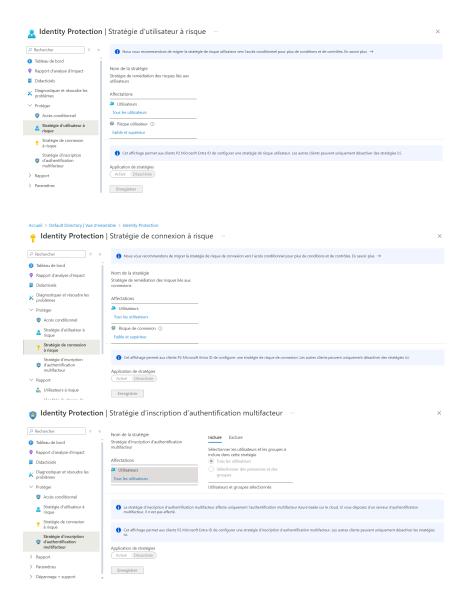
# **AD Entreprise**

# Sécurité avancée et Politiques de sécurité



#### Protection de l'identité

Affichez les utilisateurs à risque, les identités de charge de travail à risque et les connexions à risque dans votre locataire.



Cette partie est payante

### **Automatisation avec PowerShell**

#### I/ Utilisateurs

#### 1. Script : Ajouter une nouvelle recrue dans Entra ID

```
# Connexion à Entra ID (Azure AD)
Connect-MgGraph -Scopes "User.ReadWrite.All", "Group.ReadWrite.All"
# Informations sur la nouvelle recrue
$recrueDetails = @{
  accountEnabled = $true
  displayName = "Nouvelle Recrue"
  mailNickname = "nrecrue"
  userPrincipalName = "nouvelle.recrue@starfleet.com"
  passwordProfile = @{
    forceChangePasswordNextSignIn = $true
    password = "StarfleetP@ss123"
  }
  jobTitle = "Cadet"
  department = "Équipe Exploration"
}
# Création de la recrue dans Entra ID
$newRecrue = New-MgUser -BodyParameter $recrueDetails
Write-Output "Recrue créée avec ID: $($newRecrue.ld)"
# Ajout de la recrue dans le groupe "Équipe Exploration"
$groupName = "Equipe Exploration"
$groupId = (Get-MgGroup -Filter "displayName eq '$groupName'").Id
if ($null -ne $groupId) {
  Add-MgGroupMember -GroupId $groupId -DirectoryObjectId $newRecrue.Id
  Write-Output "Recrue ajoutée au groupe '$groupName'"
} else {
  Write-Output "Erreur : Le groupe '$groupName' n'a pas été trouvé."
}
```

### 2. Script : Transférer un utilisateur d'un vaisseau (ou groupe) à un autre

```
# Connexion à Entra ID (Azure AD)
Connect-MgGraph -Scopes "User.ReadWrite.All", "Group.ReadWrite.All"
# Paramètres du transfert
$userPrincipalName = "utilisateur.transfere@starfleet.com"
$ancienGroupe = "Vaisseau A"
$nouveauGroupe = "Vaisseau B"
# Récupération de l'utilisateur à transférer
$userId = (Get-MgUser -Filter "userPrincipalName eq '$userPrincipalName").Id
if ($null -ne $userId) {
  # Retrait de l'utilisateur de l'ancien groupe
  $ancienGroupId = (Get-MgGroup -Filter "displayName eg '$ancienGroupe'").Id
  if ($null -ne $ancienGroupId) {
    Remove-MgGroupMember -GroupId $ancienGroupId -DirectoryObjectId $userId
    Write-Output "Utilisateur retiré du groupe '$ancienGroupe'"
  } else {
    Write-Output "Erreur : Le groupe '$ancienGroupe' n'a pas été trouvé."
  }
  # Ajout de l'utilisateur dans le nouveau groupe
  $nouveauGroupId = (Get-MgGroup -Filter "displayName eq '$nouveauGroupe'").Id
  if ($null -ne $nouveauGroupId) {
    Add-MgGroupMember -GroupId $nouveauGroupId -DirectoryObjectId $userId
    Write-Output "Utilisateur ajouté au groupe '$nouveauGroupe'"
    Write-Output "Erreur : Le groupe '$nouveauGroupe' n'a pas été trouvé."
  }
} else {
  Write-Output "Erreur : L'utilisateur avec UPN '$userPrincipalName' n'a pas été trouvé."
}
```

## 3. Script : Mettre à jour les informations d'un utilisateur transféré

```
# Connexion à Entra ID (Azure AD)
Connect-MgGraph -Scopes "User.ReadWrite.All"

# Paramètres de mise à jour
$userPrincipalName = "utilisateur.transfere@starfleet.com"

# Nouveaux détails de l'utilisateur après transfert
$updatedUserDetails = @{
    jobTitle = "Officier Scientifique"
    department = "Recherche Intergalactique"
}

# Mise à jour des informations de l'utilisateur
Set-MgUser -UserId $userPrincipalName -JobTitle $updatedUserDetails.jobTitle
-Department $updatedUserDetails.department
Write-Output "Informations de l'utilisateur mises à jour : Titre =
$($updatedUserDetails.jobTitle), Département = $($updatedUserDetails.department)"
```

### 1. Script : Ajouter un utilisateur à un groupe spécifique

```
# Connexion à Entra ID (Azure AD) avec les permissions nécessaires
Connect-MgGraph -Scopes "User.ReadWrite.All", "Group.ReadWrite.All"
# Paramètres pour l'ajout
$userPrincipalName = "nouveau.membre@starfleet.com" # UPN de l'utilisateur
$groupName = "Equipe Exploration"
                                               # Nom du groupe cible
# Récupération de l'utilisateur
$userId = (Get-MgUser -Filter "userPrincipalName eq '$userPrincipalName'").Id
if ($null -ne $userId) {
  # Récupération de l'ID du groupe cible
  $groupId = (Get-MgGroup -Filter "displayName eq '$groupName'").Id
  if ($null -ne $groupId) {
    # Ajout de l'utilisateur au groupe
    Add-MgGroupMember -GroupId $groupId -DirectoryObjectId $userId
    Write-Output "Utilisateur ajouté au groupe '$groupName'."
  } else {
    Write-Output "Erreur : Le groupe '$groupName' n'a pas été trouvé."
  }
} else {
  Write-Output "Erreur : L'utilisateur avec UPN '$userPrincipalName' n'a pas été trouvé."
}
```

## 2. Script : Supprimer un utilisateur d'un groupe spécifique

```
# Connexion à Entra ID (Azure AD)
Connect-MgGraph -Scopes "User.ReadWrite.All", "Group.ReadWrite.All"
# Paramètres pour la suppression
$userPrincipalName = "ancien.membre@starfleet.com" # UPN de l'utilisateur
$groupName = "Equipe Exploration"
                                              # Nom du groupe cible
# Récupération de l'utilisateur
$userId = (Get-MgUser -Filter "userPrincipalName eq '$userPrincipalName'").Id
if ($null -ne $userId) {
  # Récupération de l'ID du groupe cible
  $groupId = (Get-MgGroup -Filter "displayName eq '$groupName'").Id
  if ($null -ne $groupId) {
    # Suppression de l'utilisateur du groupe
    Remove-MgGroupMember -GroupId $groupId -DirectoryObjectId $userId
    Write-Output "Utilisateur retiré du groupe '$groupName'."
    Write-Output "Erreur : Le groupe '$groupName' n'a pas été trouvé."
  }
} else {
  Write-Output "Erreur : L'utilisateur avec UPN '$userPrincipalName' n'a pas été trouvé."
}
```

# 3. Script : Ajout de plusieurs utilisateurs dans des groupes en fonction de leur rôle

```
# Connexion à Entra ID (Azure AD)
Connect-MgGraph -Scopes "User.ReadWrite.All", "Group.ReadWrite.All"
# Définir les critères pour l'équipe médicale
$department = "Médical"
                                  # Département ou titre cible
$groupName = "Equipe Médicale"
                                       # Nom du groupe cible
# Récupération de l'ID du groupe cible
$groupId = (Get-MgGroup -Filter "displayName eq '$groupName'").Id
if ($null -ne $groupId) {
  # Recherche des utilisateurs dans Entra ID selon les critères
  $users = Get-MgUser -Filter "department eq '$department'"
  # Ajout de chaque utilisateur dans le groupe
  foreach ($user in $users) {
    Add-MgGroupMember -GroupId $groupId -DirectoryObjectId $user.Id
    Write-Output "Utilisateur $($user.DisplayName) ajouté au groupe '$groupName'."
  }
} else {
  Write-Output "Erreur : Le groupe '$groupName' n'a pas été trouvé."
```

## III/ Politiques de sécurité

#### Création d'une politique de sécurité

Accédez à Azure Active Directory > Sécurité > Accès conditionnel.

Créez une nouvelle politique en cliquant sur « Nouvelle politique ».

**Définissez les utilisateurs et groupes cibles** : Sélectionnez les groupes spécifiques (par exemple, les équipes de mission sensible) pour appliquer la politique.

**Définissez les conditions d'accès** : Choisissez les conditions comme l'exigence de MFA, la localisation géographique, ou les appareils conformes.

**Définissez les contrôles de session** : Activez des contrôles supplémentaires si nécessaire (comme l'accès restreint aux applications sensibles).

#### I/ Script PowerShell : Ajouter des utilisateurs à un groupe de mission sensible

```
# Connexion à Entra ID (Azure AD)
Connect-MgGraph -Scopes "User.ReadWrite.All", "Group.ReadWrite.All"
# Paramètres du groupe de mission sensible
$missionSensGroupName = "Mission Sensible"
$missionSensGroupId = (Get-MgGroup -Filter "displayName eq
'$missionSensGroupName'").Id
if ($null -ne $missionSensGroupId) {
  # Liste des utilisateurs à ajouter au groupe de mission sensible
  $userPrincipalNames = @("membre1@starfleet.com", "membre2@starfleet.com",
"membre3@starfleet.com")
  foreach ($upn in $userPrincipalNames) {
    $userId = (Get-MgUser -Filter "userPrincipalName eq '$upn'").Id
    if ($null -ne $userId) {
       Add-MgGroupMember - GroupId $missionSensGroupId - DirectoryObjectId $userId
       Write-Output "Utilisateur $upn ajouté au groupe de mission sensible."
    } else {
       Write-Output "Erreur : Utilisateur $upn non trouvé."
} else {
  Write-Output "Erreur : Le groupe de mission sensible n'a pas été trouvé."
}
```

# II/ Automatiser la Mise à Jour des Politiques de Sécurité avec Microsoft Graph API (Facultatif)

```
# Connexion avec des autorisations d'administration
Connect-MgGraph -Scopes "Policy.ReadWrite.ConditionalAccess"

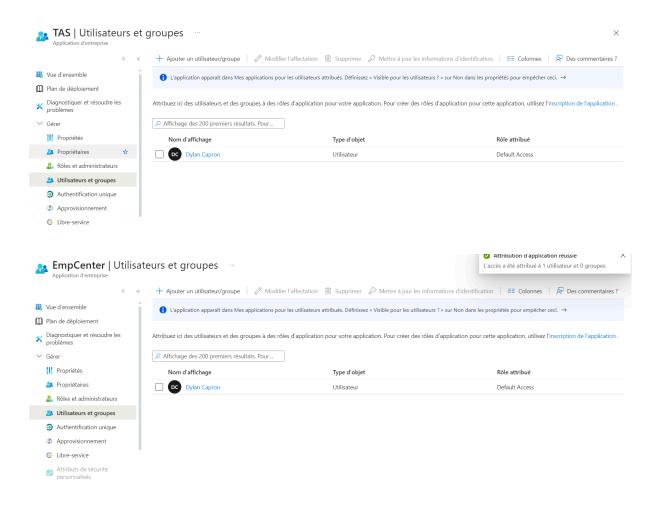
# Exemple de récupération des politiques d'accès conditionnel
$policies = Get-MgConditionalAccessPolicy
foreach ($policy in $policies) {
    Write-Output "Nom de la politique : $($policy.DisplayName)"
}

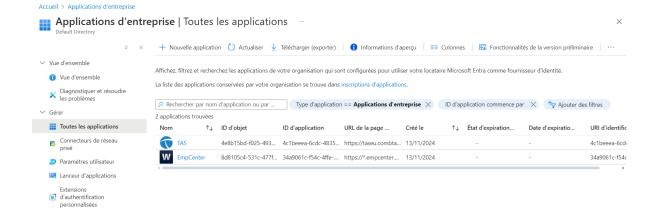
# Modification d'une politique d'accès conditionnel pour exiger MFA
$policyId = "<ID de la politique>"
Update-MgConditionalAccessPolicy -ConditionalAccessPolicyId $policyId -Conditions @{
    "clientAppTypes" = @("Browser") } -GrantControls @{ "builtInControls" = @("mfa") }
Write-Output "Politique mise à jour pour exiger MFA."
```

Exécuter les scripts pour vérifier s'ils fonctionnent correctement

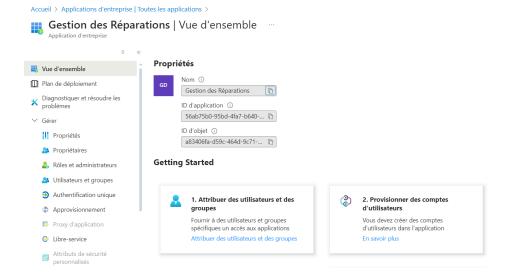
# Intégration et Sécurisation des Applications

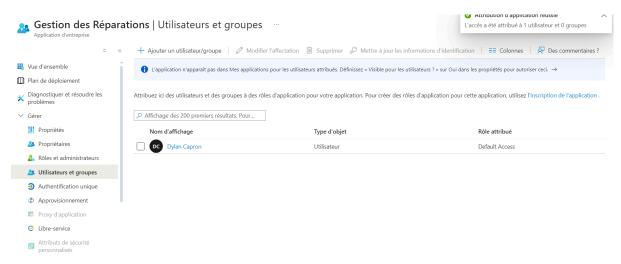
## I/ Intégration d'une application SaaS avec Entra ID



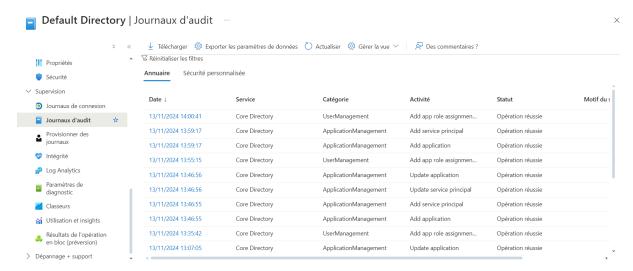


## II/ Ajouter une application personnalisée :





# Surveillance et Réponse aux Incidents



## Créer un espace de travail Log Analytics

Grâce aux journaux Azure Monitor, vous pouvez facilement stocker, conserver et interroger les données collectées à partir de vos ressources supervisées dans Azure et d'autres environnements pour obtenir des insights intéressants. Un espace de travail Log Analytics est l'unité de stockage logique où vos données de journal sont collectées et stockées.

#### Détails du projet

Sélectionnez l'abonnement pour gérer les coûts et les ressources déployées. Utilisez les groupes de ressources comme les dossiers pour organiser et gérer toutes vos ressources.

Abonnement * ①	(Désactivé) Azure for Students	~
Groupe de ressources * ①		~
	Créer nouveau	
Détails de l'instance		
Nom * ①		
Région * ①	East US	