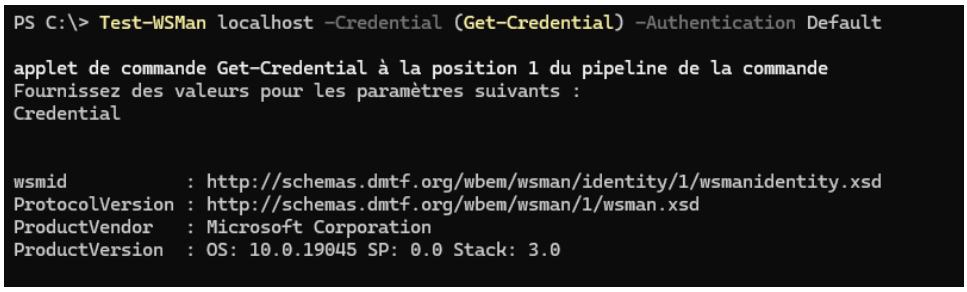
- Avec la commande **"Test-WsMan"** :

```
Test-WSMan localhost -Credential (Get-Credential) -Authentication Default
```

  - Rentrer le nom d'utilisateur ainsi que le mot de passe de l'utilisateur de supervision :




  - Résultat attendu :




    Cela confirme que le service **WinRM** et l'**utilisateur de supervision** sont actifs et opérationnels sur la machine.

Une fois ces deux tests validés, la configuration du poste Windows **est terminée** et il est prêt à être supervisé.

L'étape suivante est de choisir, d'accrocher et de paramétrer les modèles d'hôtes fournis dans le pack ( *Voir la page Modèles d'hôtes du pack windows-by-WinRM__shinken* ).