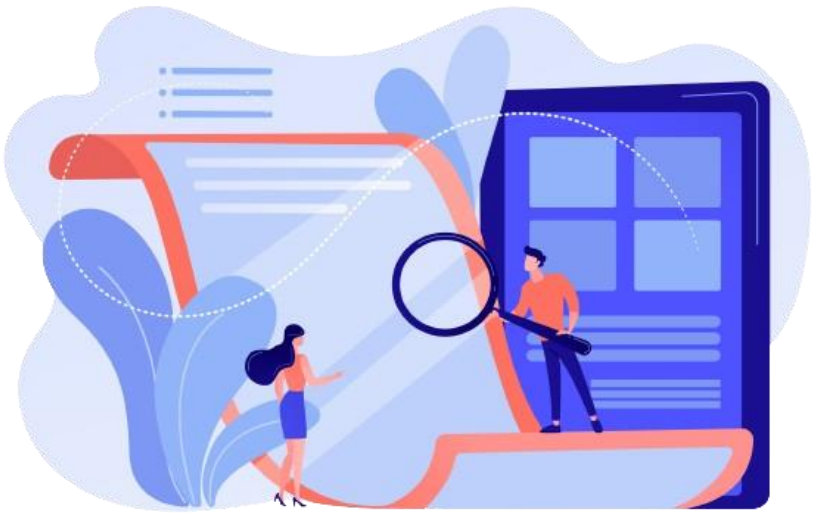
BTS Services Informatiques aux Organisations

Option Solutions d'Infrastructure, Systèmes et Réseaux
(S.I.S.R)



Documentation Technique

Épreuve E5 - Situation 2

WAÏ-LUNE Nathan

Session 2024

Historique des modifications

Par	Date	Description
WAÏ-LUNE Nathan	06/06/2024	Première Version du document

Table des matières

Mode Opératoire Active Directory / Contrôleur de domaine

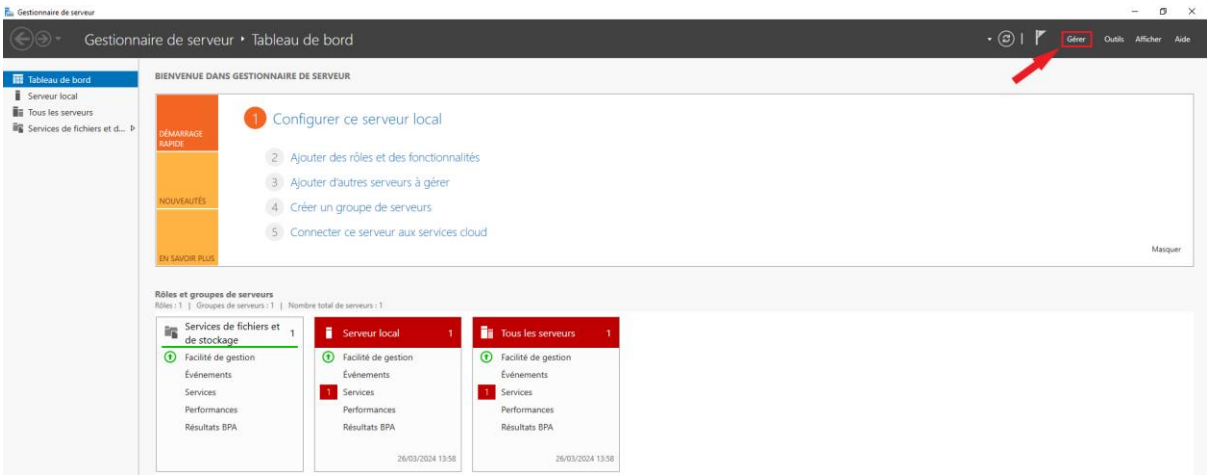
Ce mode opératoire explique comment créer un domaine sur Windows Server 2019. Pour ce faire, nous installerons le rôle "Active Directory Domain Service".

Les services de domaine Active Directory (AD DS) stockent des informations à propos des objets sur le réseau et rendent ces informations disponibles pour les utilisateurs et les administrateurs du réseau. Les services AD DS utilisent les contrôleurs de domaine pour donner aux utilisateurs du réseau un accès aux ressources autorisées n'importe où sur le réseau via un processus d'ouverture de session unique.

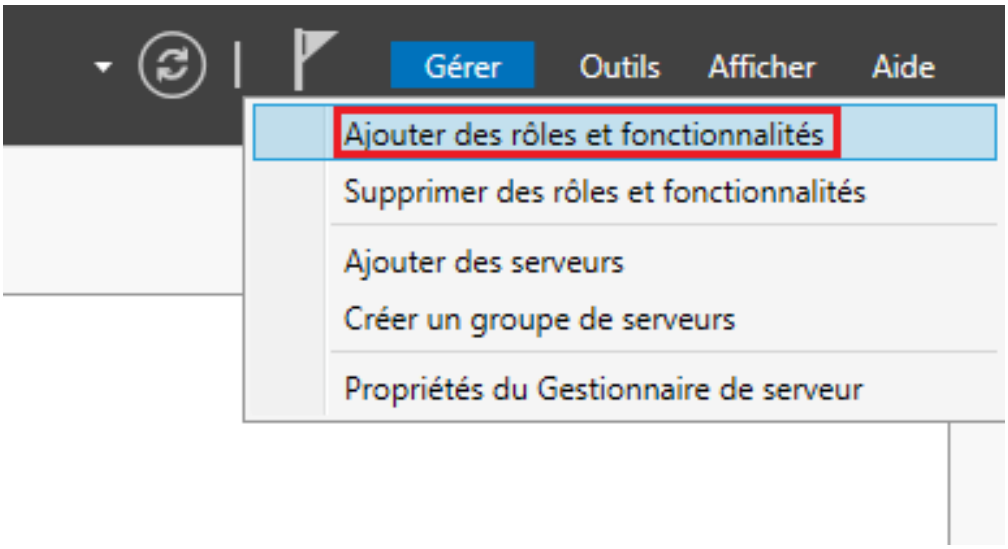
AD DS aide les administrateurs à gérer de manière sécurisée ces informations et facilite la collaboration entre les utilisateurs d'un même domaine.

Installation du service AD DS

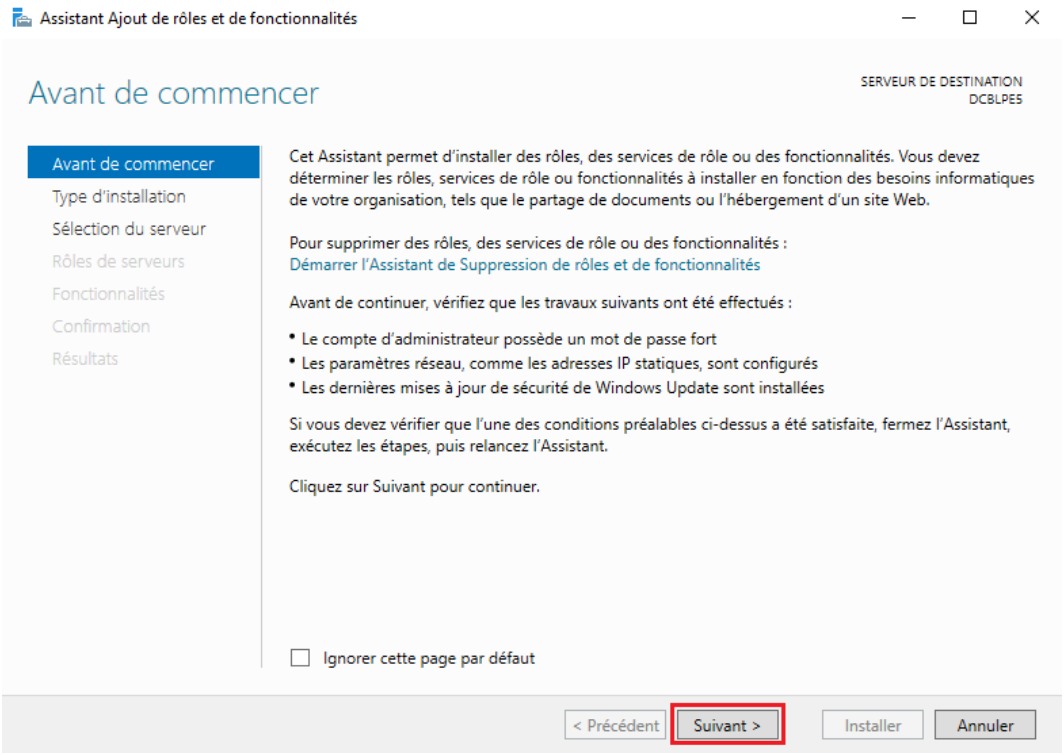
Après avoir installé Windows Server 2019, il vous faudra ouvrir le gestionnaire de serveur (s'il ne s'est pas ouvert automatiquement), cliquer sur Gérer :



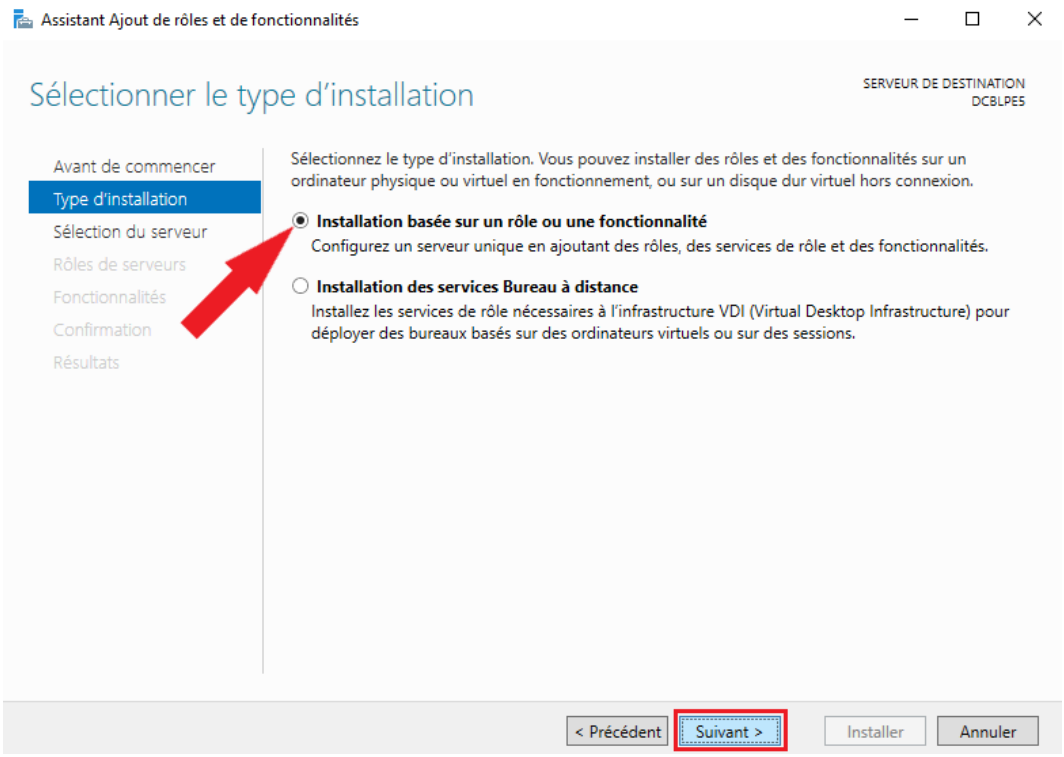
Puis « Ajouter des rôles et fonctionnalités » :



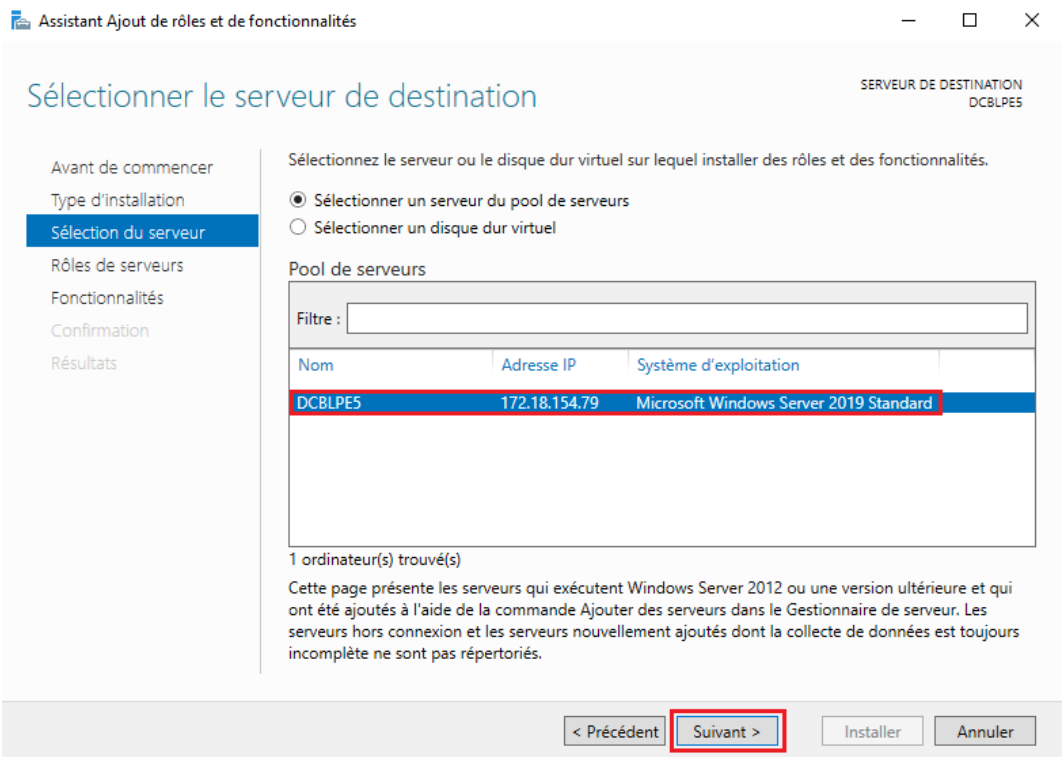
Dans l'assistant d'Ajout de rôles et de fonctionnalités, lisez attentivement les informations qui vous sont présentées et cliquez sur suivant :



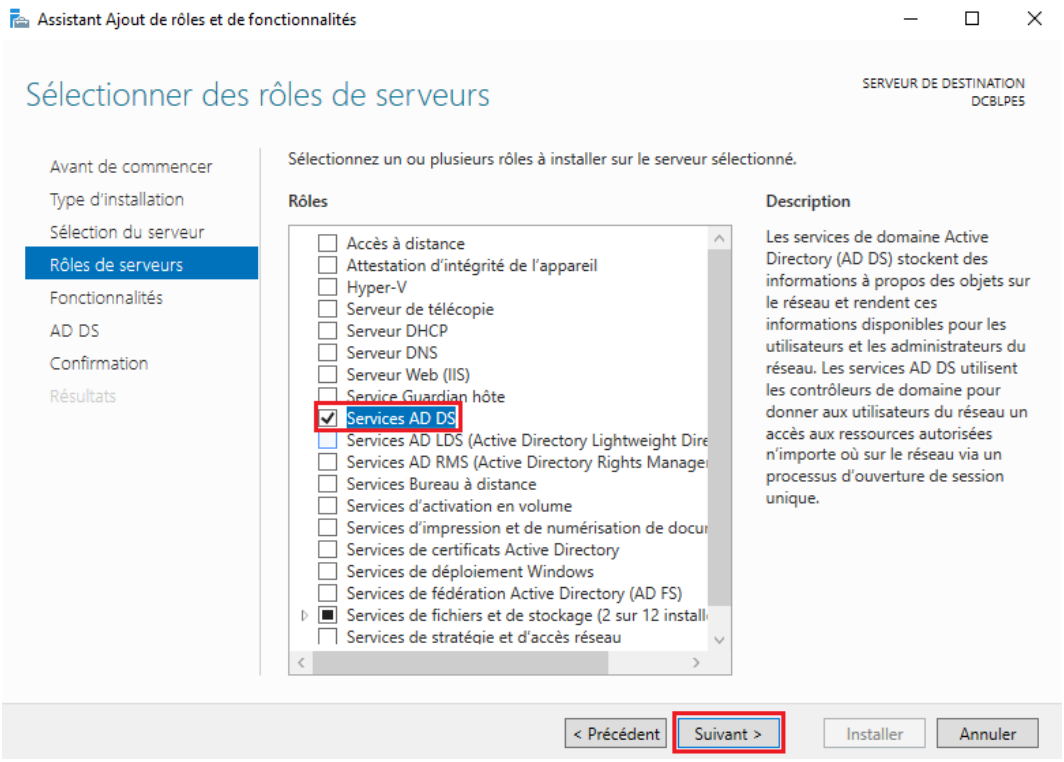
Sélectionner le type d'installation de votre choix, dans cette situation, on choisira une Installation basée sur un rôle ou une fonctionnalité. Ensuite, cliquez sur suivant :



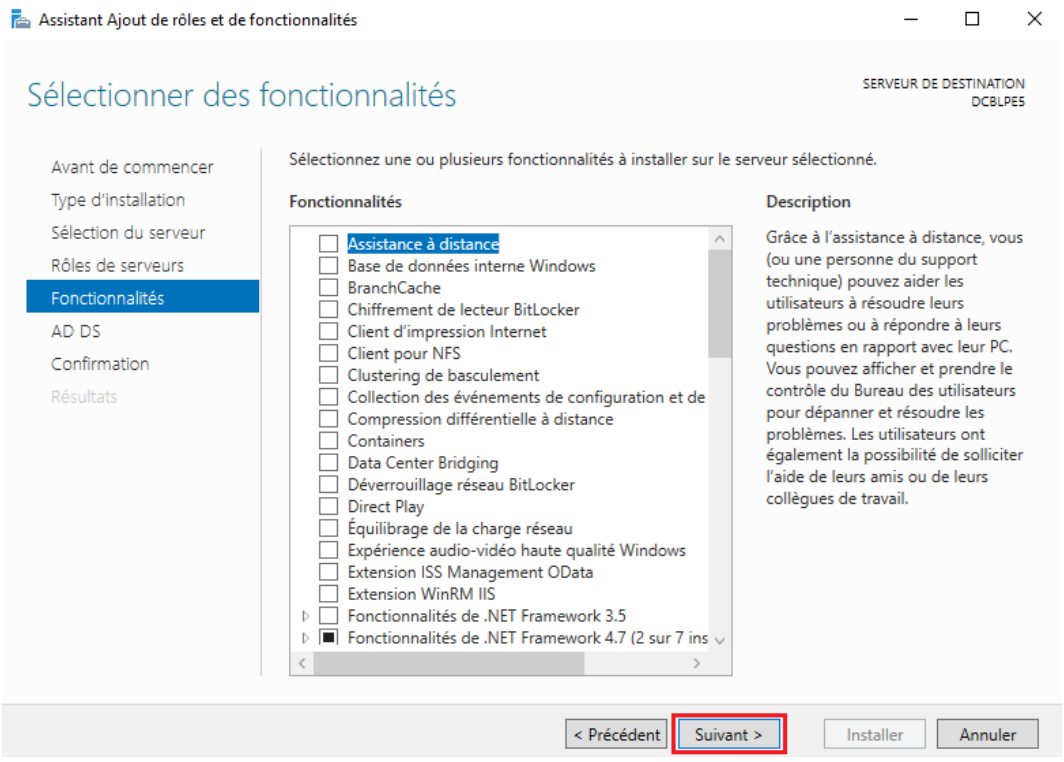
On sélectionne le serveur de destination et on clique sur suivant :



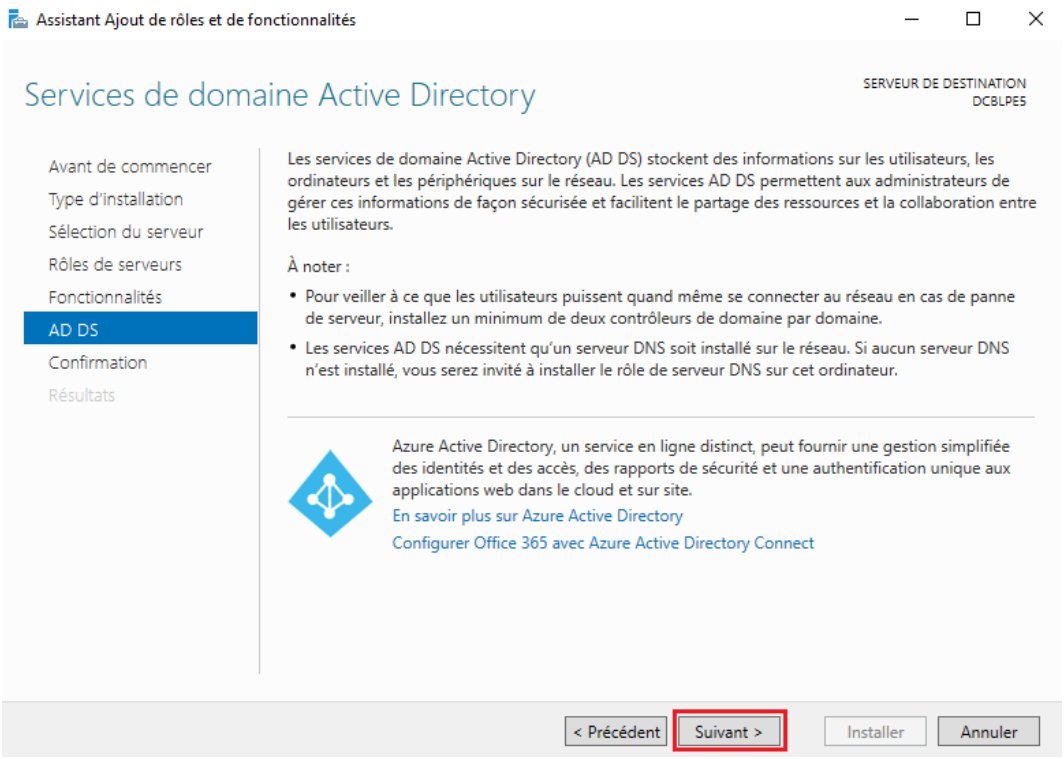
On sélectionne ensuite le ou les rôles qu'on souhaite installer, en l'occurrence, on souhaite installer toute la pile AD DS et on clique sur suivant :



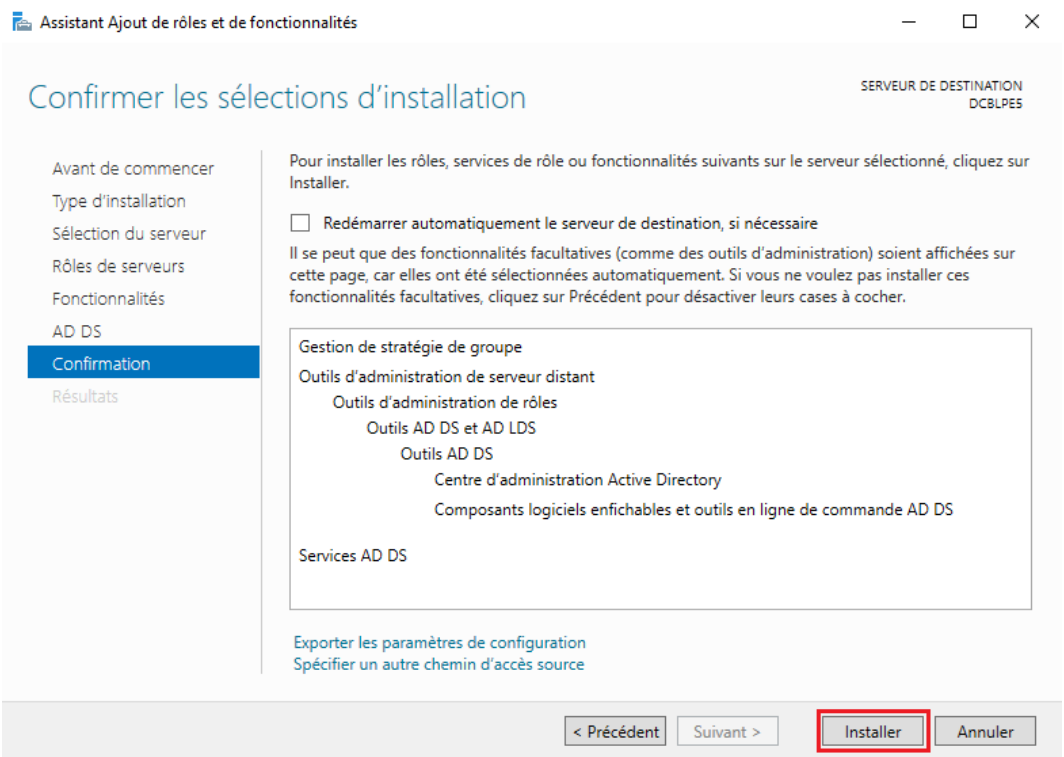
Vous pouvez sélectionner les fonctionnalités de votre choix, dans notre situation aucune fonctionnalités a été ajouter, et cliquez sur suivant :



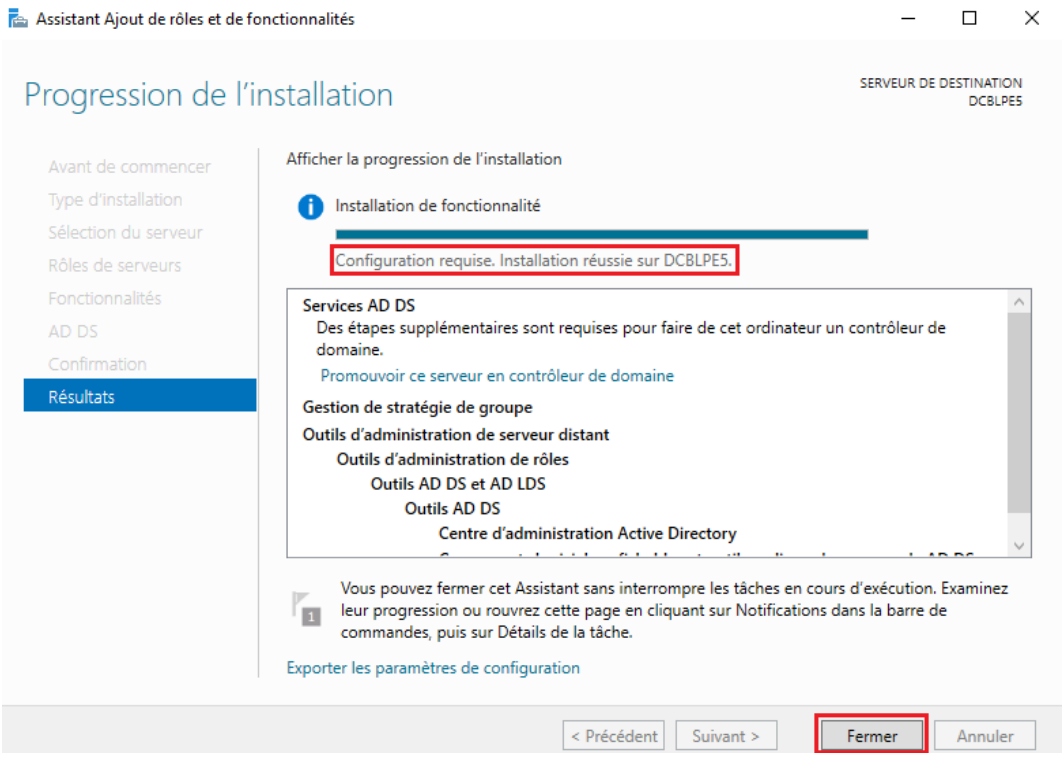
Prenez connaissance de la description du rôle AD DS et cliquez sur suivant :



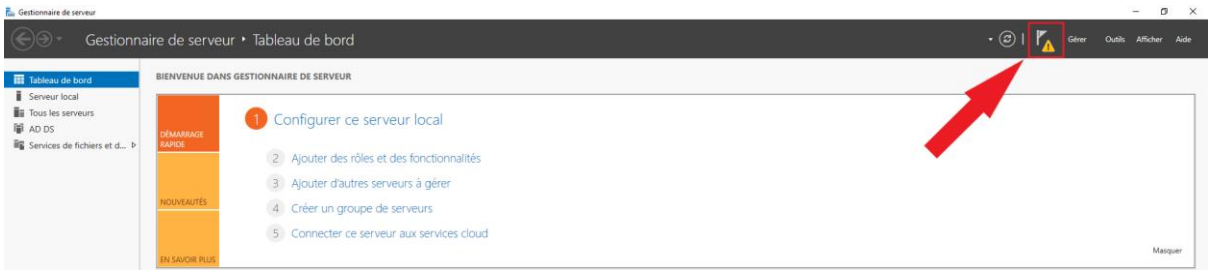
Assurez-vous d'avoir bien sélectionné les éléments à installer et cliquez sur Installer :



Lorsque l'installation est terminée et réussie, vous pouvez fermer l'assistant d'ajout de rôles et de fonctionnalités en cliquant sur fermer, car comme vous pourrez le voir, une configuration est requise :

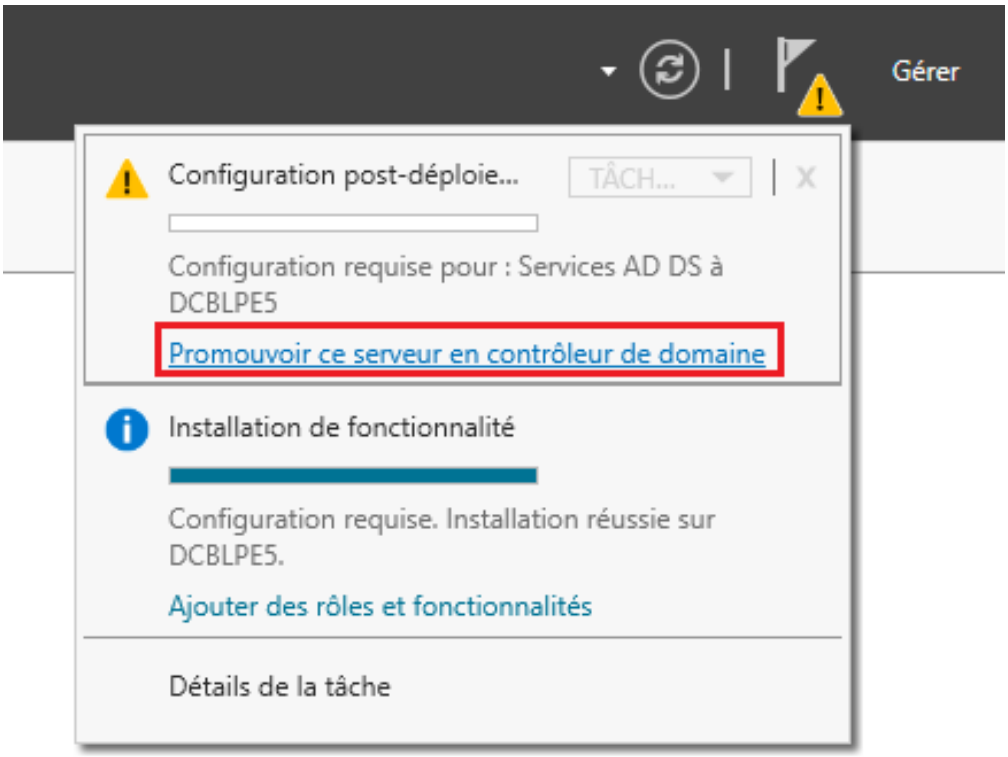


En haut à droite, vous remarquerez un triangle jaune à côté d'un drapeau, cliquez sur ce dernier :

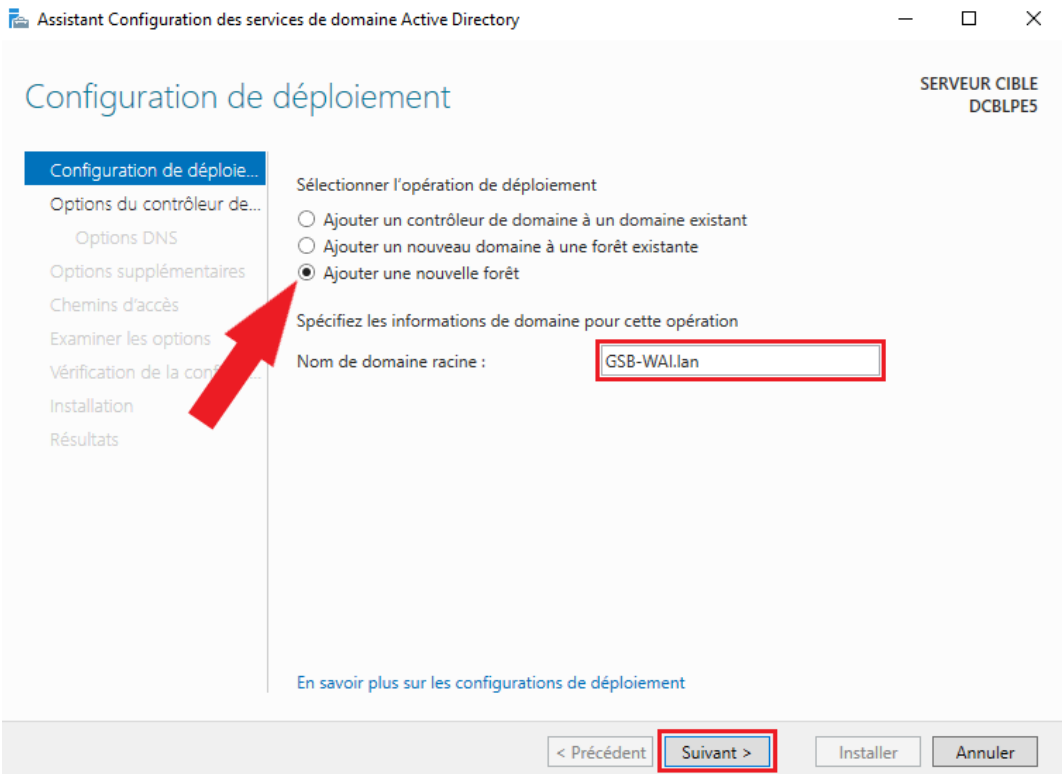


Promotion du serveur en DC

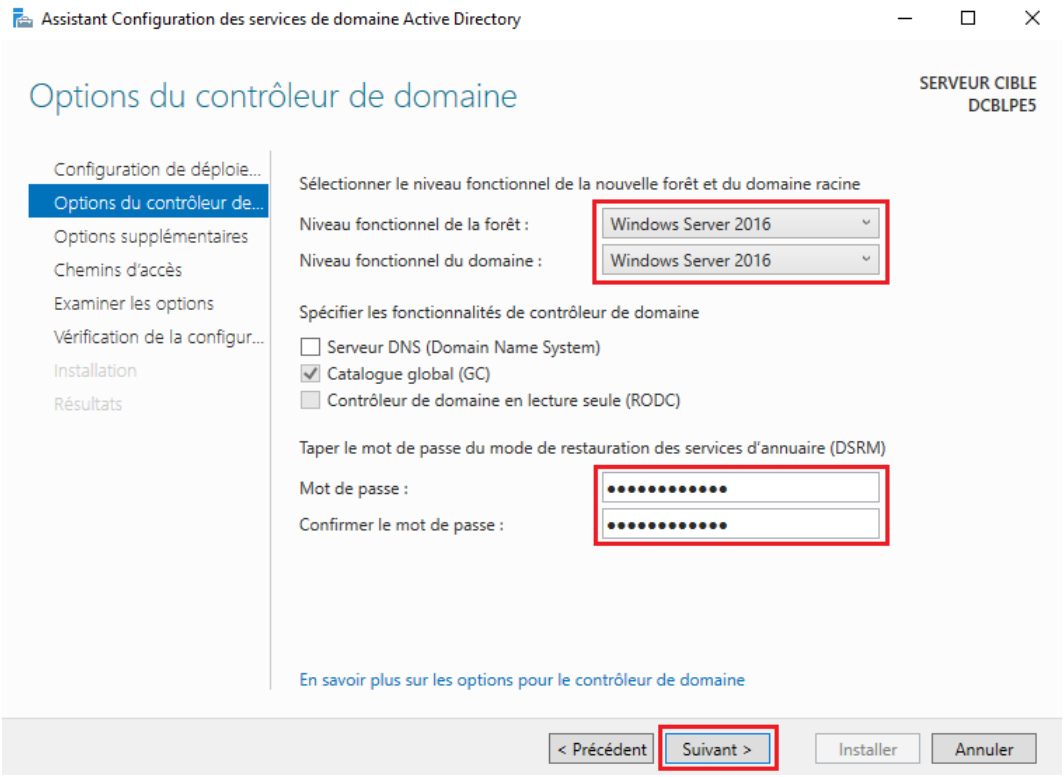
Vous pourrez voir une configuration post déploiement qui consiste à promouvoir votre serveur en contrôleur de domaine, cliquez sur promouvoir ce serveur en contrôleur de domaine :



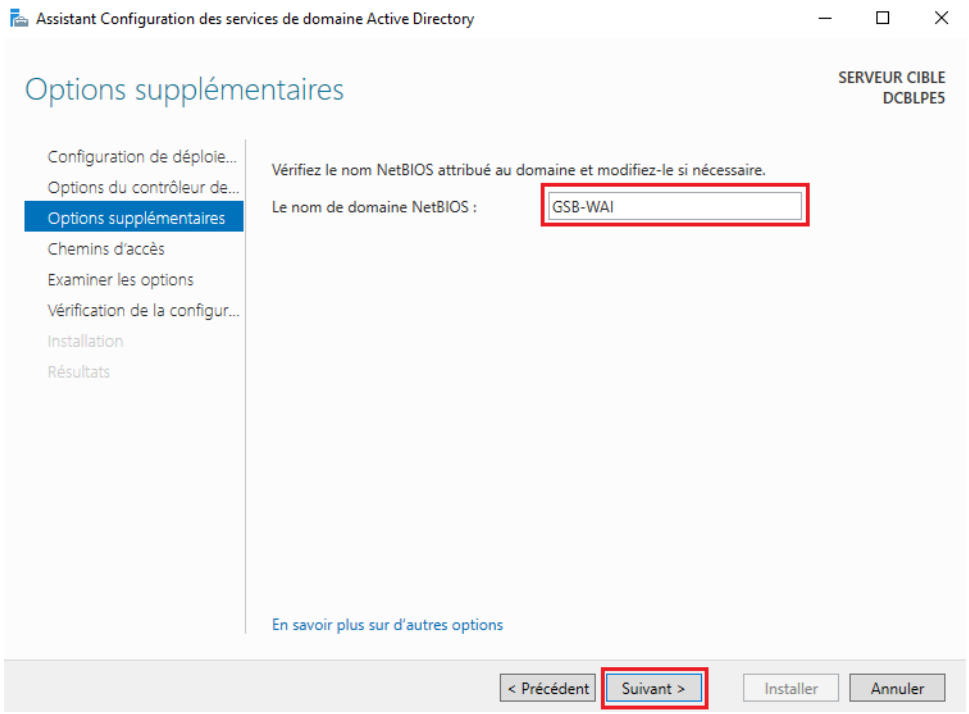
L'assistant de configuration des services de domaine Active Directory se lance et vous demande de choisir une configuration de déploiement. Dans notre situation, n'ayant pas de forêt ou de domaine déjà existant, nous allons créer une nouvelle forêt :



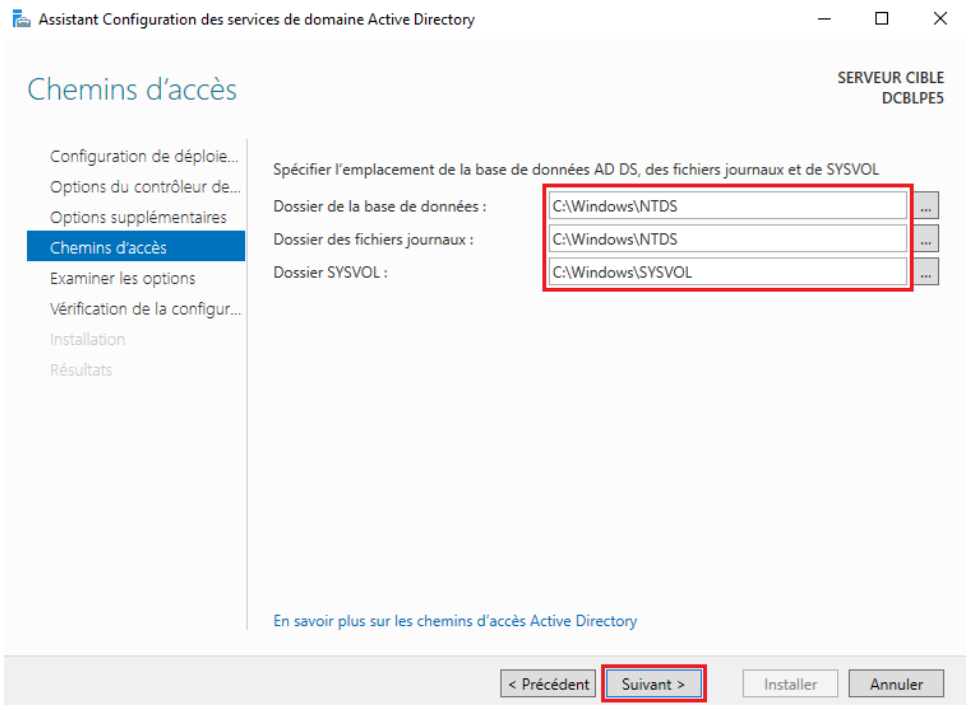
Il vous faudra ensuite choisir les différentes options du contrôleur de domaine :



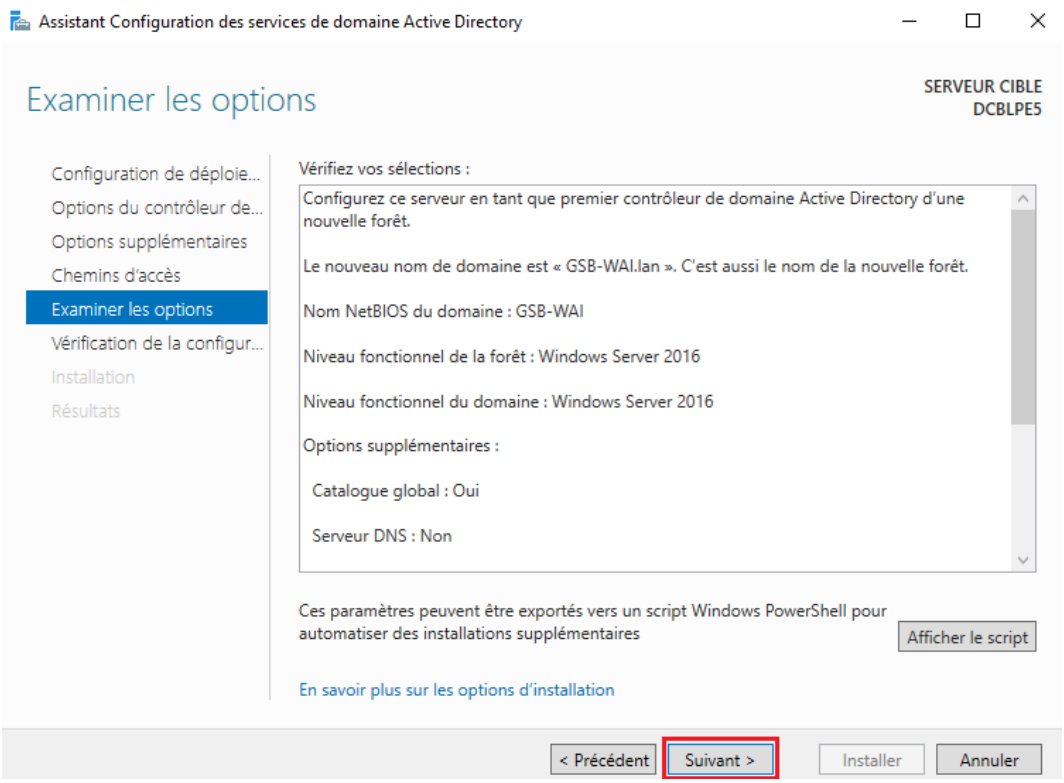
Vérifiez votre nom de domaine :



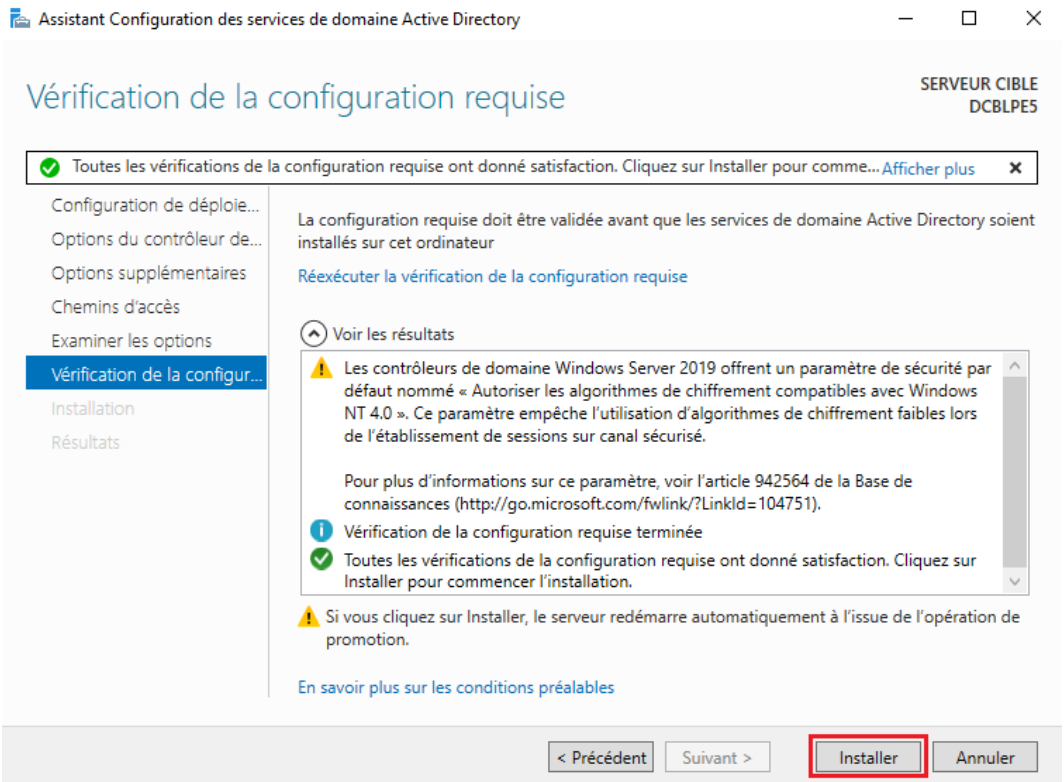
Vous devrez spécifier l'emplacement de la base de données AD DS, des fichiers journaux et de SYSVOL. Dans notre situation, nous laisserons les chemins par défaut :



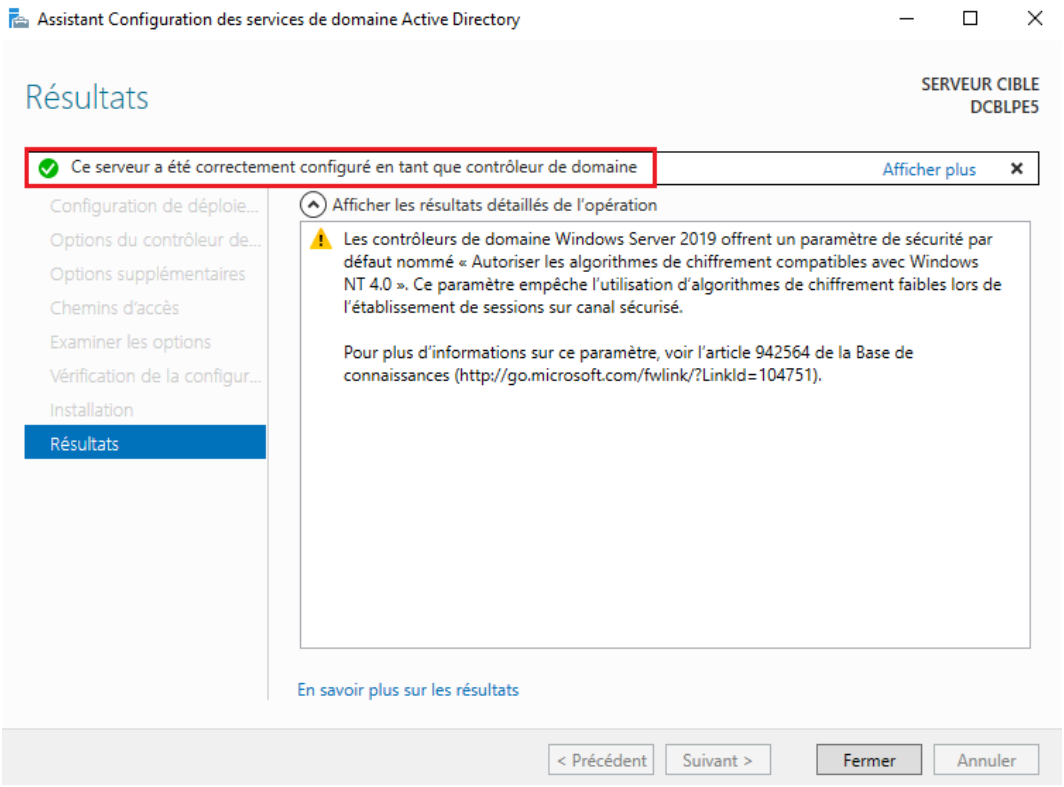
Vous devrez confirmer vos sélections :



L'assistant de configuration des services de domaine Active Directory vérifie ensuite la conformité de la configuration et vous pourrez installer la configuration :



Le serveur va redémarrer automatiquement afin d'appliquer les modifications :



Conclusion

En résumé, la mise en place du contrôleur de domaine Active Directory sous Windows Server a été réalisée avec succès. Cette configuration permet une gestion centralisée des utilisateurs, des groupes et des ressources, ainsi que l'implémentation d'options avancées telles que l'authentification unique et la politique de sécurité centralisée. Active Directory renforce la sécurité des environnement Windows, simplifie leurs administrations et les rends plus adaptable à nos besoins informatiques en constante évolution.

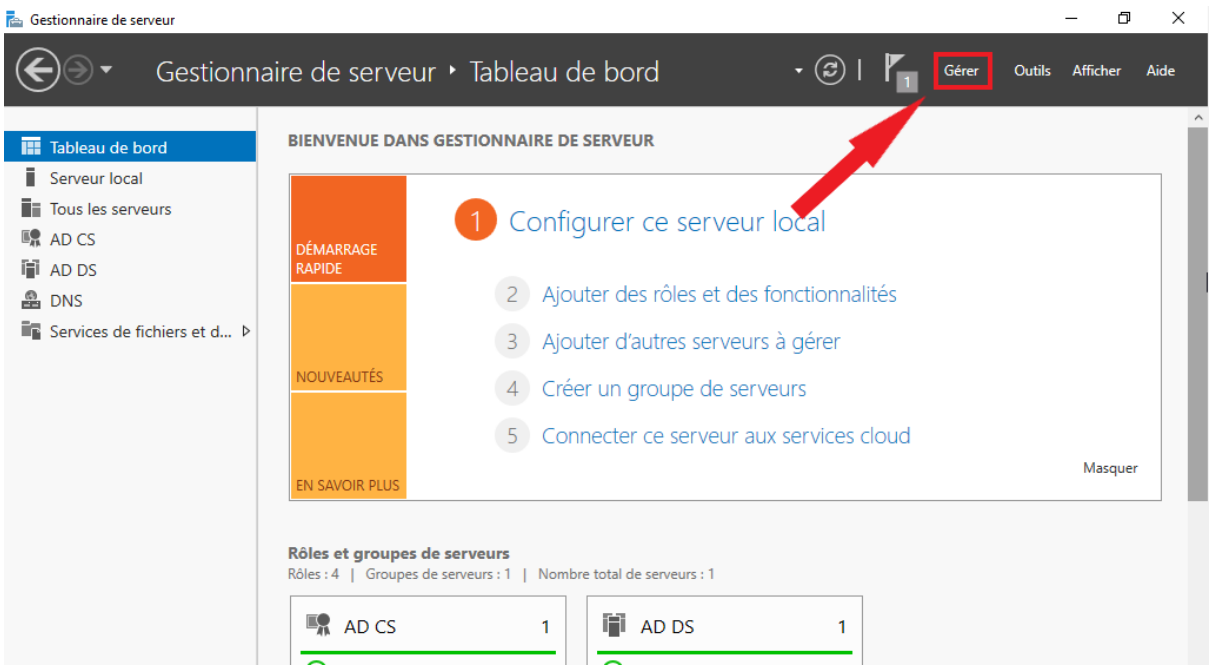
Mode Opératoire NPS

Dans ce mode opératoire, nous aborderons comment mettre en place et configurer un serveur Network Policy Server sous Windows Server 2016.

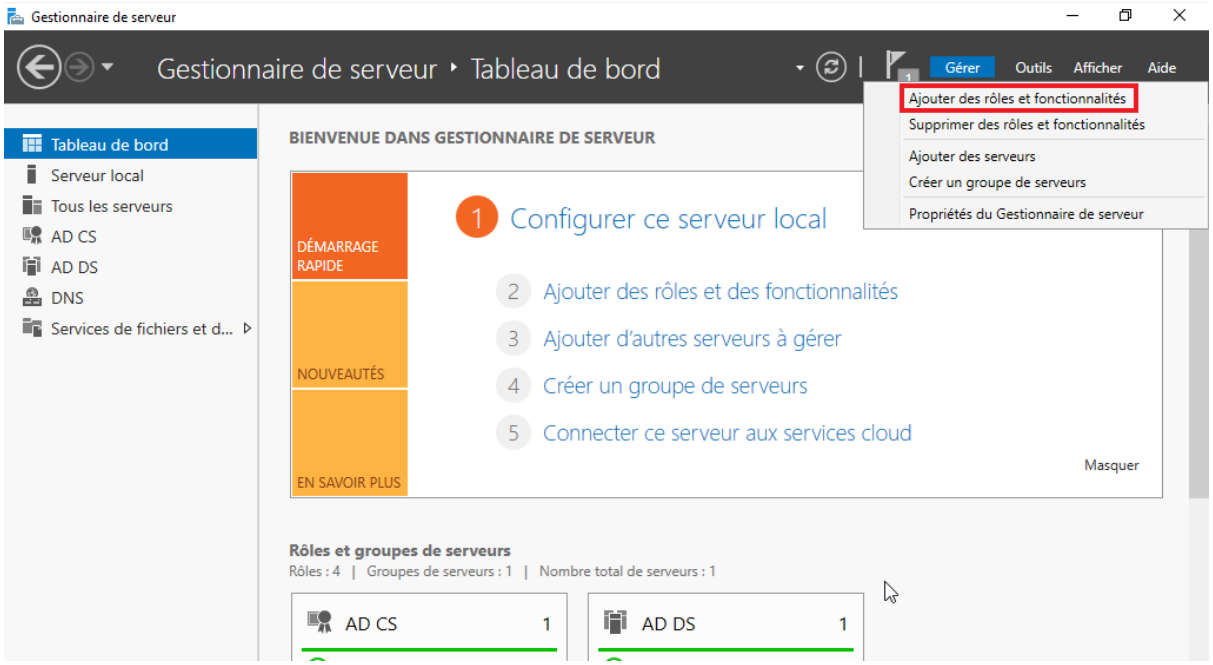
Le rôle NPS permet de gérer l'authentification et l'autorisation des utilisateurs du réseau. Il sert à établir des politiques qui contrôlent l'accès au réseau et à centraliser la configuration de la sécurité. NPS utilise souvent RADIUS pour authentifier les connexions et peut également enregistrer les activités pour la surveillance et le rapport. En résumé, NPS aide à sécuriser et à administrer efficacement les accès réseau dans un environnement d'entreprise.

Installation Rôle NPS

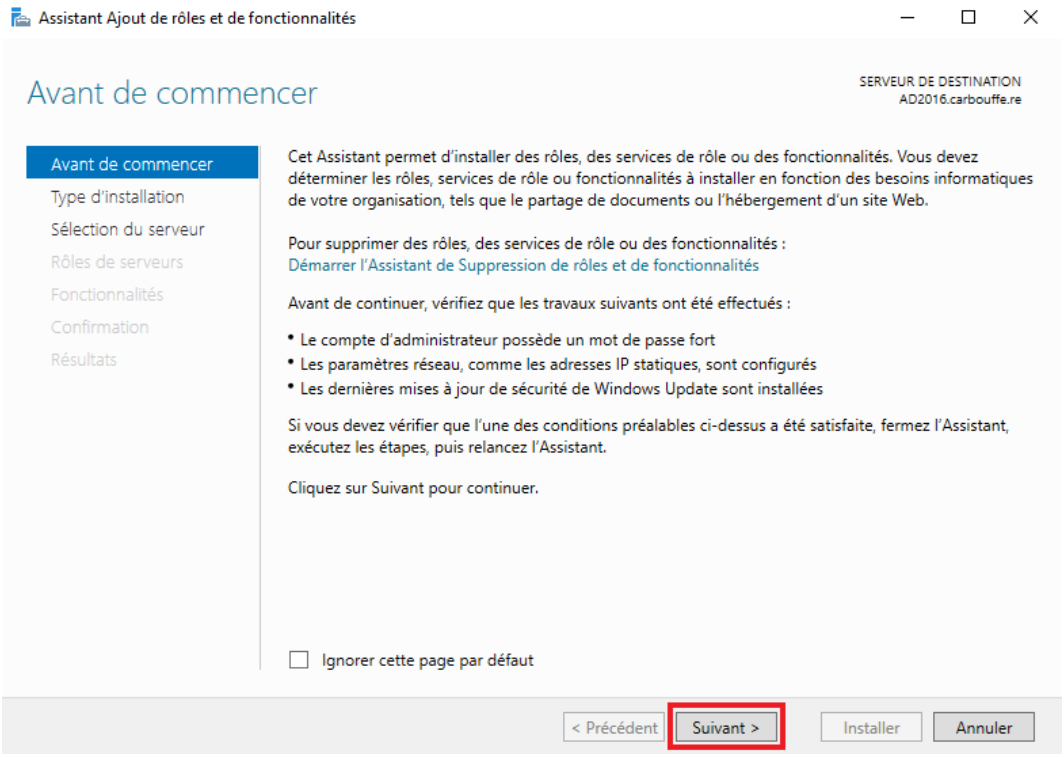
Sur le serveur WS2016, dans le gestionnaire de serveur, nous allons commencer par installer le **Rôle** NPS. Pour cela on va dans « Gérer » en haut à droite de la fenêtre :



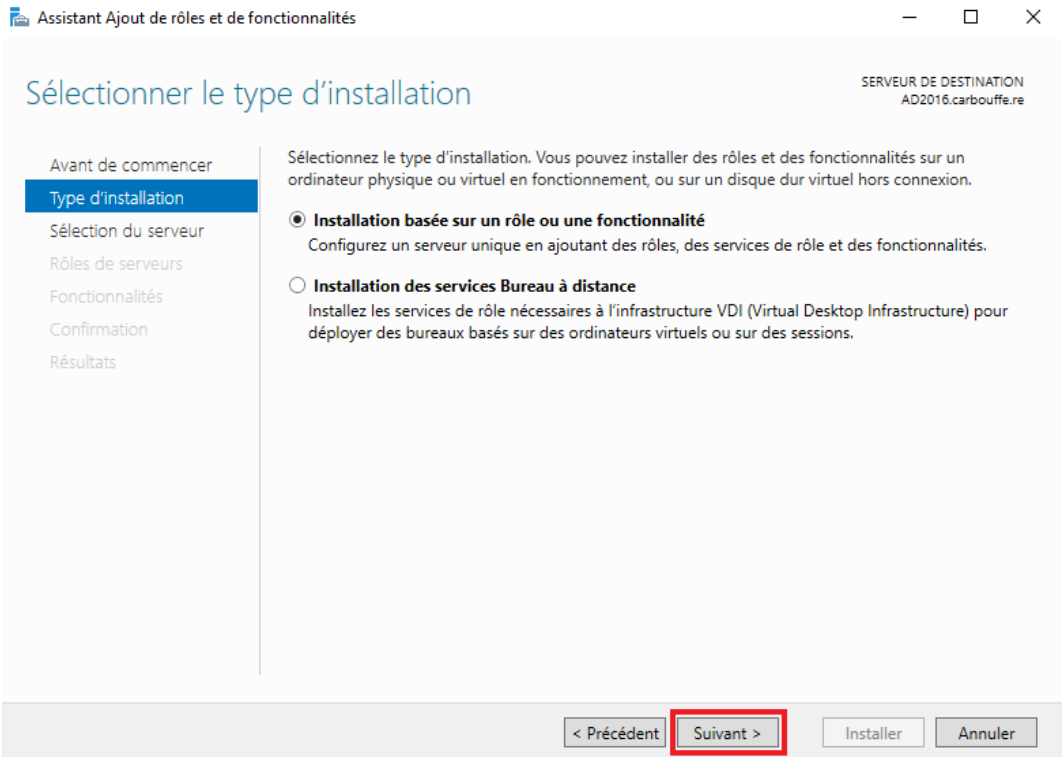
Puis cliquer sur « Ajouter des rôles et des fonctionnalités » :



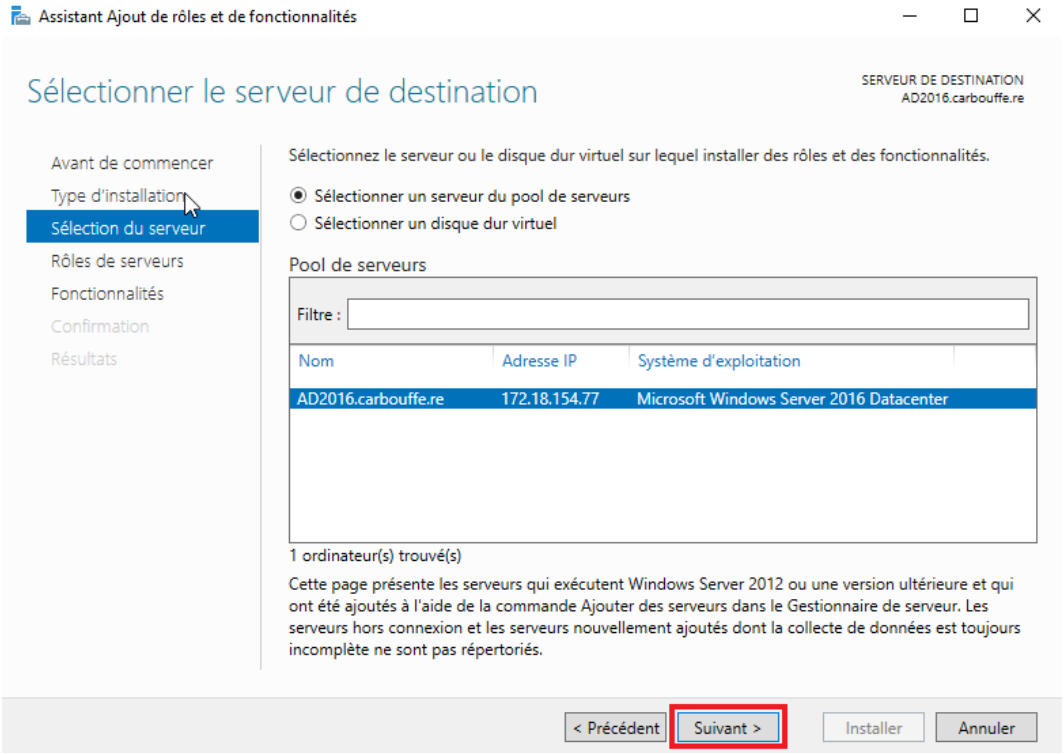
L'« Assistant Ajout de rôles et de fonctionnalités » s'ouvre. Lisez attentivement les informations qui vous sont présentés et cliquez sur suivant :



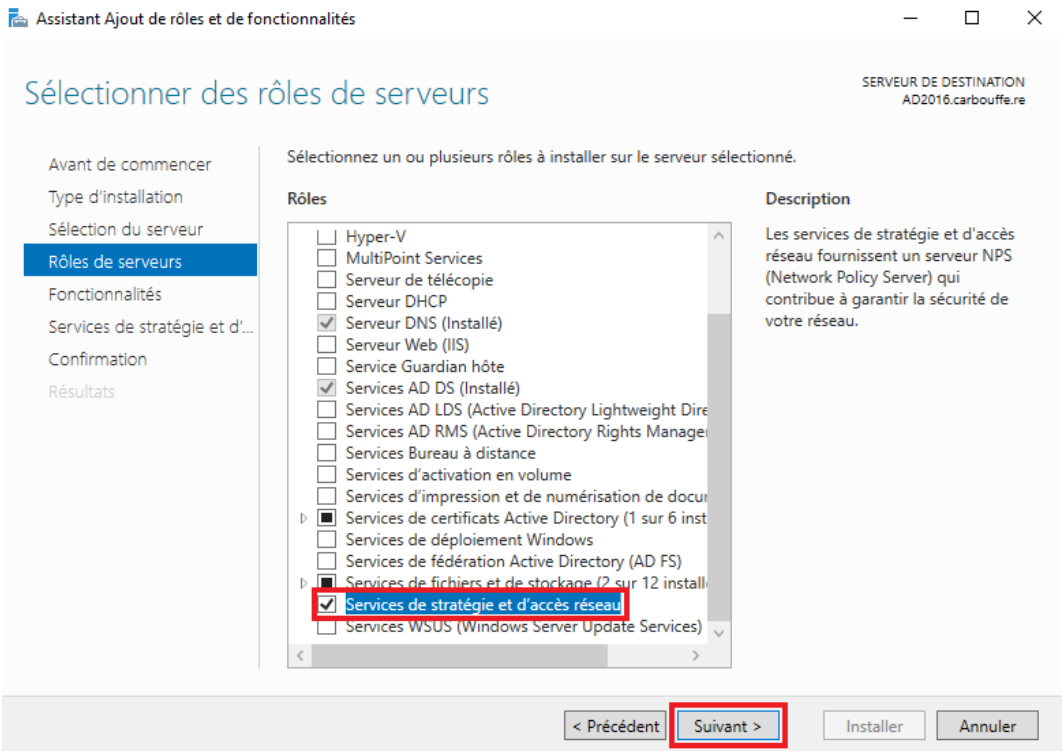
Sélectionner le type d'installation de votre choix, dans cette situation on choisira une Installation basée sur un rôle ou une fonctionnalité. Ensuite cliquez sur suivant :



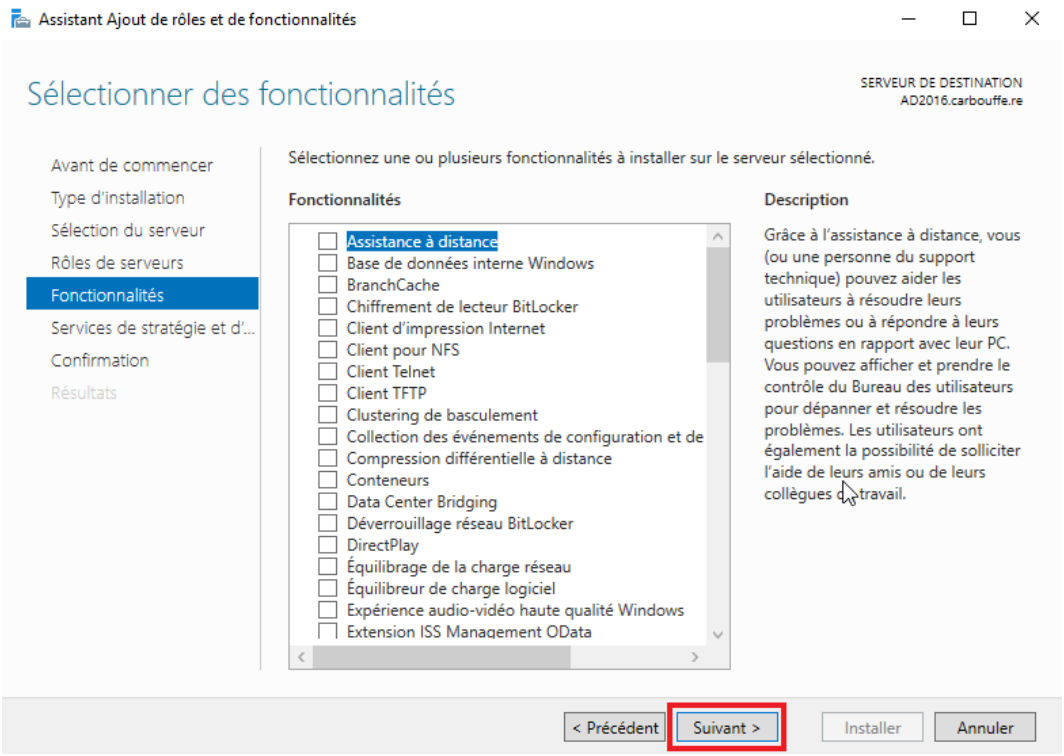
On sélectionne le serveur de destination et on clique sur suivant :



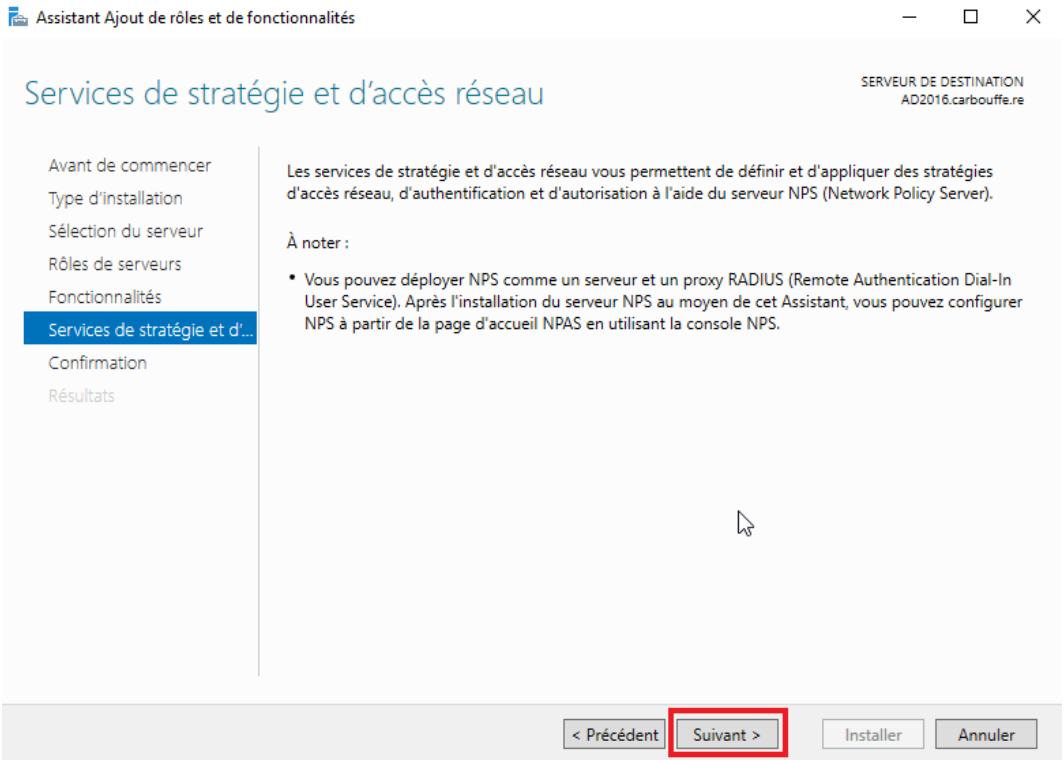
On sélectionne ensuite le ou les rôles qu'on souhaite installé, en l'occurrence on souhaite installé toute la pile NPS :



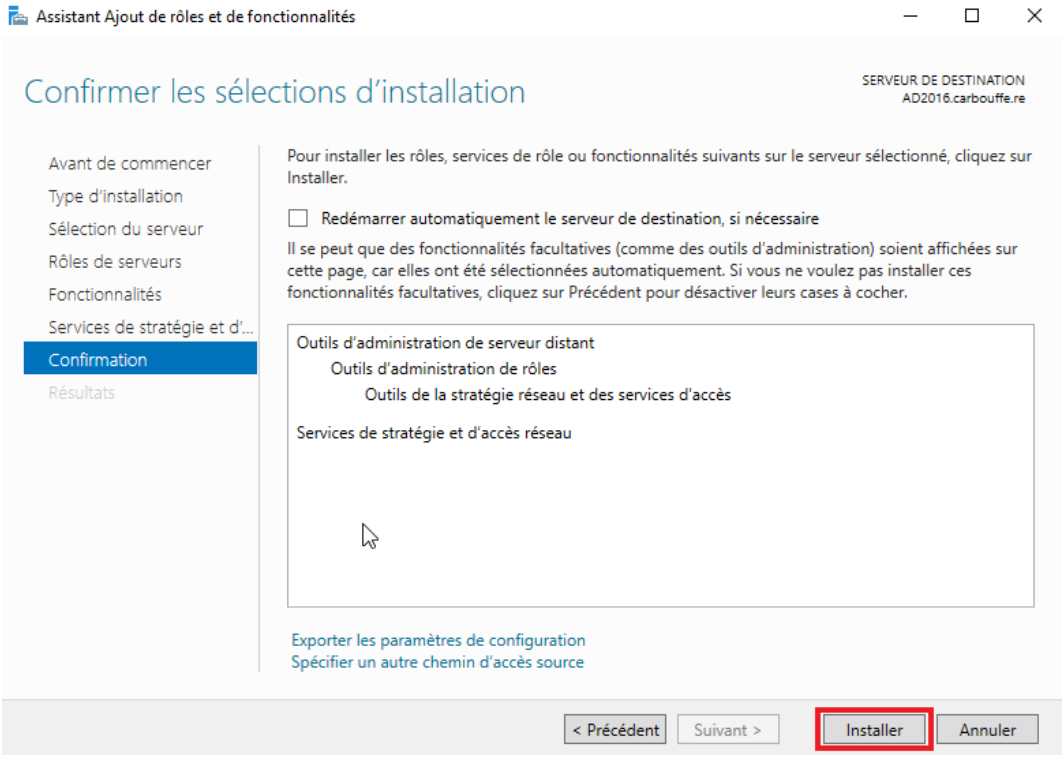
Vous pouvez sélectionner les fonctionnalités de votre choix et cliquez sur suivant :

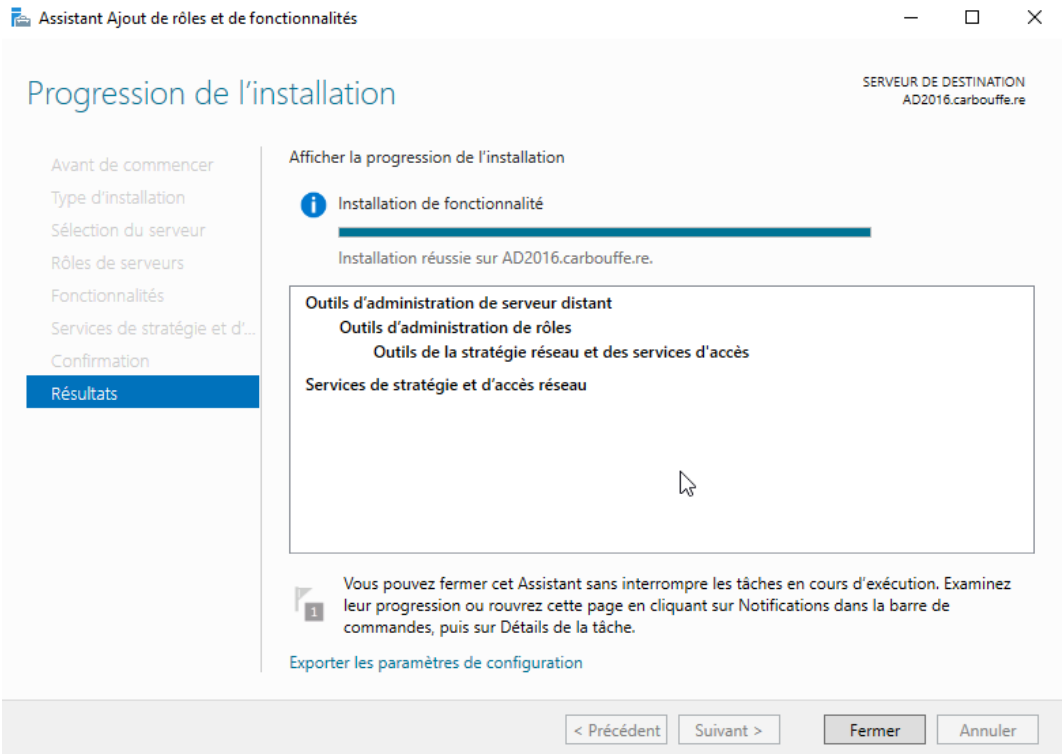


Prenez connaissance de la description du rôle NPS et cliquez sur suivant :



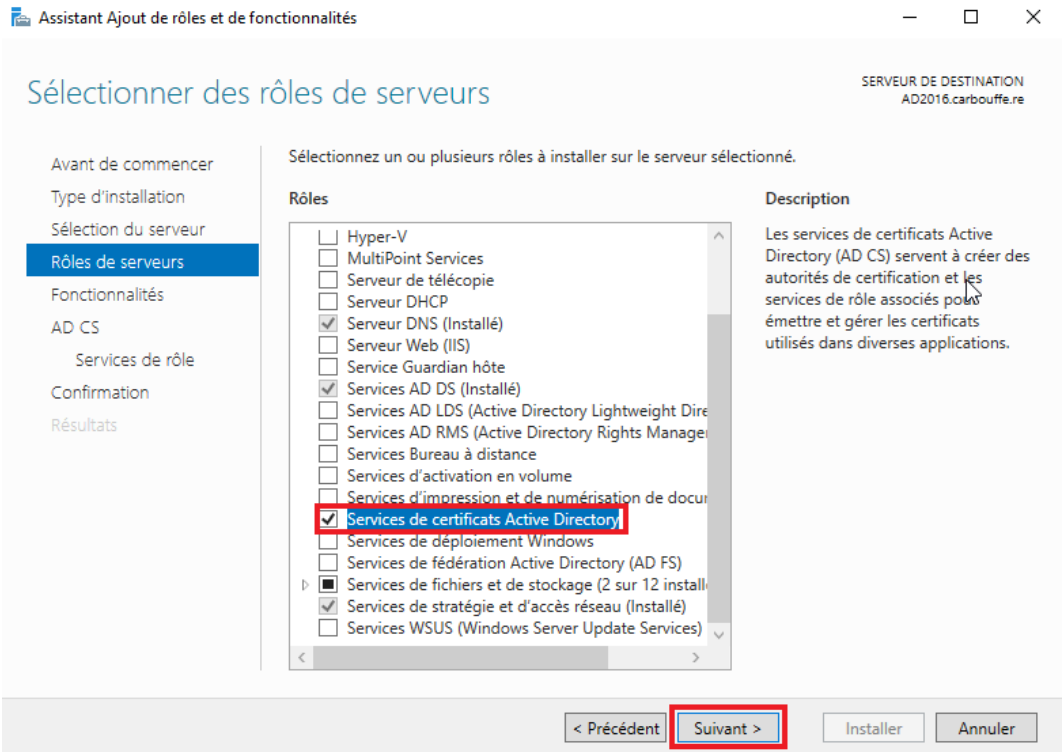
Assurez-vous d'avoir bien sélectionnez les éléments à installer et cliquez sur Installer :



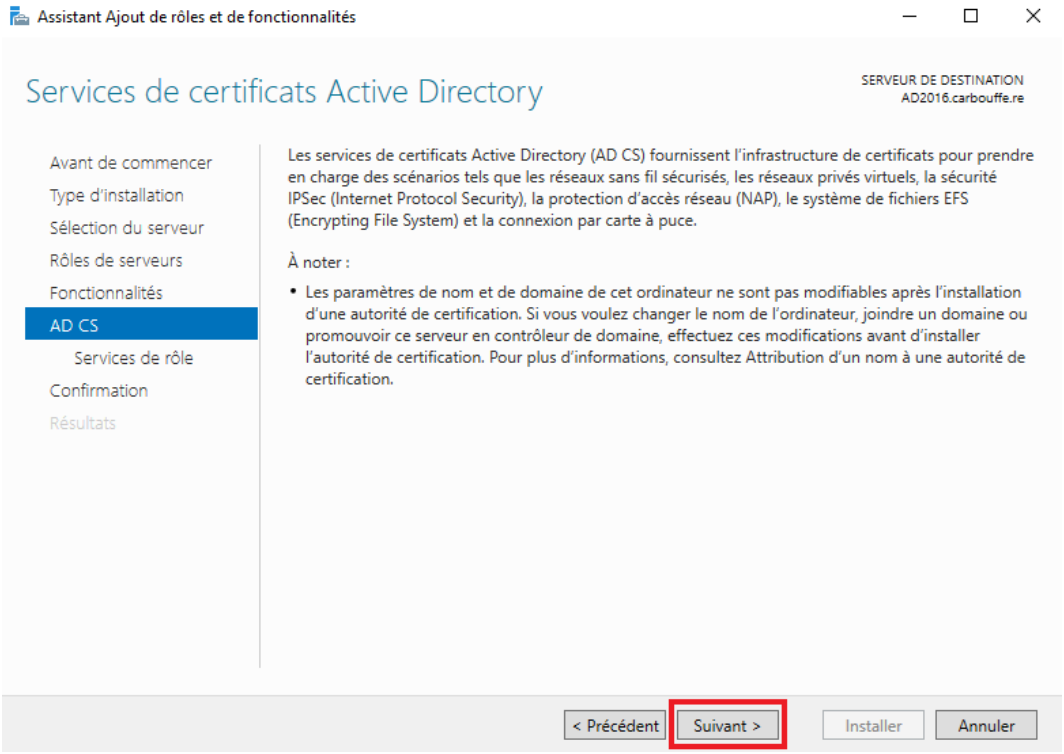


Installation du Rôle de Services de Certificats Active Directory

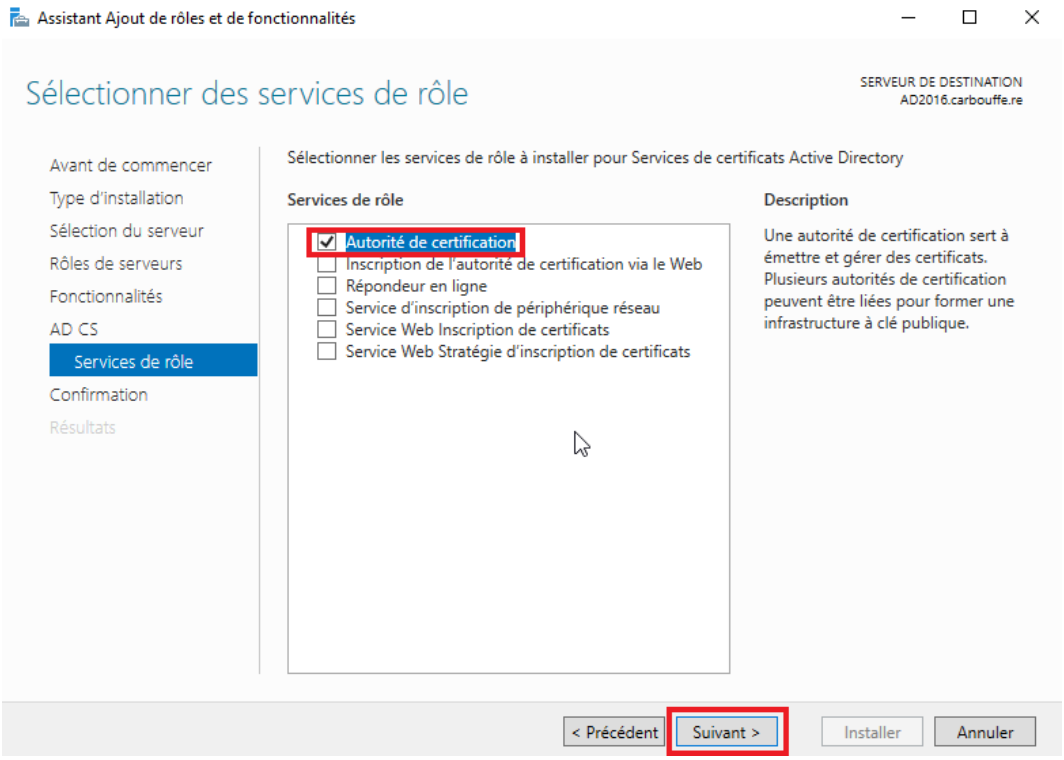
L'installation du rôle NPS étant terminée, passons à l'installation du Rôle Services de certificats Active Directory. De la même manière que pour NPS, on va ajouter ce rôle :



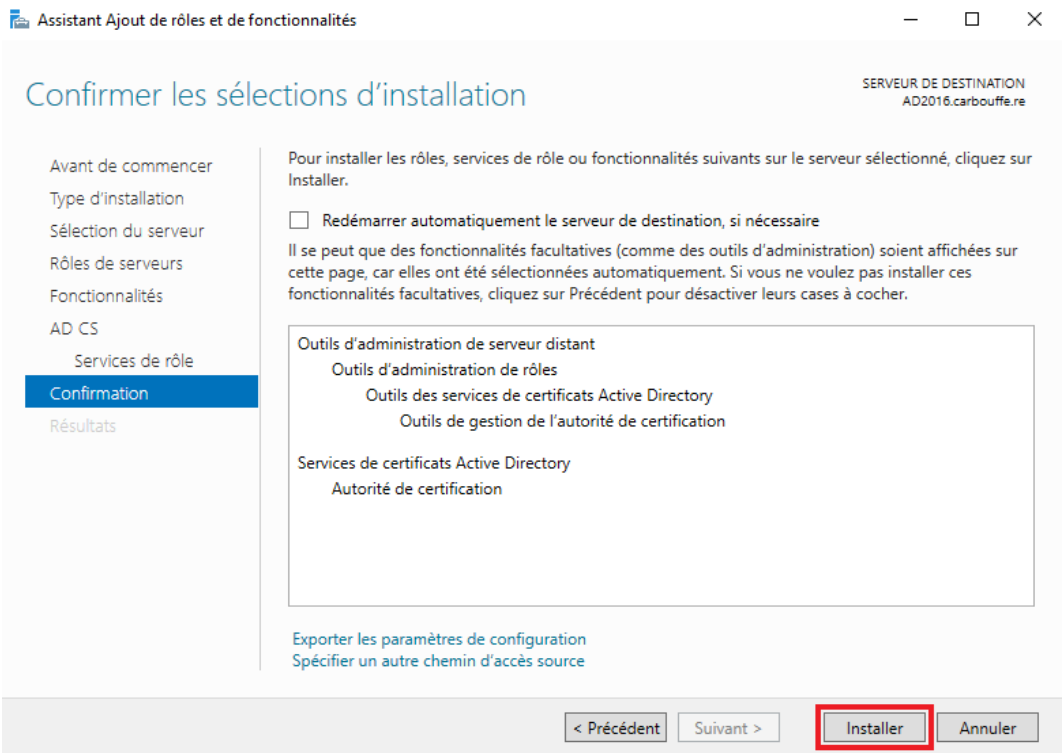
Vous pouvez sélectionner les fonctionnalités de votre choix, prendre connaissance de la description du rôle et cliquez sur suivant :

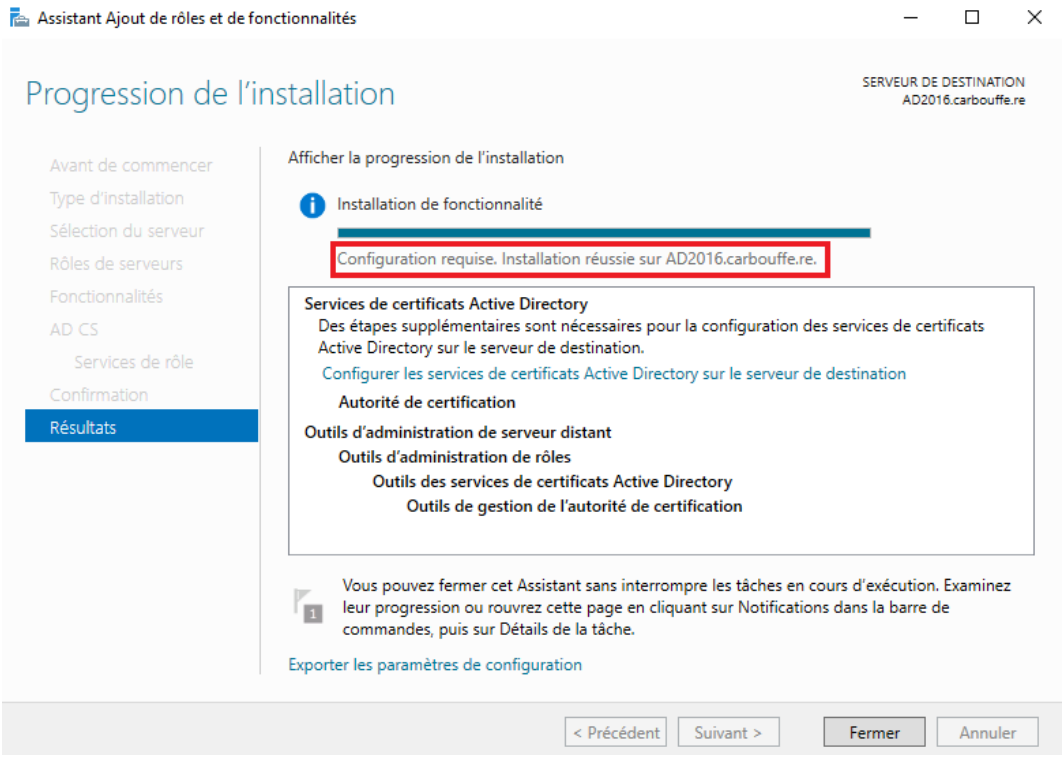


On sélectionne les services de rôles de notre choix, en l'occurrence « Autorité de certification » :

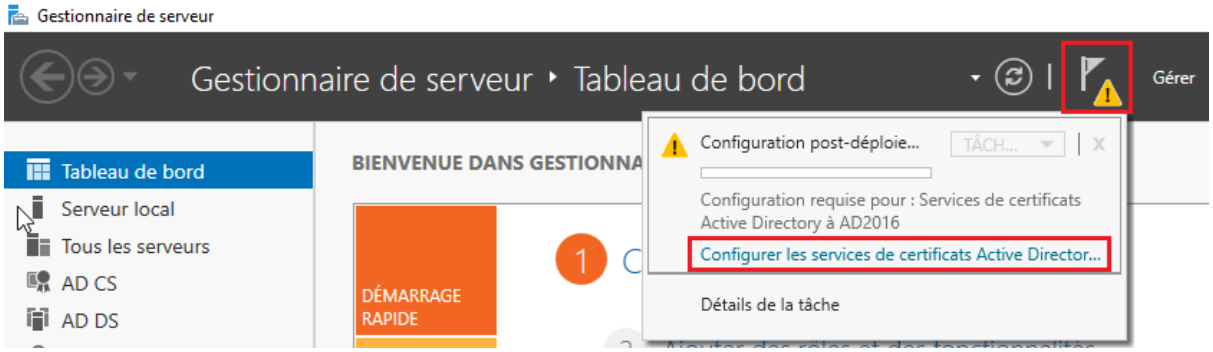


Assurez-vous d'avoir bien sélectionnez les éléments à installer et cliquez sur Installer :

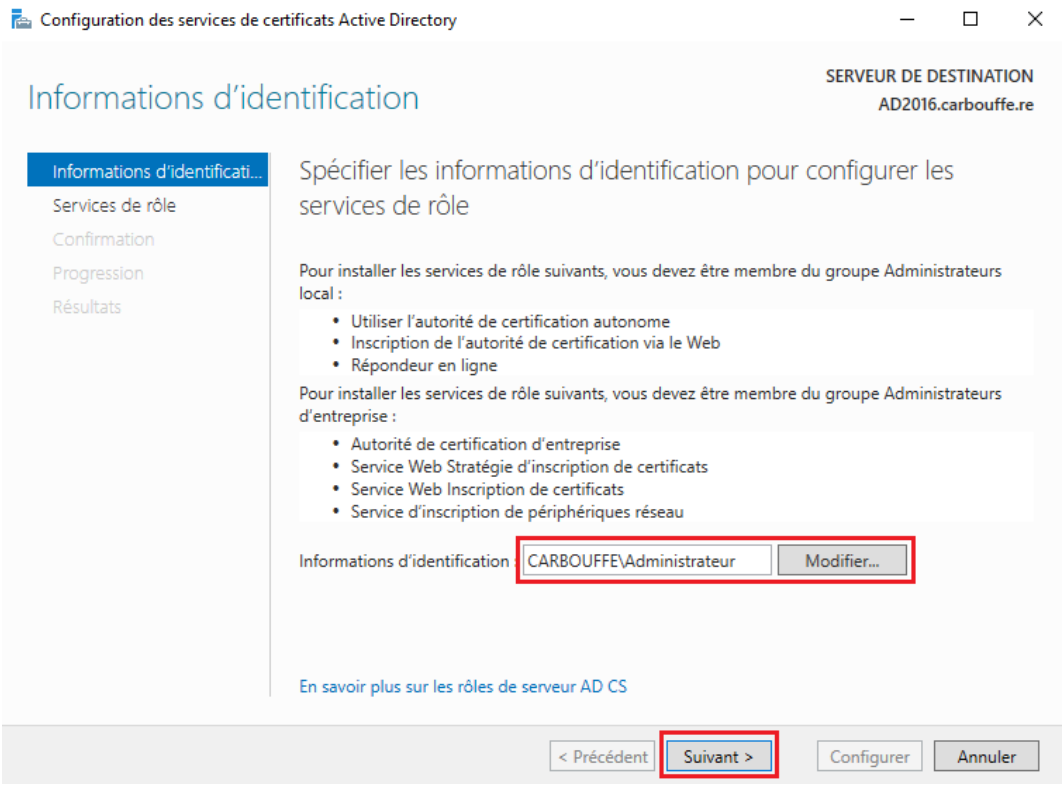




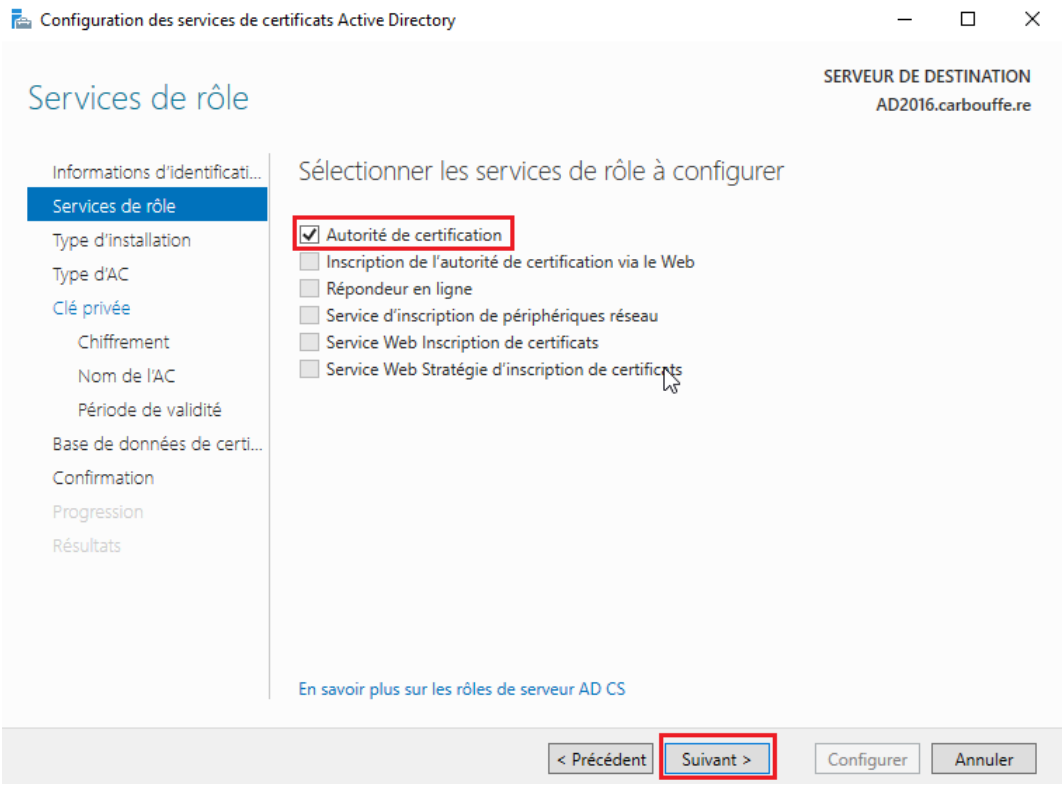
Comme vous pouvez le voir, une configuration supplémentaire est requise. Fermer l'assistant d'ajout de rôles et de fonctionnalités et cliquer sur le drapeau avec le triangle jaune en haut à droite du gestionnaire de serveur puis sur Configurer les services de certificats Active Directory :



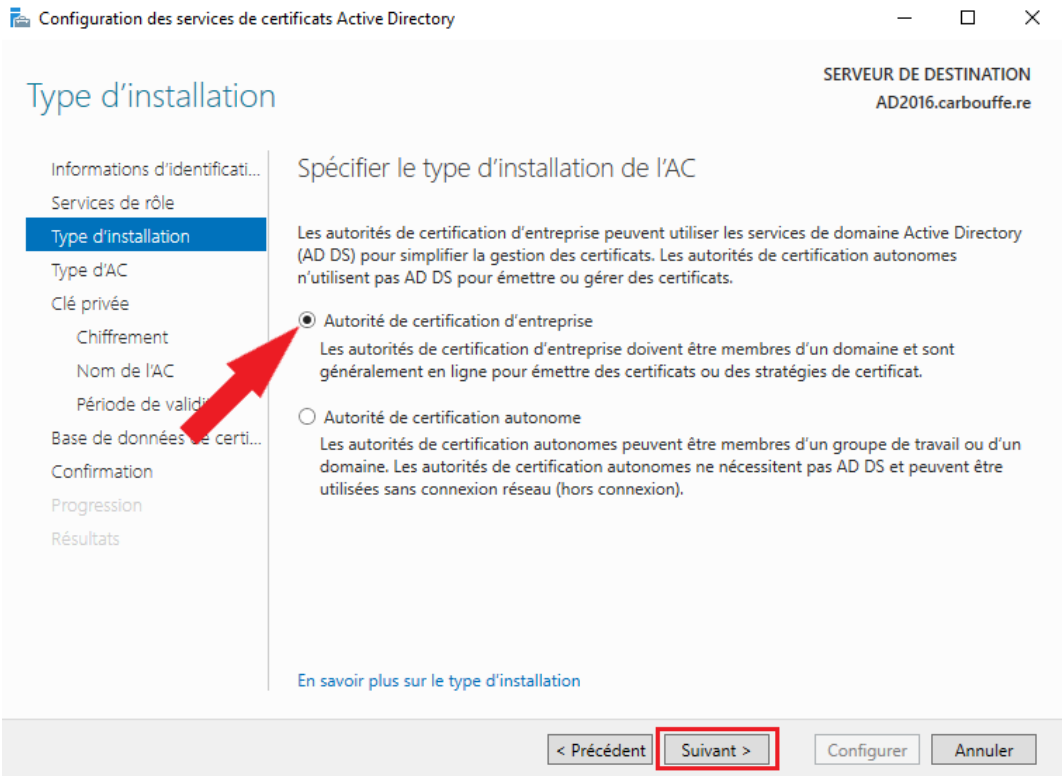
Lisez attentivement les informations présentes sur la fenêtre de configuration des services de certificats Active Directory sans oublier de renseigner les informations nécessaires :



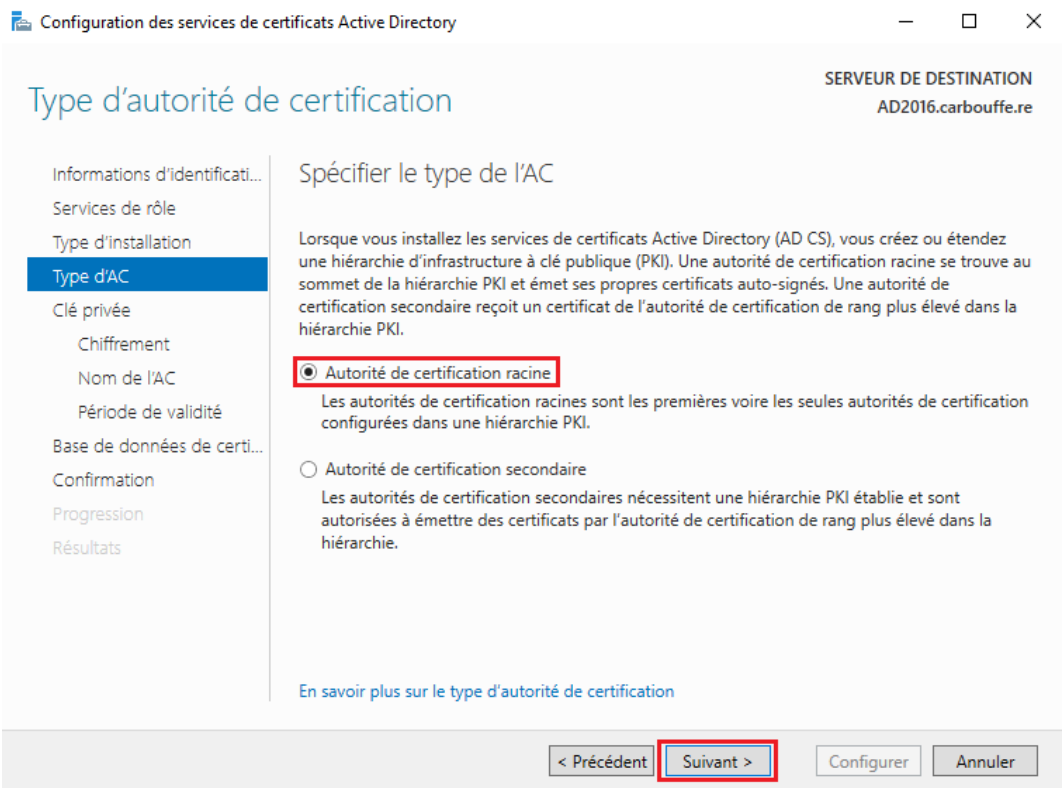
Sélectionnez les services de rôle à configurer, en l'occurrence l'autorité de certification :



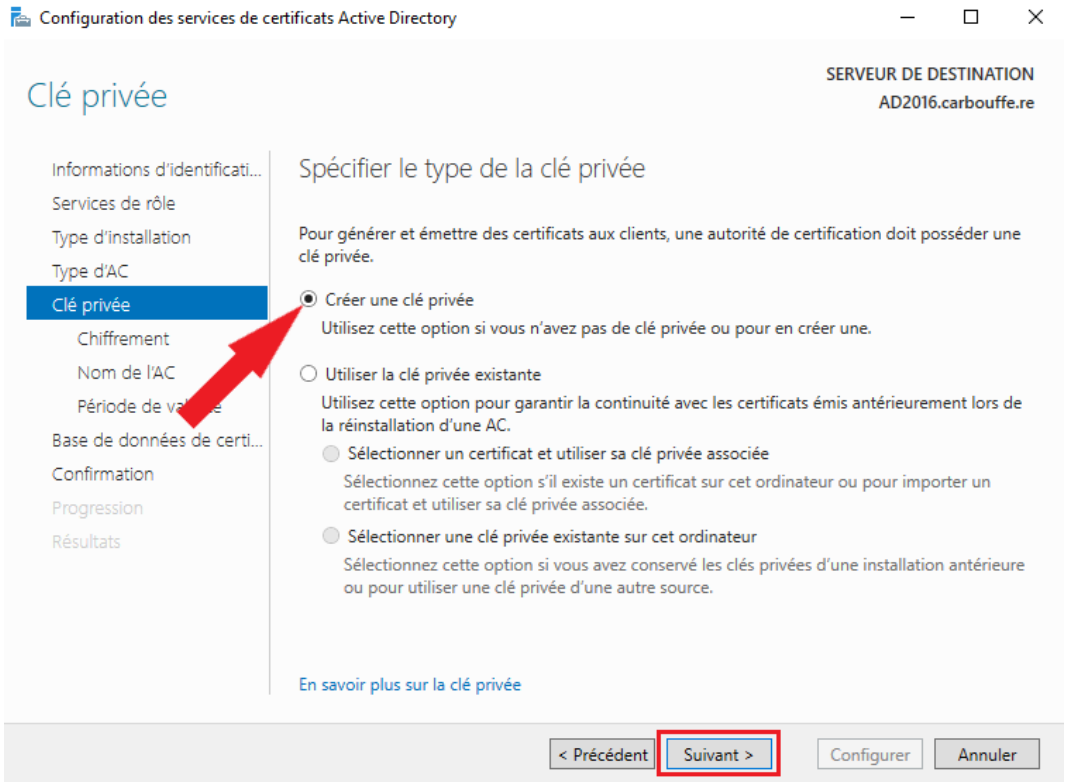
On sélectionne le type d'installation souhaité pour l'AC :



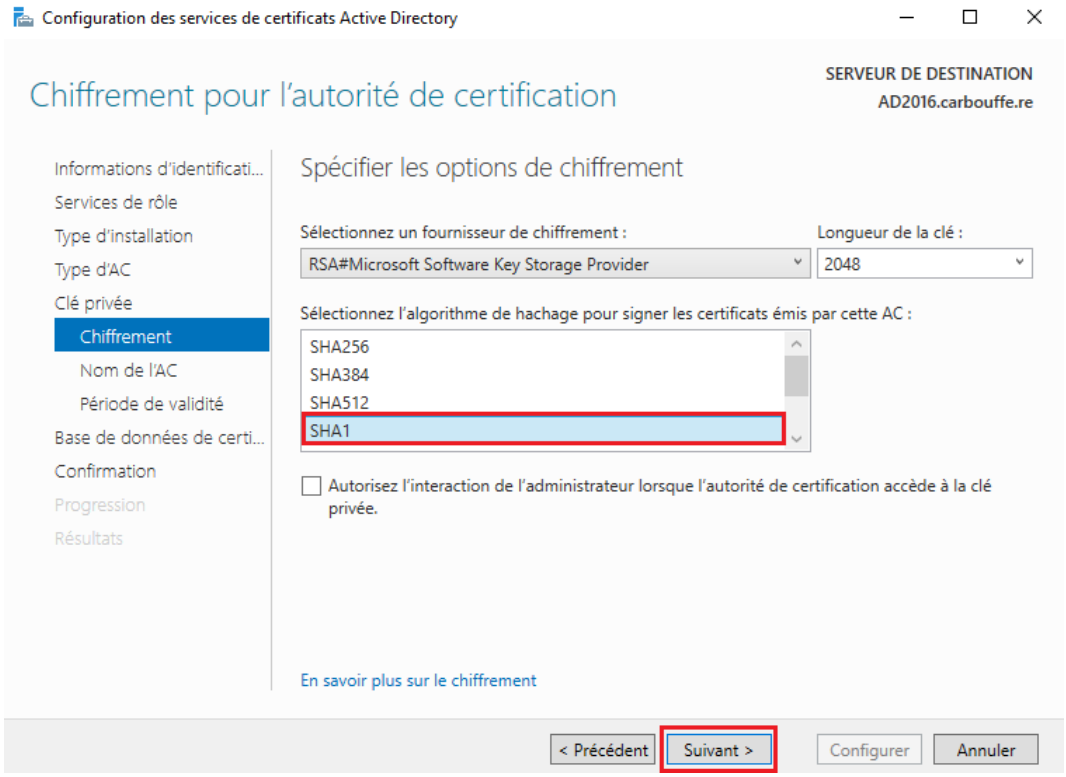
On sélectionne le type d'AC :



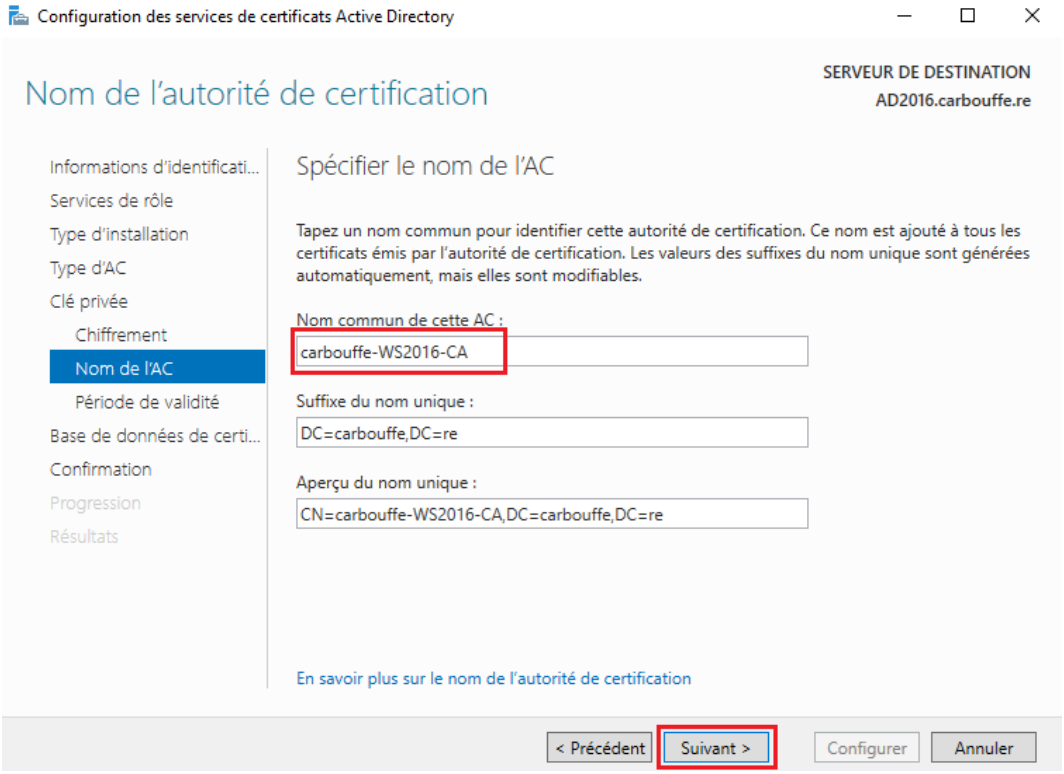
On crée une nouvelle clé privée :



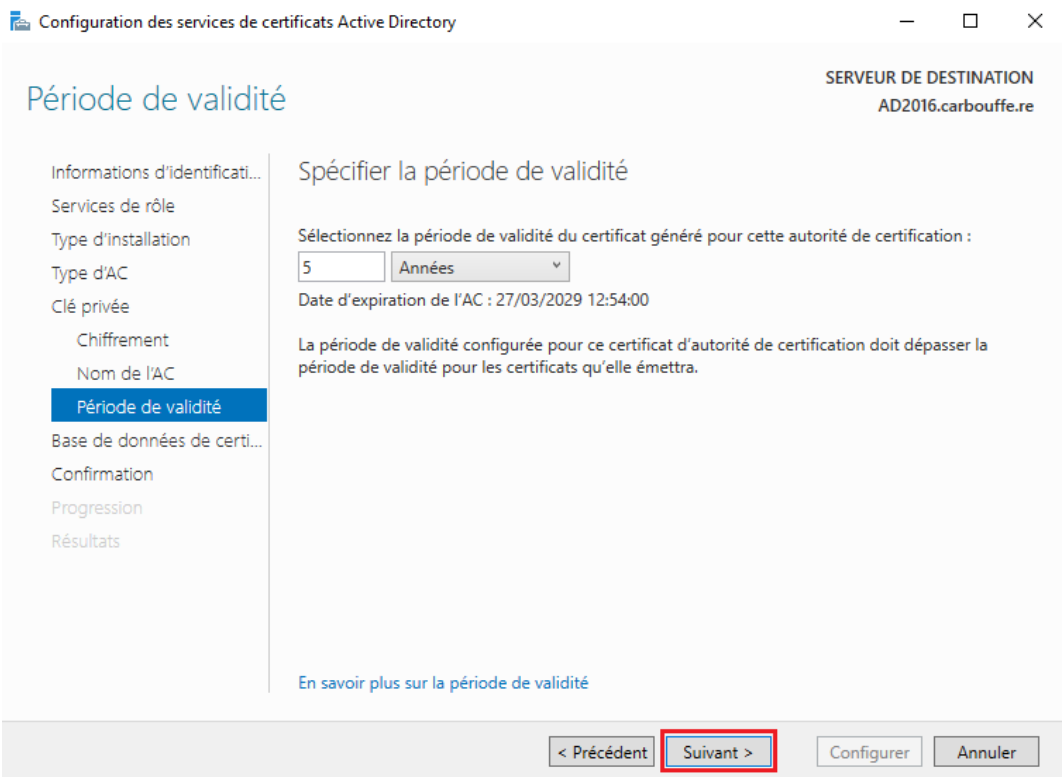
On sélectionne les options de chiffrement de notre choix :



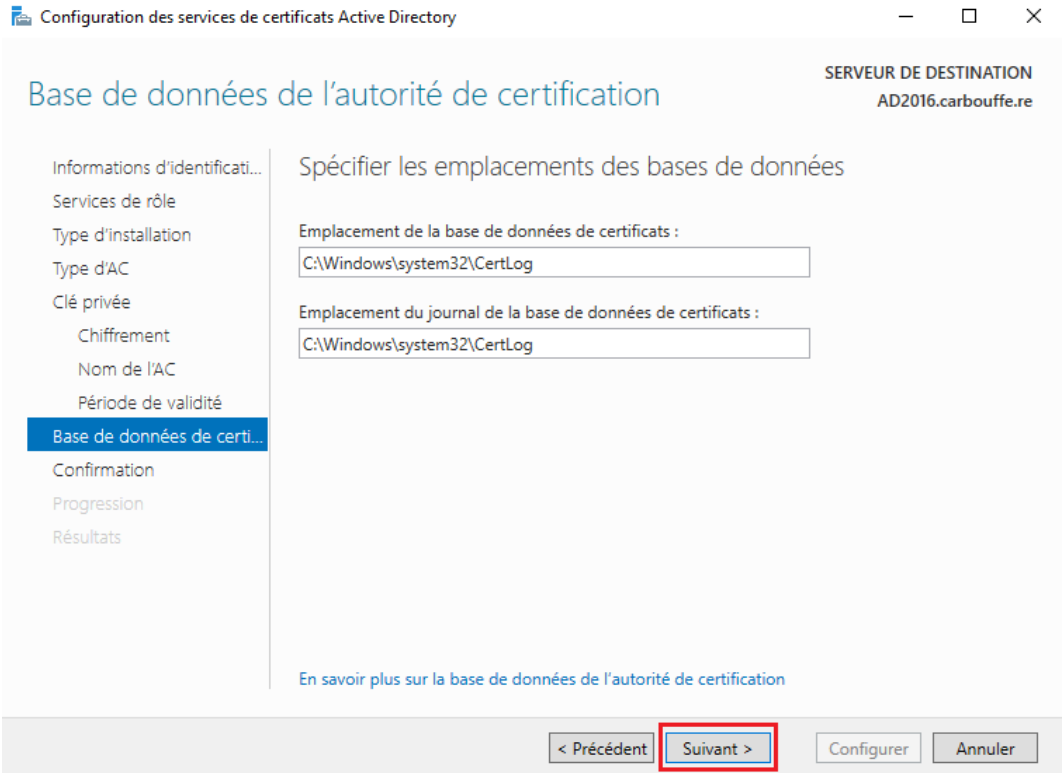
Spécifier le nom de l'AC :



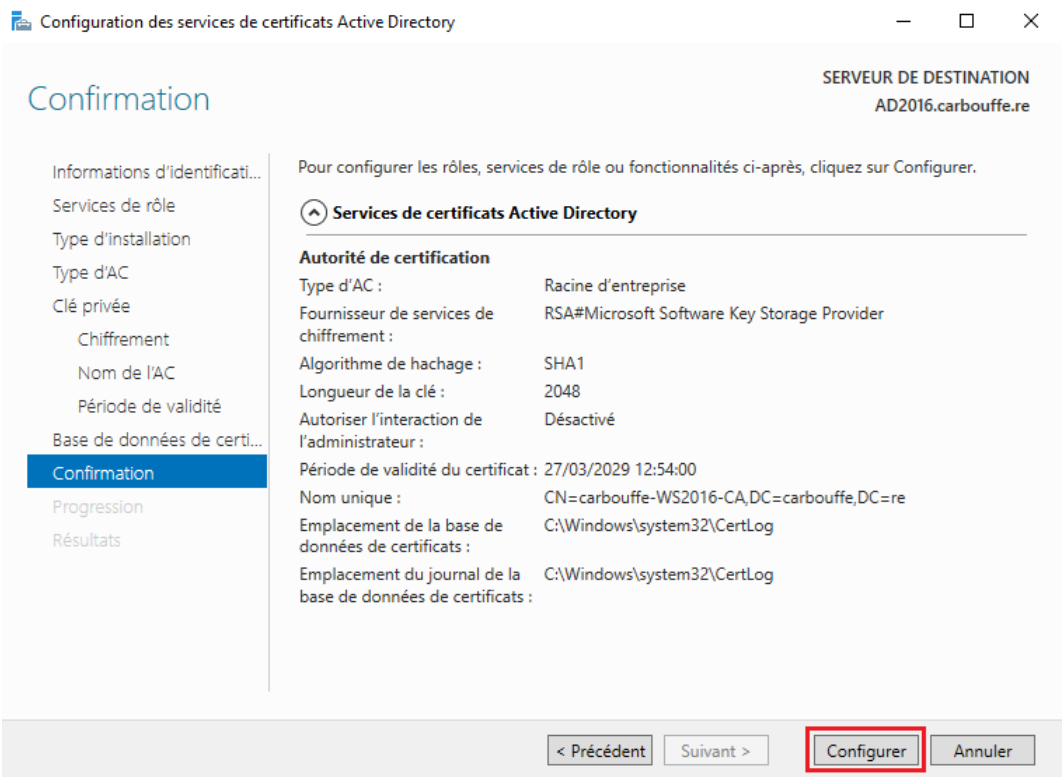
Spécifier la période de validité du certificat :

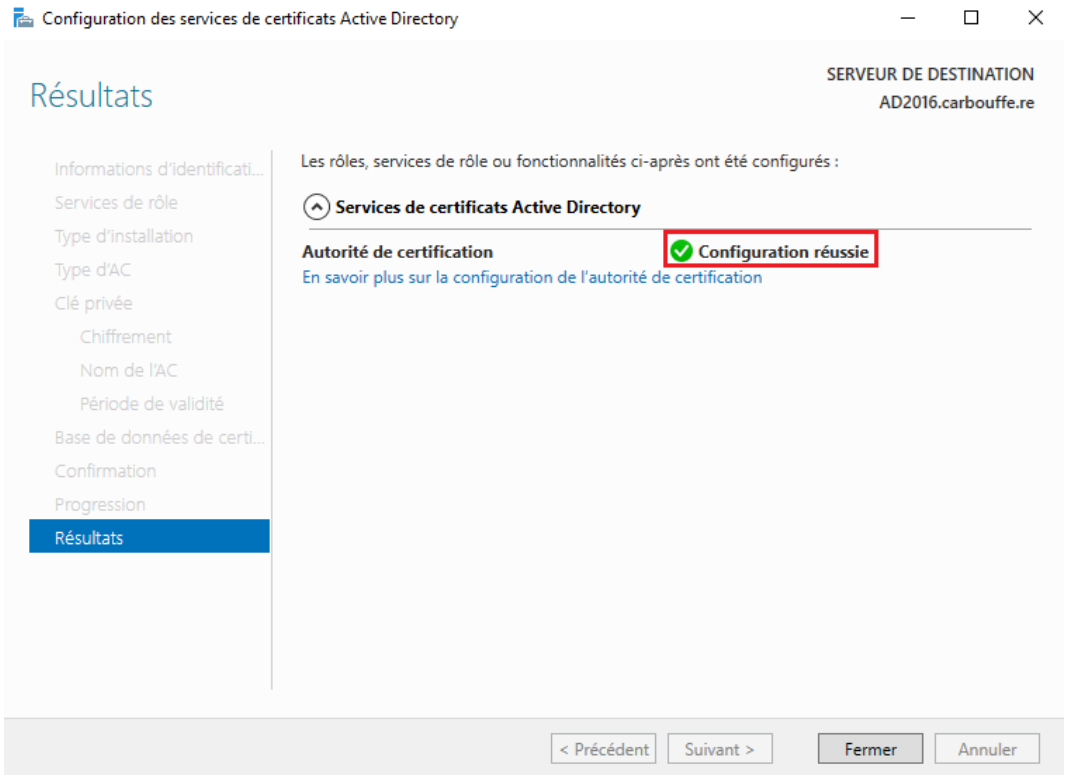


Spécifier les emplacements des bases de données :



Vérifier votre configuration puis valider la :



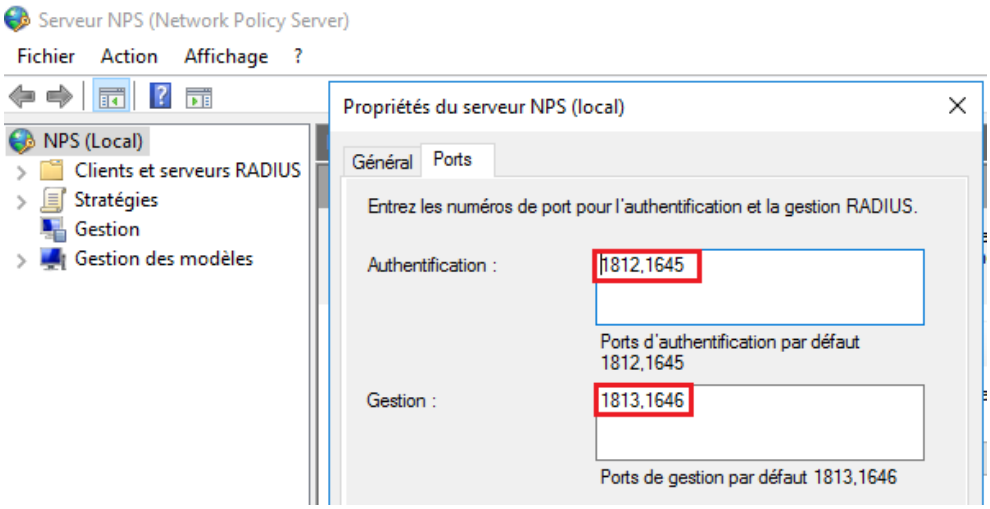

Configuration des services de certificats Active Directory
Résultats
SERVEUR DE DESTINATION
AD2016.carbouffe.re
Informations d'identificati...
Services de rôle
Type d'installation
Type d'AC
Clé privée
Chiffrement
Nom de l'AC
Période de validité
Base de données de certi...
Confirmation
Progression
Résultats
Les rôles, services de rôle ou fonctionnalités ci-après ont été configurés :
Services de certificats Active Directory
Autorité de certification
Configuration réussie
En savoir plus sur la configuration de l'autorité de certification
< Précédent
Suivant >
Fermer
Annuler

Configuration NPS

Pour vérifier le bon fonctionnement du service NPS sur le serveur, vous pouvez afficher les ports en écoute sur celui-ci avec la commande **netstat -a** :

```
UDP    [fe80::b10c:c5ed:a13f:5c83%6]:1645  *:*
UDP    [fe80::b10c:c5ed:a13f:5c83%6]:1646  *:*
UDP    [fe80::b10c:c5ed:a13f:5c83%6]:1812  *:*
UDP    [fe80::b10c:c5ed:a13f:5c83%6]:1813  *:*
```

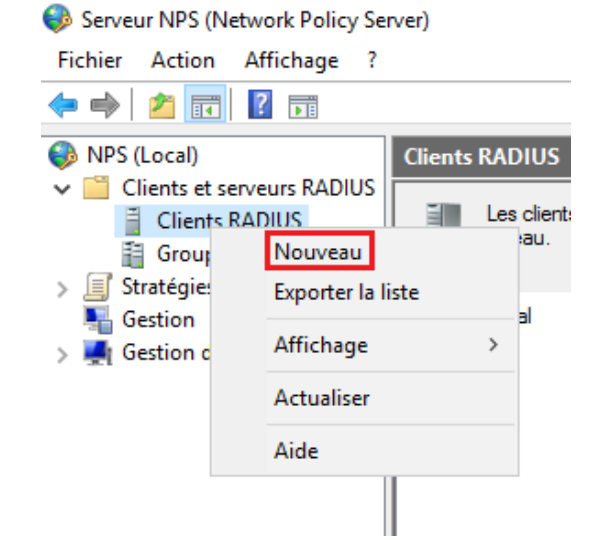
Si vous voyez ces 4 lignes, c'est que le service NPS fonctionne correctement car pour rappels NPS écoute sur les ports 1812,1645 pour l'authentification et 1813,1646 pour la gestion, du moins dans sa configuration par défaut :

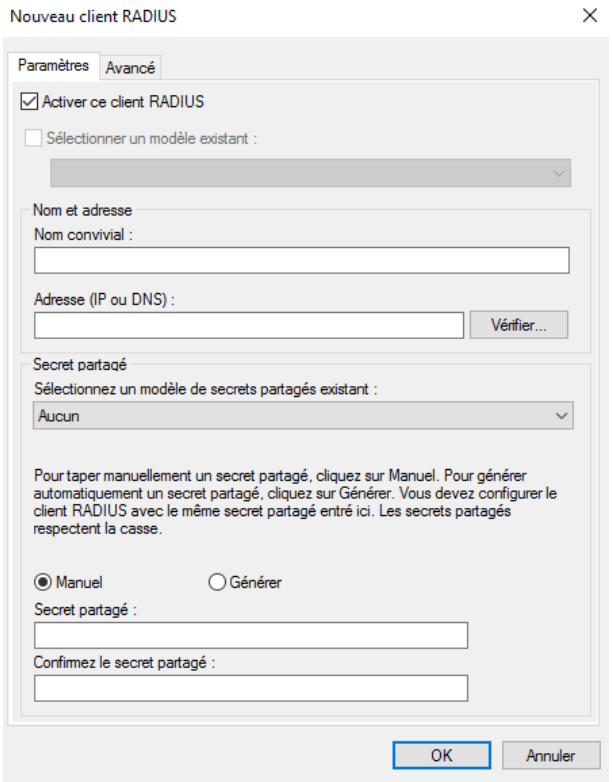


Au préalable, il faudra inscrire NPS dans l'Active Directory pour lui permettre d'interroger la base des utilisateurs. La prochaine étape est la déclaration d'un client RADIUS. Dans le schéma général d'une connexion 802.1x, l'élément central est l'équipement de réseau (commutateur, borne wifi, ...) désigné comme client RADIUS. Cet équipement doit donc être en capacité de gérer le protocole 802.1x et le protocole d'authentification EAP.

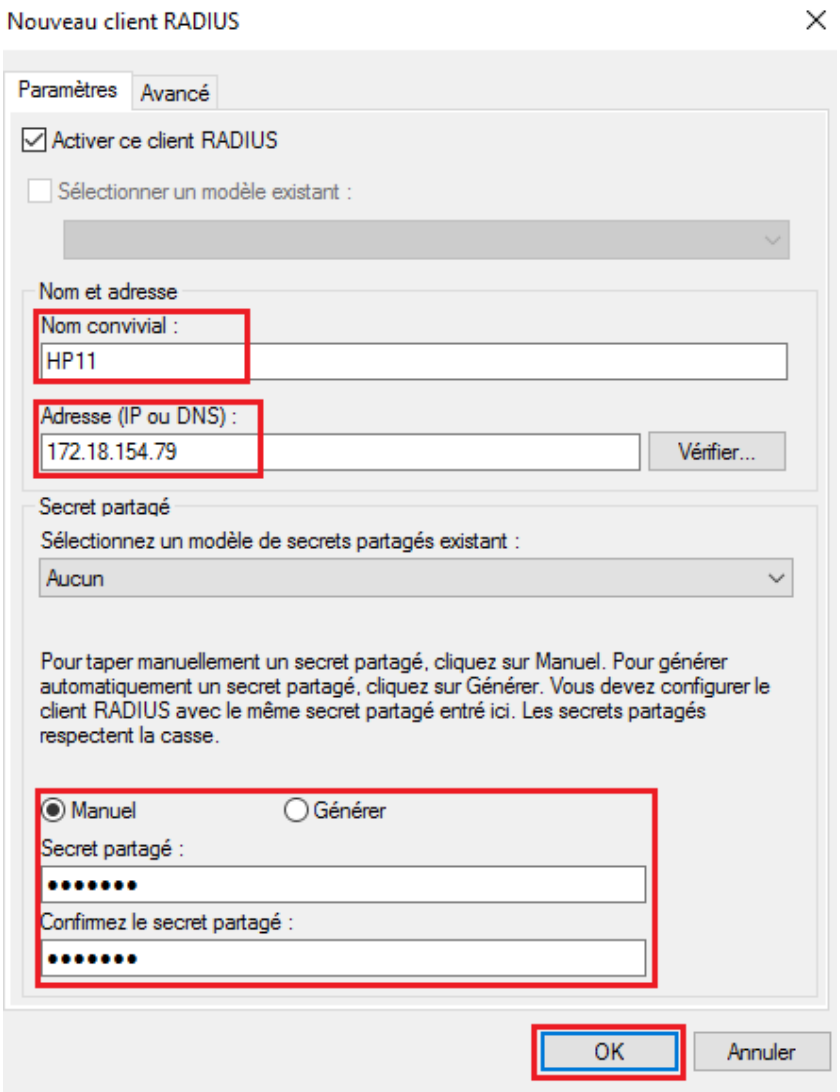
Déclaration Client RADIUS

Dans notre cas, le commutateur est un HP compatible 802.1x. Les éléments à renseigner sont : le nom "convivial" du client-RADIUS, son adresse IP et la chaîne de caractères du "secret partagé" entre le serveur RADIUS et le client RADIUS. Cette chaîne doit évidemment être identique à celle déclarée dans le client radius, c'est-à-dire le commutateur HP. Cliquez droit sur « Client Radius » puis sur « Nouveau » :

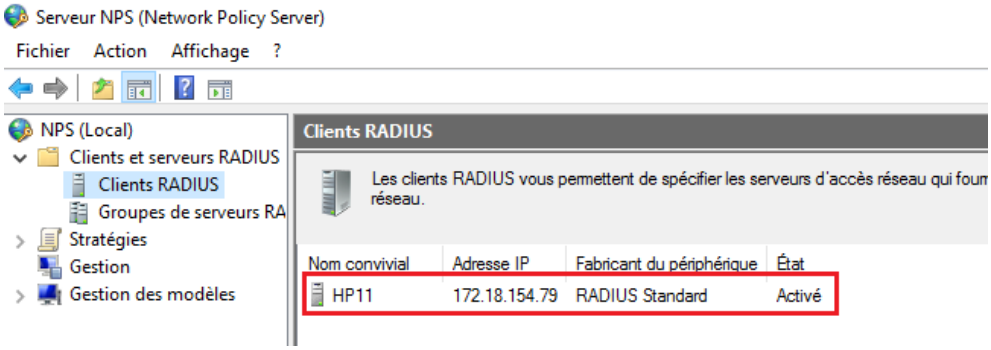




Remplissez les champs du nouveau Client Radius puis cliquer sur « OK » :

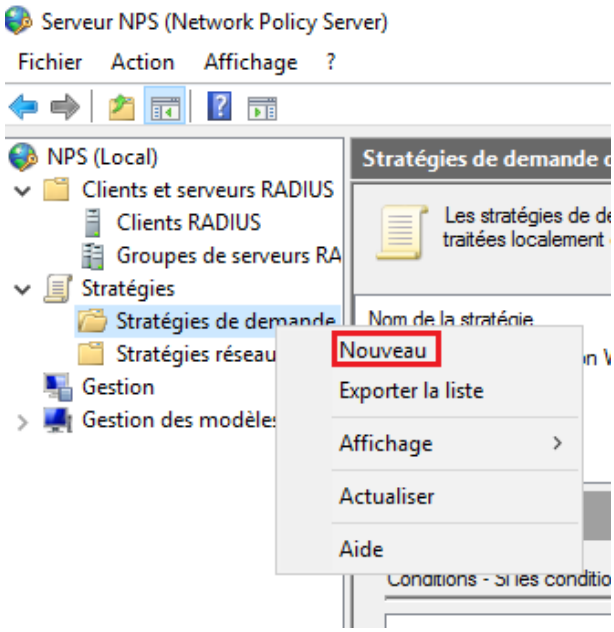


Ce qui donne :

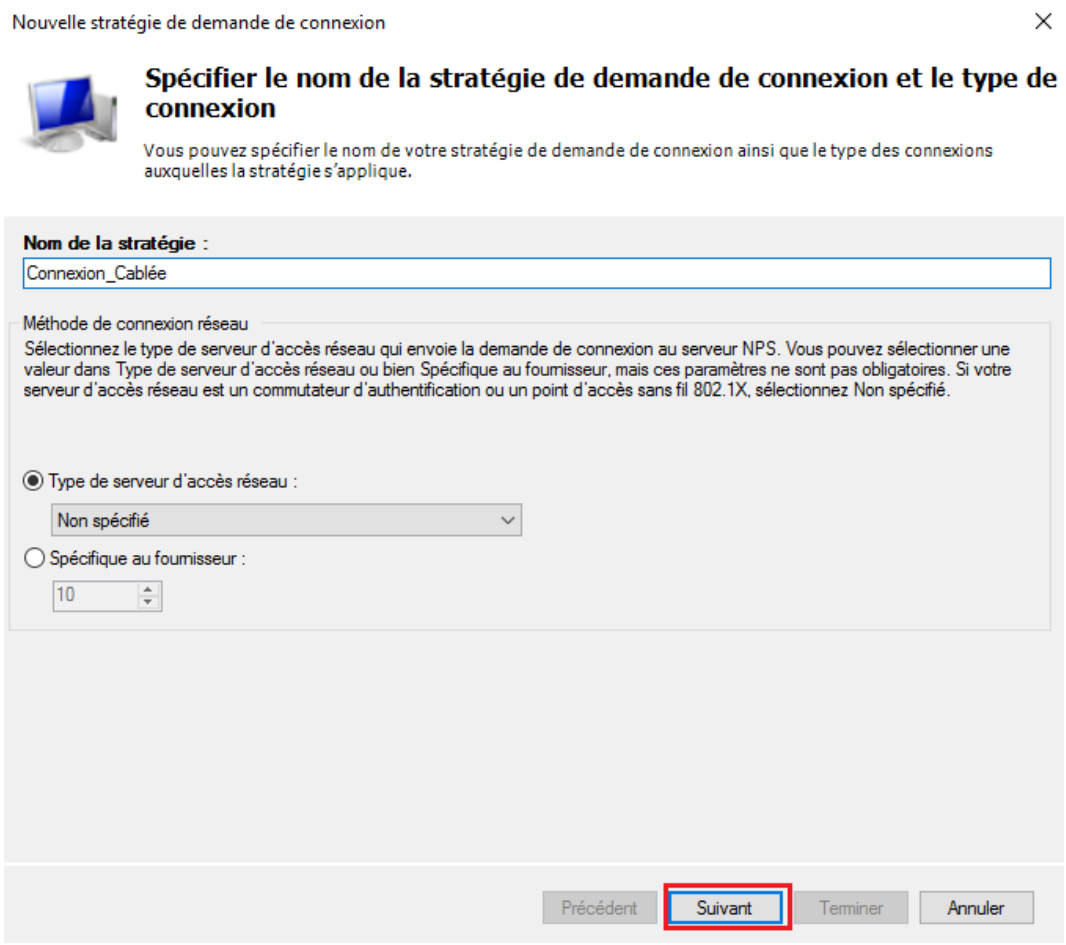


Déclaration Stratégie de demande de connexion

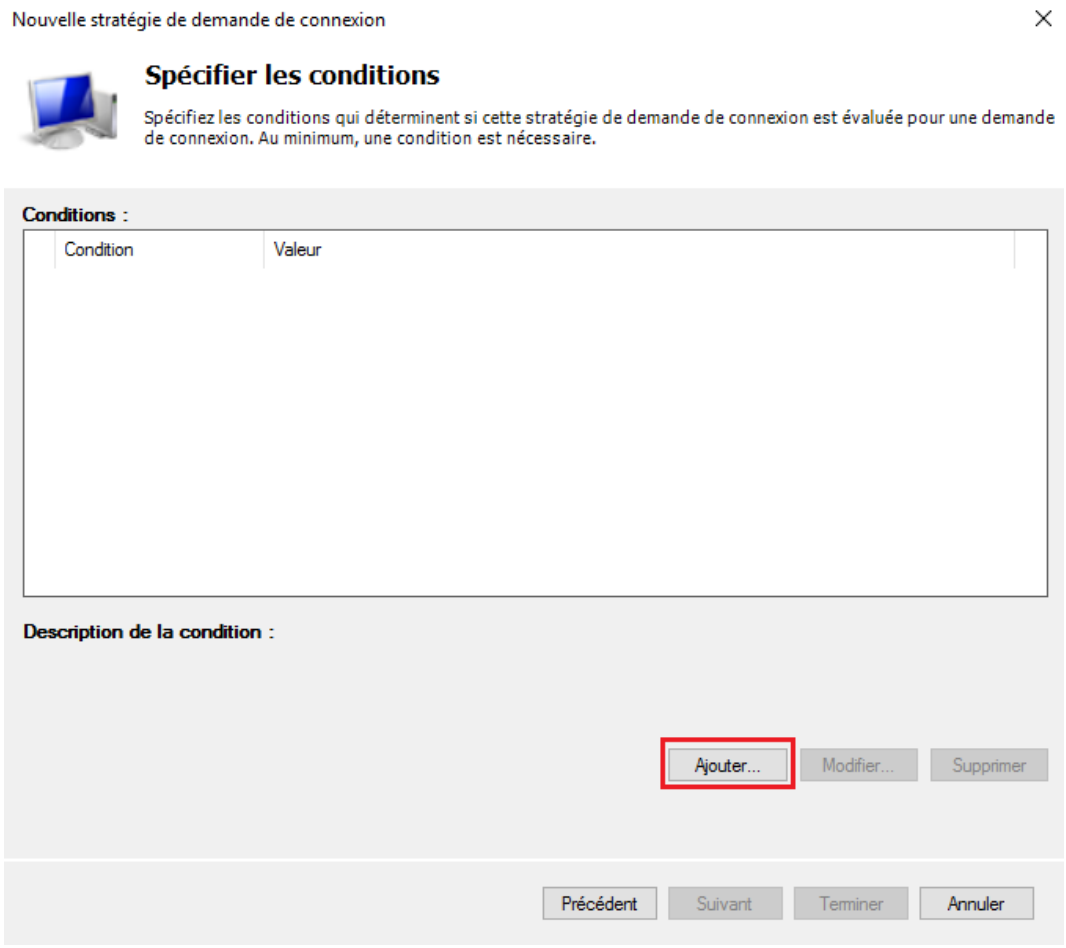
Ensuite, il faut déclarer une ou plusieurs stratégies de demande de connexion (notamment pour Ethernet, il s'agit de la connexion physique au média). Pour cela, on fait un clic droit sur « Stratégies de demande de connexion » puis « Nouveau » :



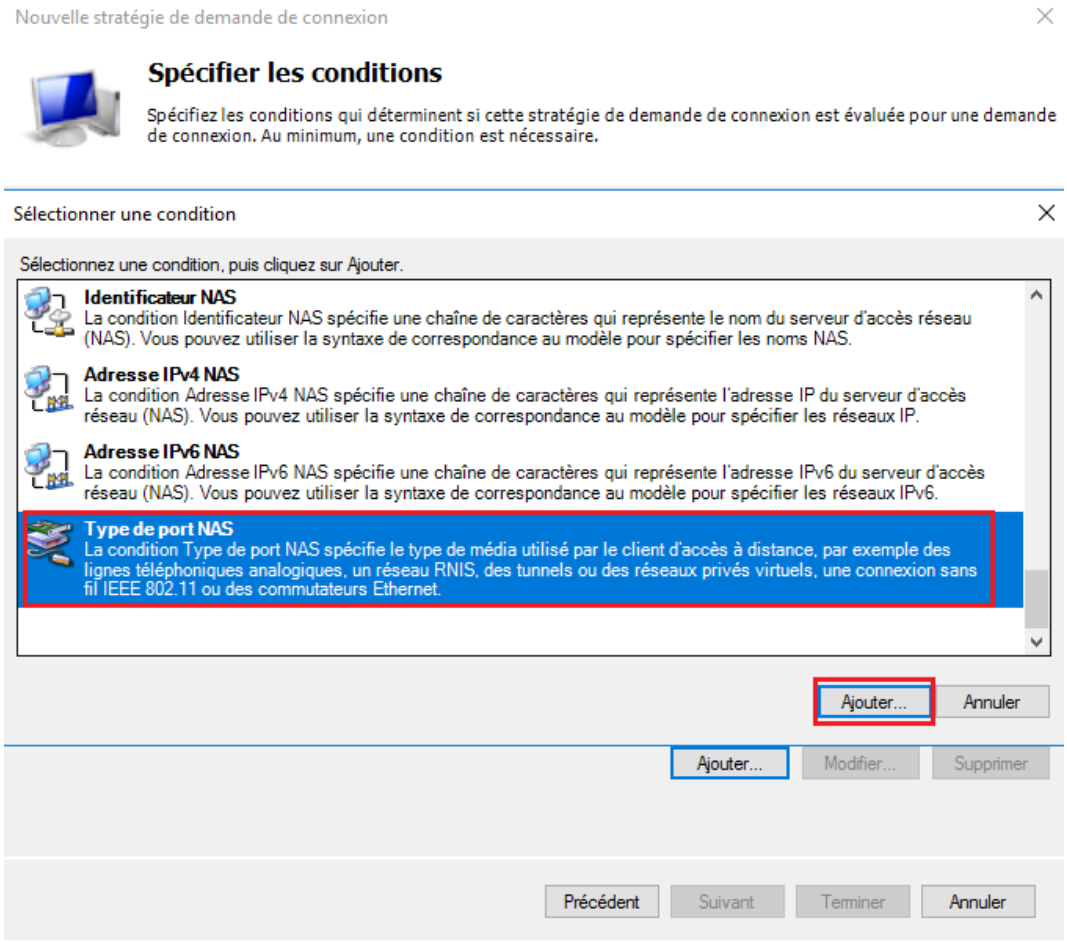
On rentre le nom de la stratégie et on laisse le type de serveur sur « Non spécifié » (nous utilisons un commutateur en tant que client Radius), puis on clique sur « Suivant » :



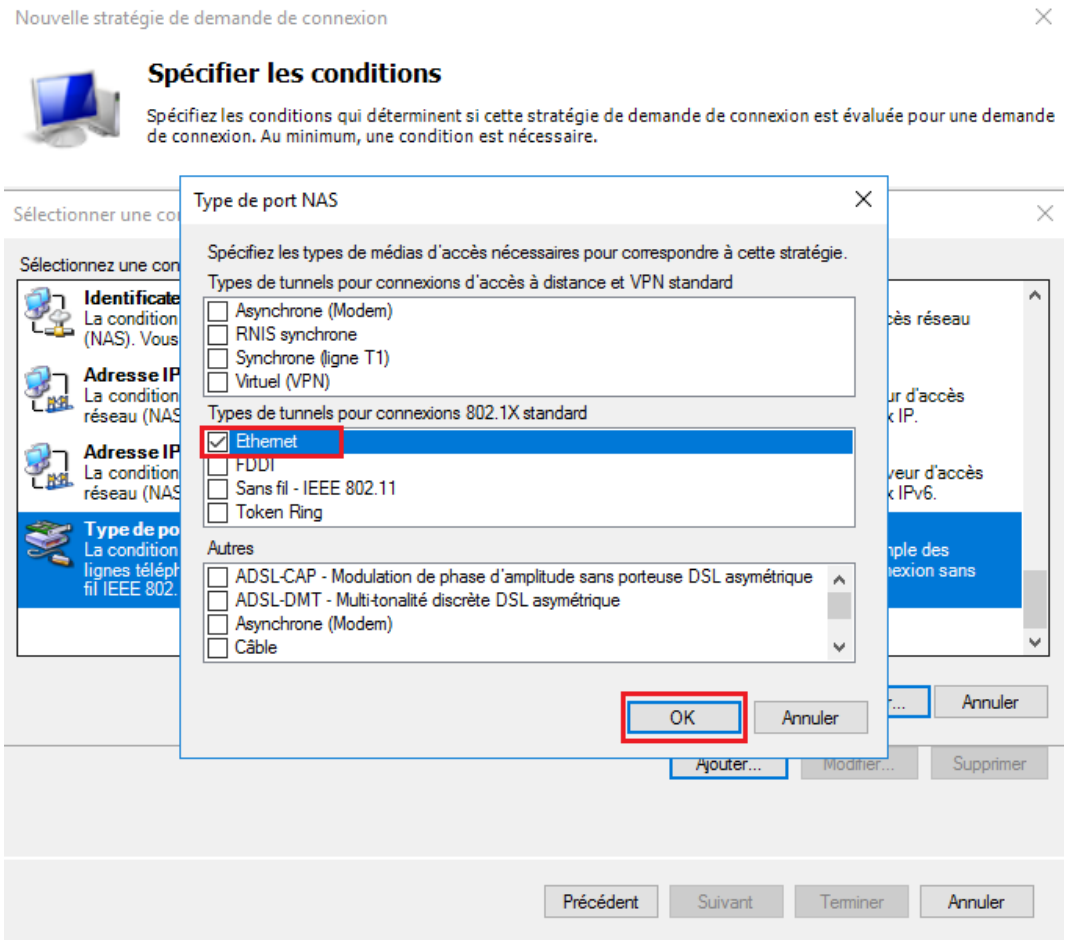
On vient ajouter une nouvelle condition :



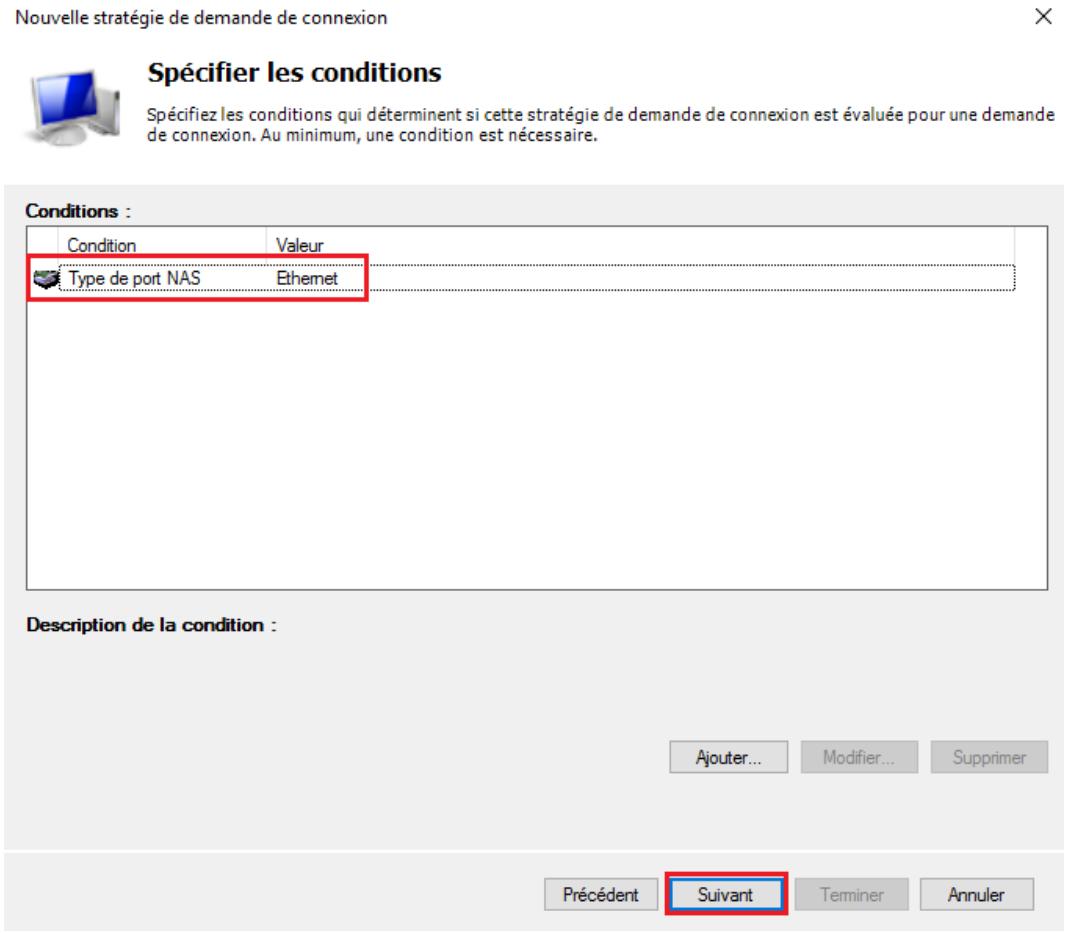
On choisit d'indiquer un type de port NAS (type de média connecté). NAS est ici l'acronyme "Network Access Server" et désigne le client RADIUS. Ne pas confondre avec Network Authentication Server, qui désigne le serveur Radius lui-même :



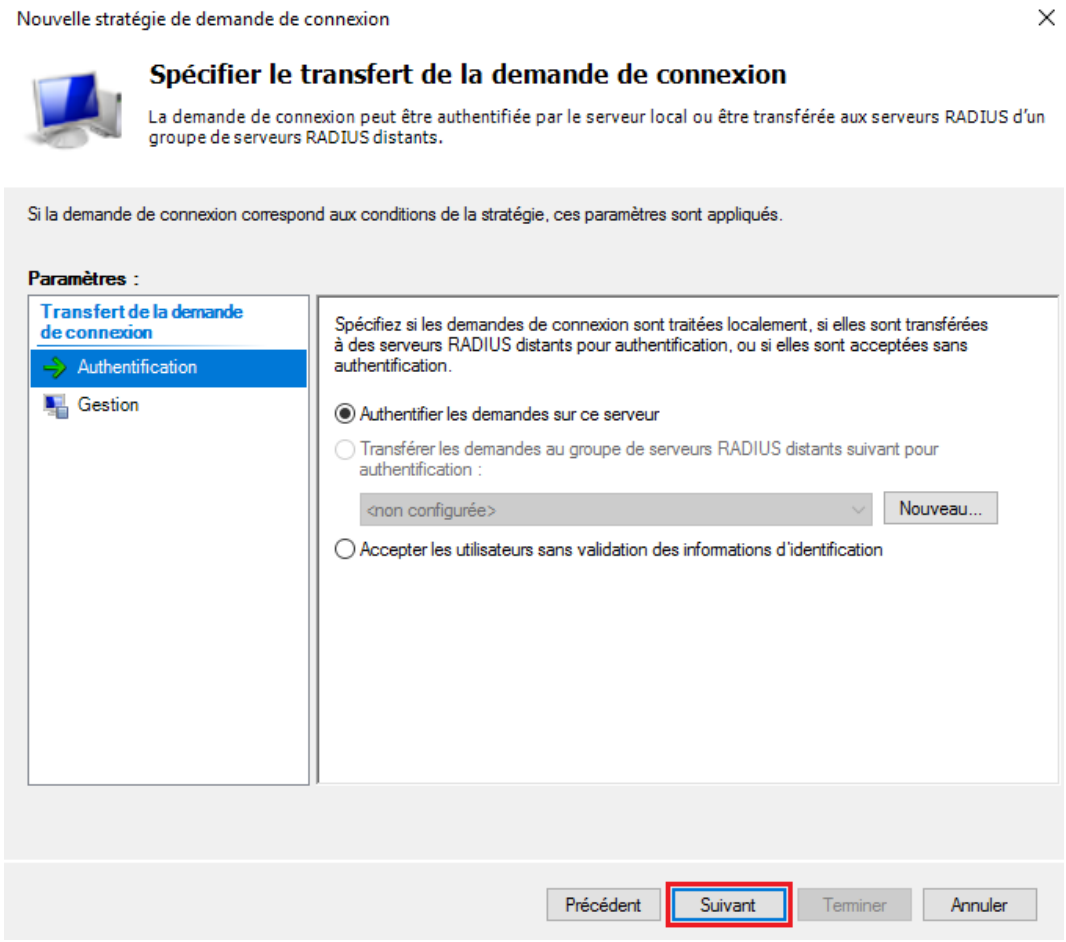
On coche « Ethernet » puis « OK » :



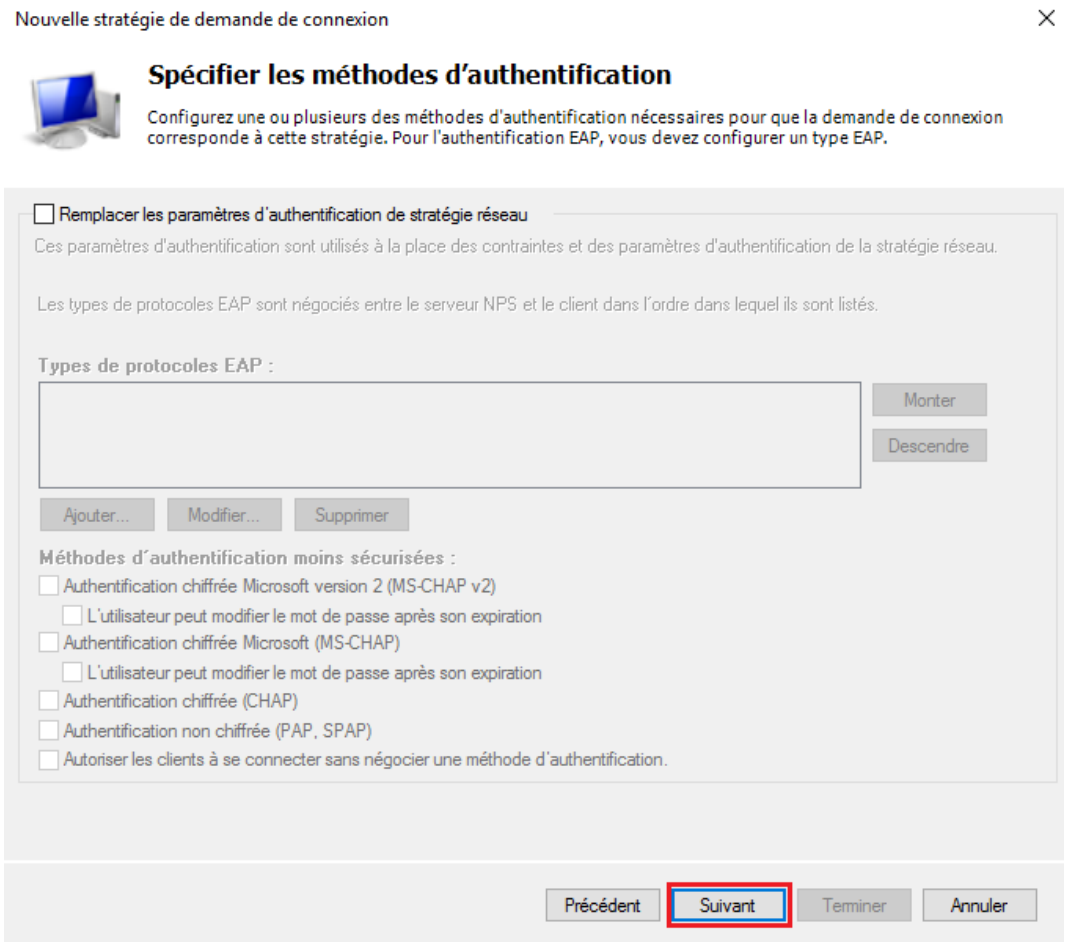
Une nouvelle condition de Type de port NAS avec la valeur Ethernet a été ajouter :



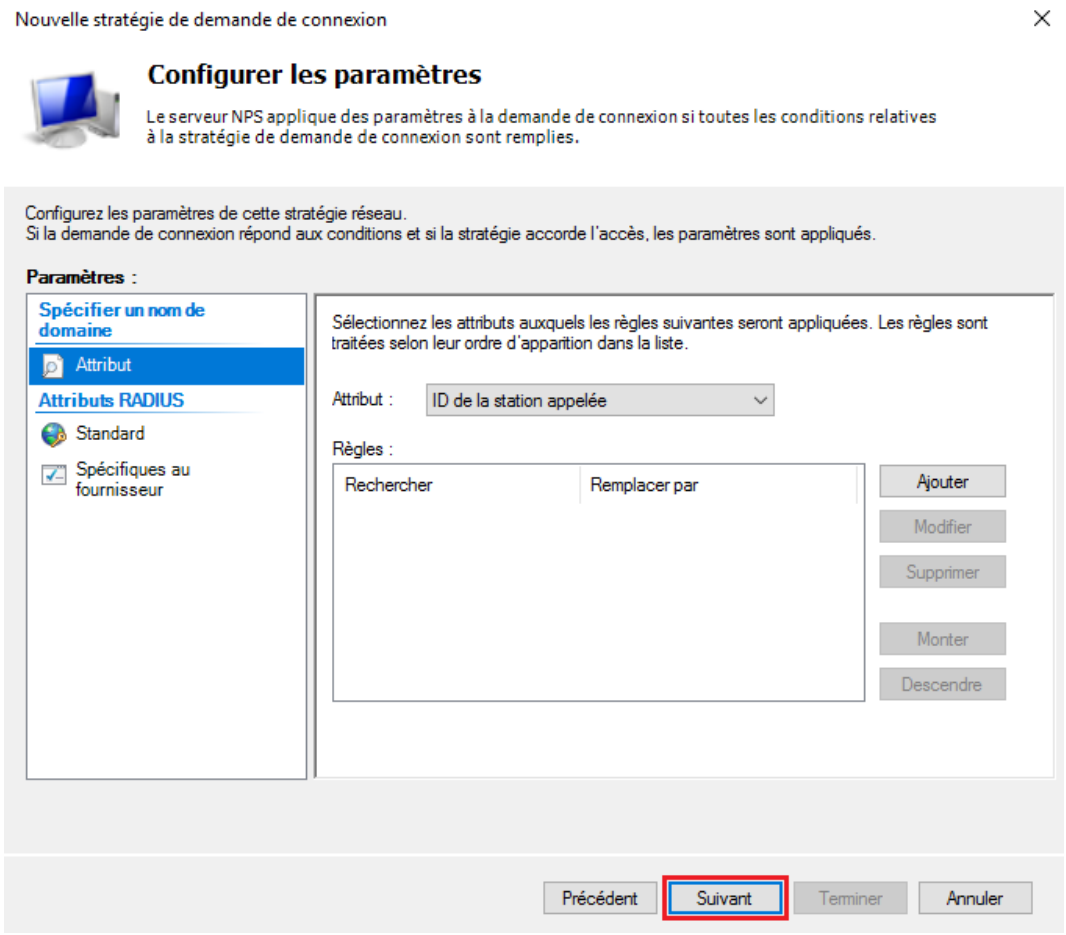
Pour le transfert de la demande de connexion, on laisse les paramètres par défaut. Dans notre situation, les demandes seront traitées sur ce serveur et non sur un autre. Ce qui veut dire que ce NPS pourrait jouer un rôle de "PROXY NPS" s'il relayait les demandes à un autre serveur :



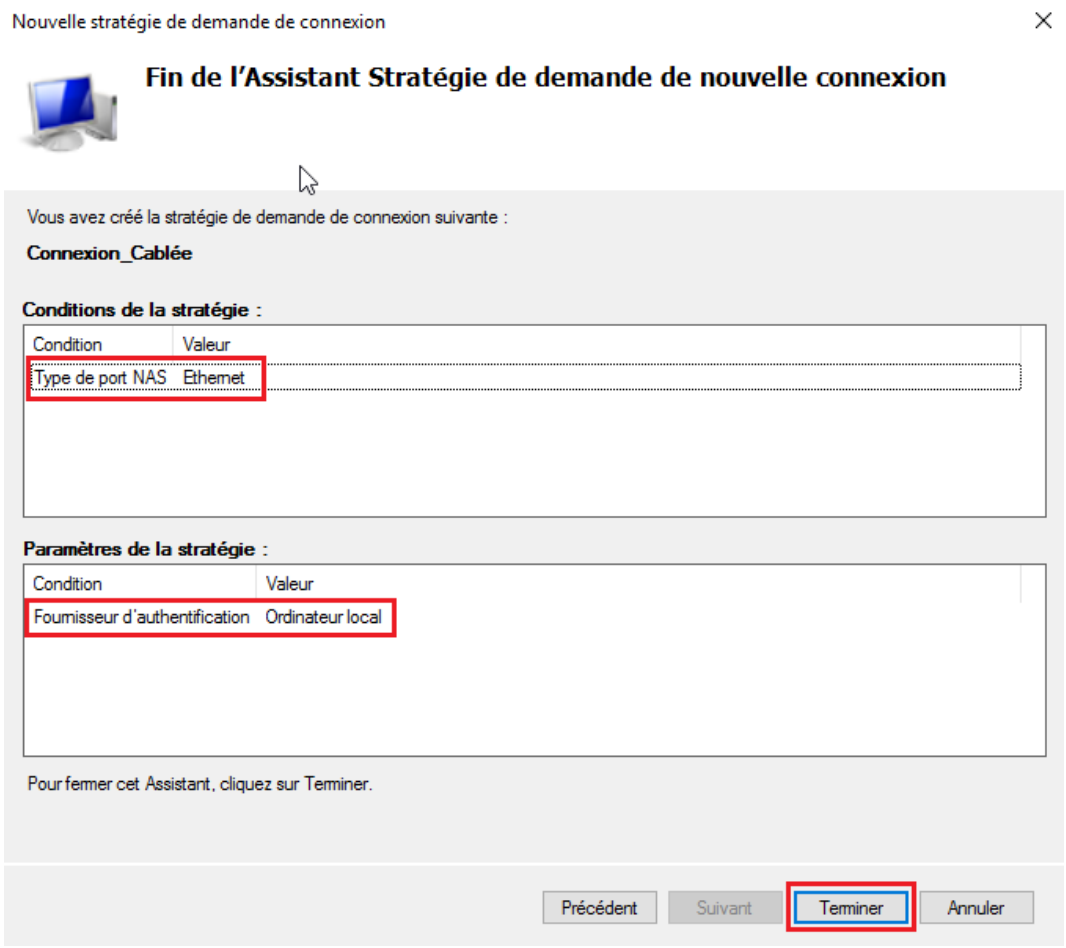
De même pour les méthodes d'authentification, c'est la stratégie d'accès réseau que l'on va maintenant déclarer qui va primer :



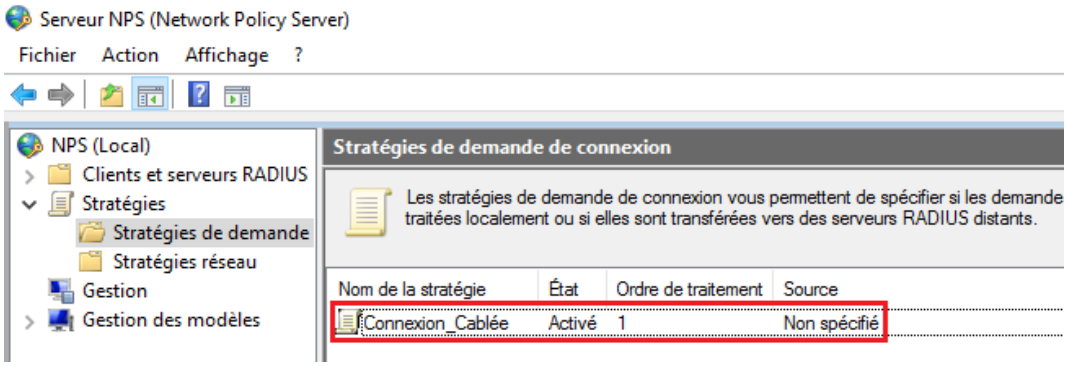
Les paramètres sont également laissés par défaut :



Vous vérifiez votre configuration grâce à la console de l'assistant de stratégie de demande de nouvelle connexion et vous validez si cela vous convient :

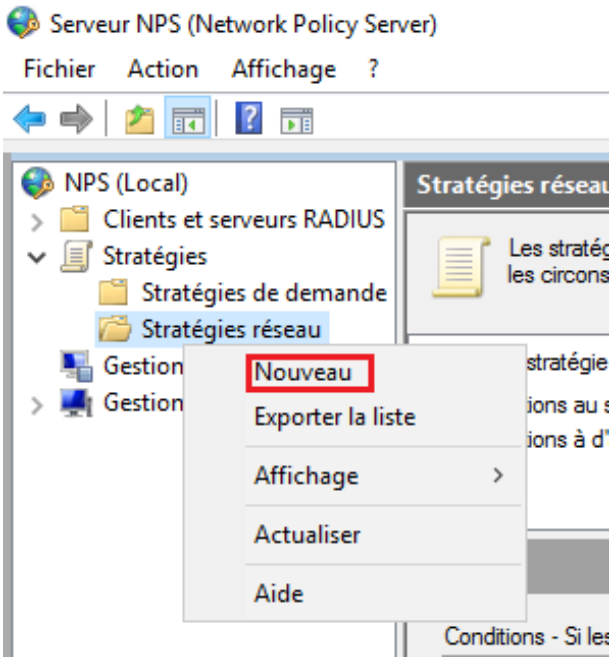


Ce qui donnera :

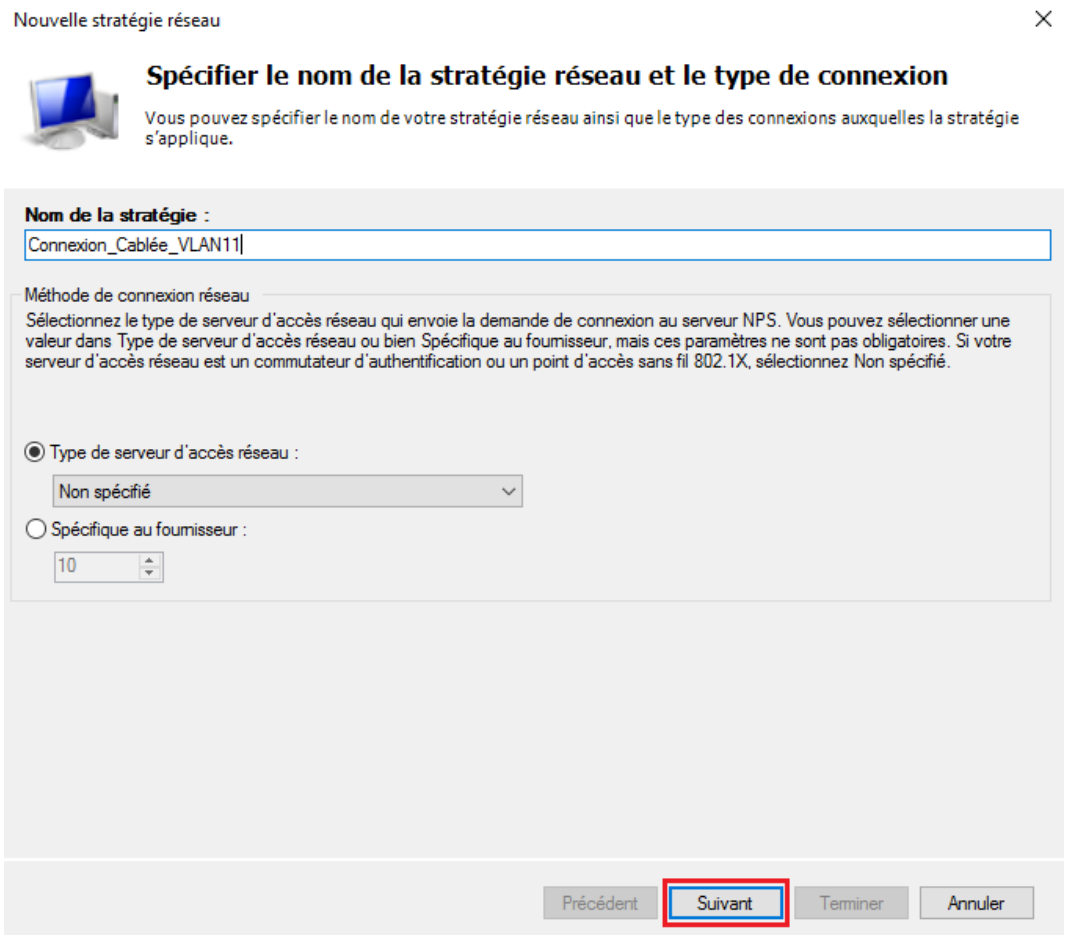


Déclaration Stratégie Réseau

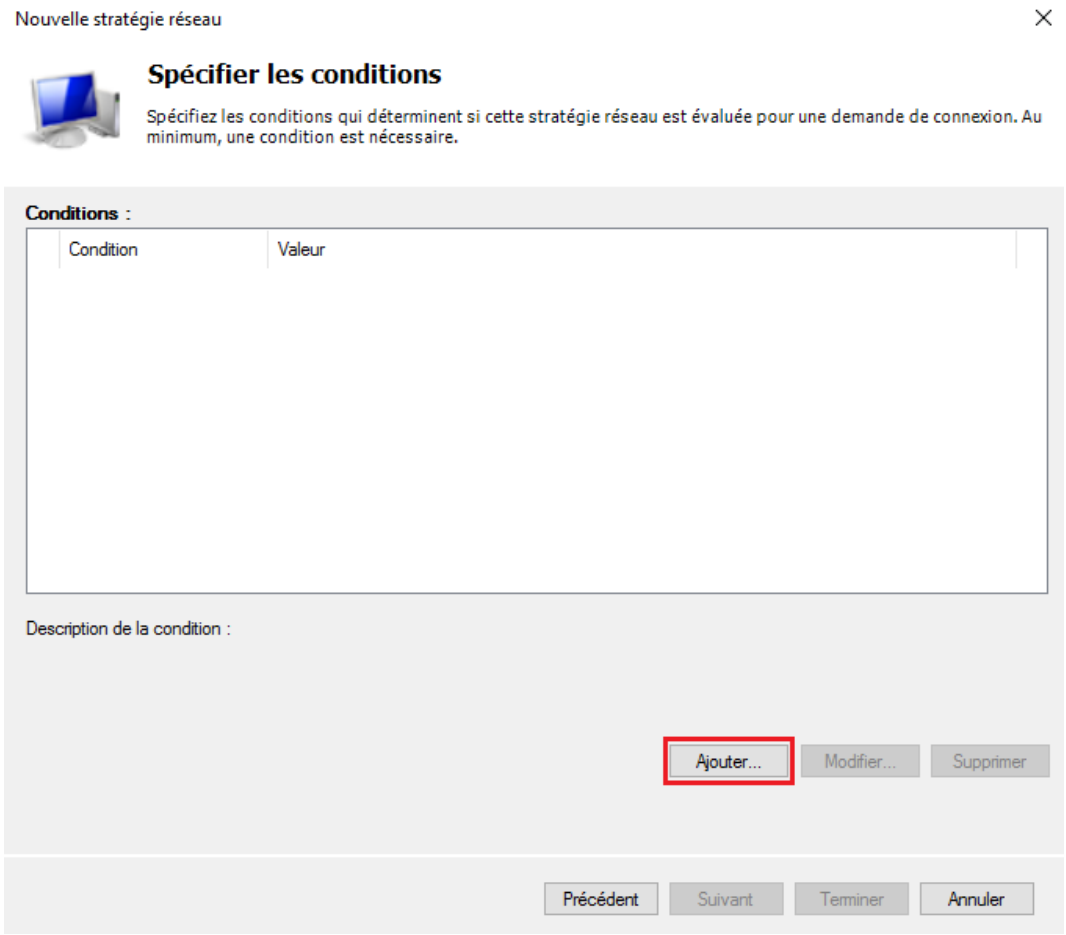
On va maintenant mettre en place une stratégie réseau. Cette dernière va permettre un placement dynamique dans le VLAN 11 pour les membres du groupe d'utilisateurs "Carbouffe_RH". Le commutateur client-RADIUS se chargera lui-même du placement dans un VLAN "guest" des utilisateurs non authentifiés. On créer une nouvelle stratégie réseau comme pour une stratégie de demande d'accès :

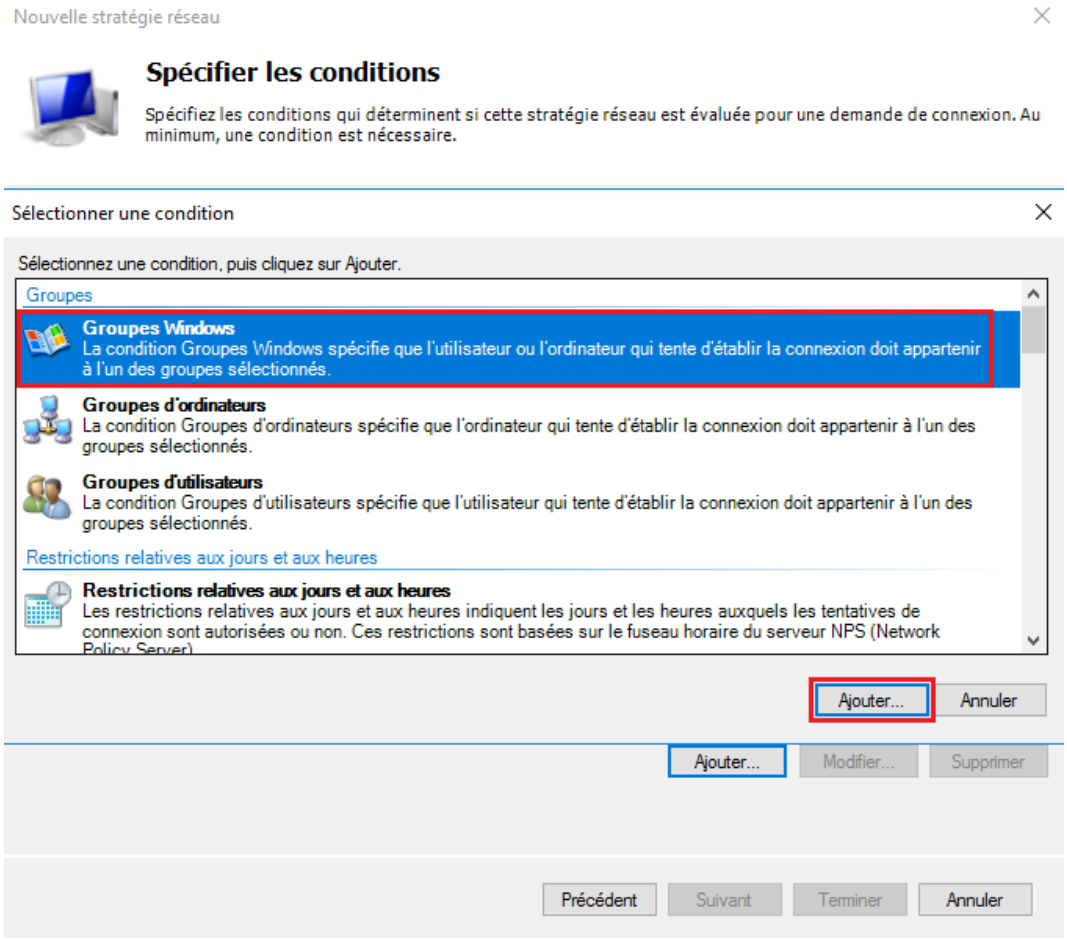


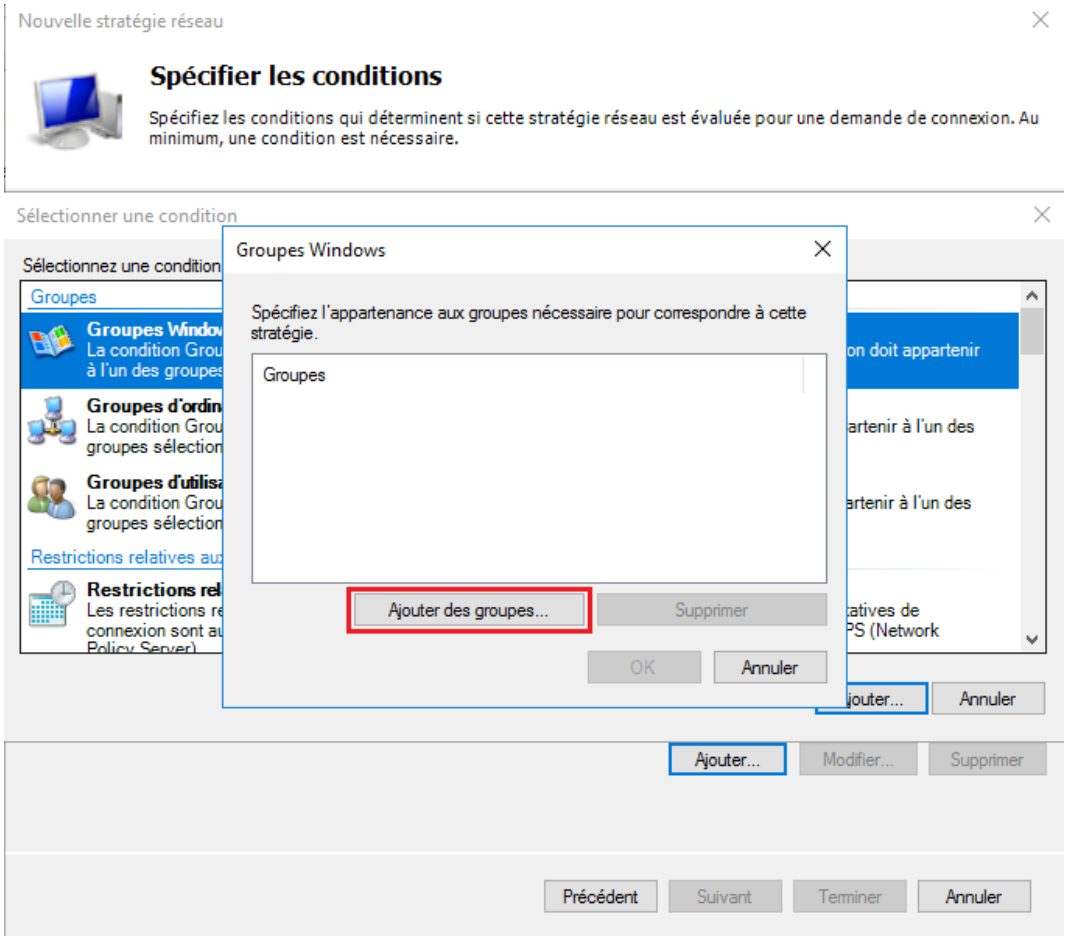
On reste sur un type non spécifié car s'agit d'une authentification via un commutateur 802.1x :

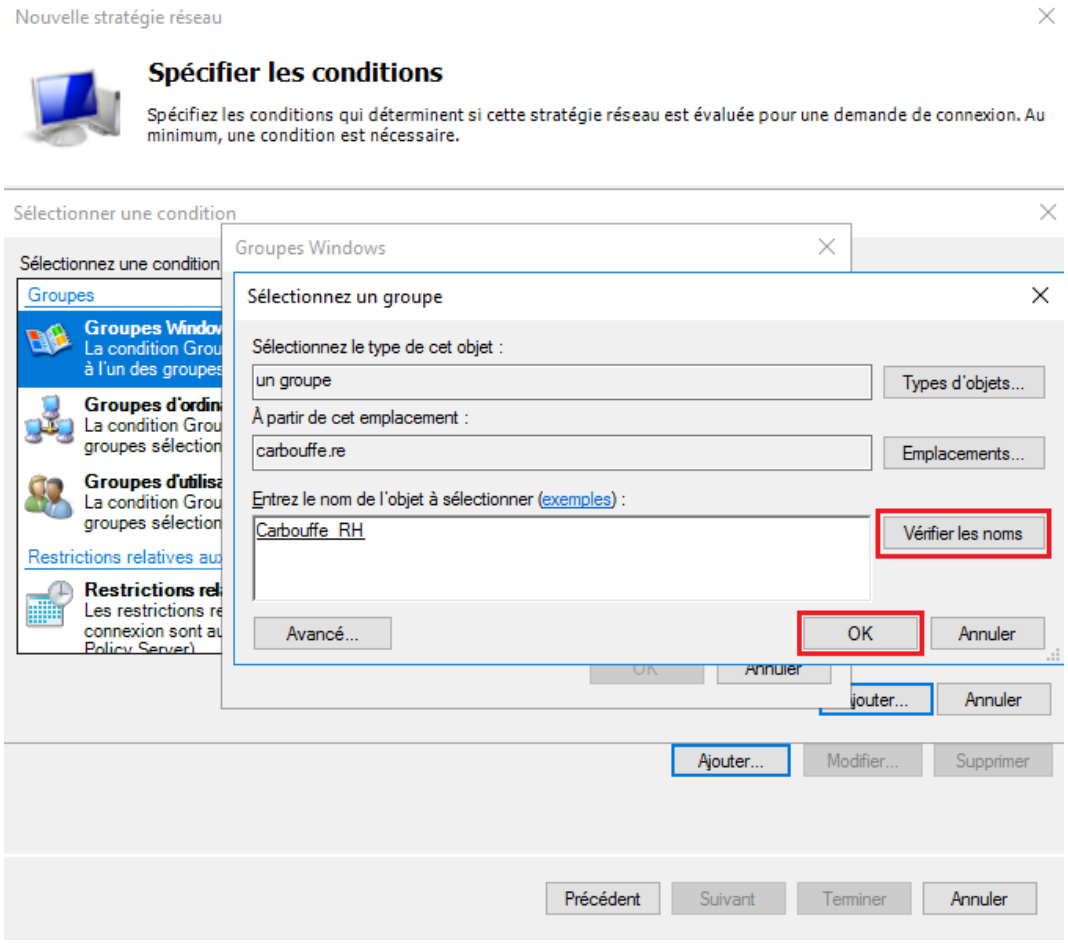


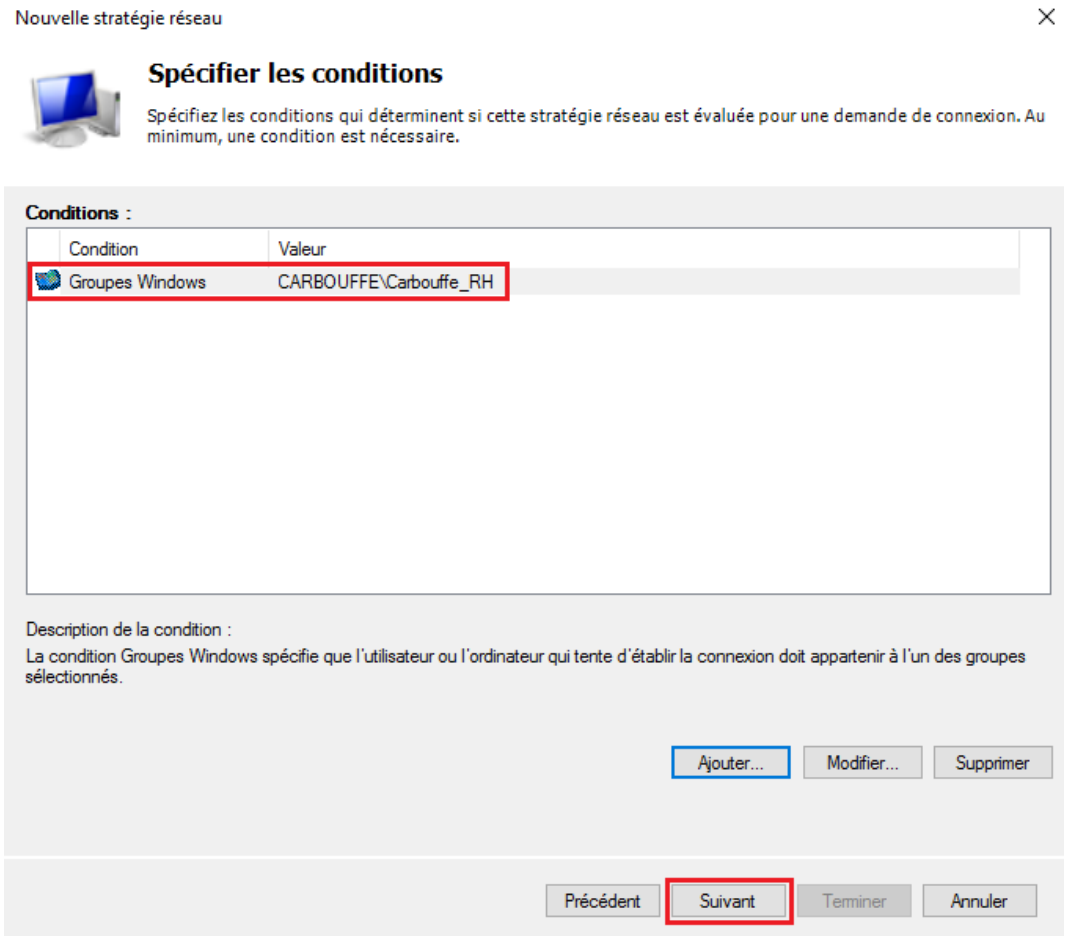
On ajoute une nouvelle condition à la validation de la stratégie : il faut que l'utilisateur soit membre d'un groupe AD qui s'appelle "Carbouffe_RH". Pour les membres de ce groupe, on accorde l'accès. Choisir Groupe Windows et non Groupes d'utilisateurs :



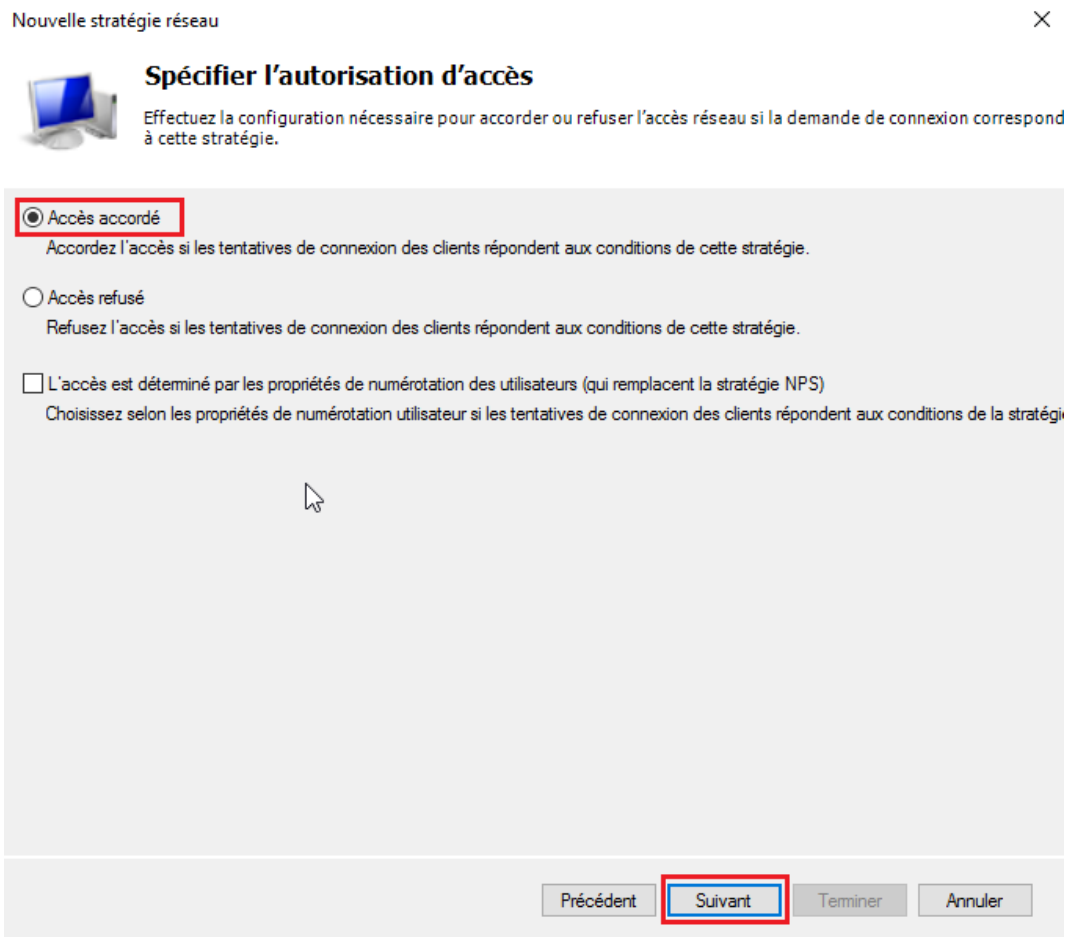

Nouvelle stratégie réseau
Spécifier les conditions
Spécifiez les conditions qui déterminent si cette stratégie réseau est évaluée pour une demande de connexion. Au minimum, une condition est nécessaire.
Sélectionner une condition
Sélectionnez une condition, puis cliquez sur Ajouter.
Groupes
Groupes Windows
La condition Groupes Windows spécifie que l'utilisateur ou l'ordinateur qui tente d'établir la connexion doit appartenir à l'un des groupes sélectionnés.
Groupes d'ordinateurs
La condition Groupes d'ordinateurs spécifie que l'ordinateur qui tente d'établir la connexion doit appartenir à l'un des groupes sélectionnés.
Groupes d'utilisateurs
La condition Groupes d'utilisateurs spécifie que l'utilisateur qui tente d'établir la connexion doit appartenir à l'un des groupes sélectionnés.
Restrictions relatives aux jours et aux heures
Restrictions relatives aux jours et aux heures
Les restrictions relatives aux jours et aux heures indiquent les jours et les heures auxquels les tentatives de connexion sont autorisées ou non. Ces restrictions sont basées sur le fuseau horaire du serveur NPS (Network Policy Server)
Ajouter...
Annuler
Ajouter...
Modifier...
Supprimer
Précédent
Suivant
Terminer
Annuler


Nouvelle stratégie réseau
Spécifier les conditions
Spécifiez les conditions qui déterminent si cette stratégie réseau est évaluée pour une demande de connexion. Au minimum, une condition est nécessaire.
Sélectionner une condition
Sélectionnez une condition
Groupes
Groupes Windov
La condition Grou
à l'un des groupes
Groupes d'ordin
La condition Grou
groupes sélection
Groupes d'utilisa
La condition Grou
groupes sélection
Restrictions relatives au
Restrictions rel
Les restrictions re
connexion sont au
Policy Server)
Groupes Windows
Spécifiez l'appartenance aux groupes nécessaire pour correspondre à cette stratégie.
Groupes
Ajouter des groupes...
Supprimer
OK
Annuler
on doit appartenir
artenir à l'un des
artenir à l'un des
atives de
PS (Network
jouter...
Annuler
Ajouter...
Modifier...
Supprimer
Précédent
Suivant
Terminer
Annuler

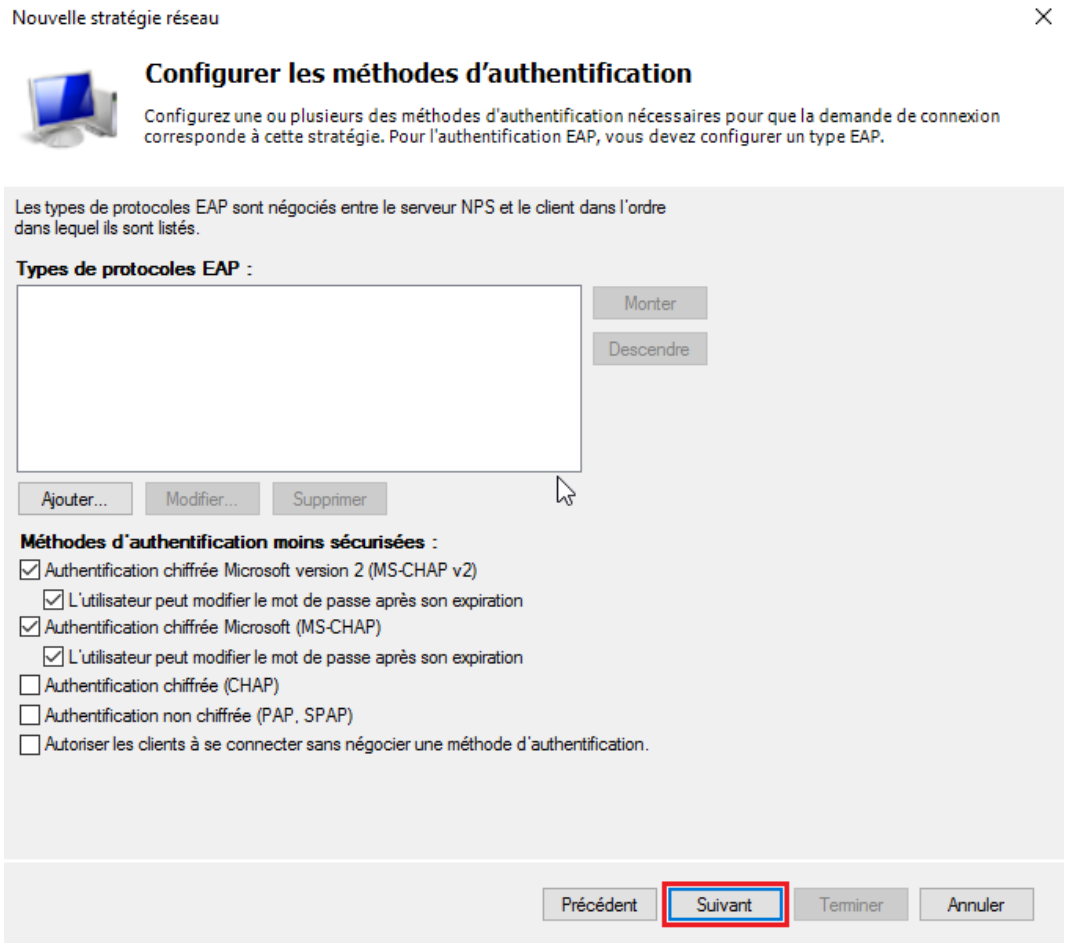

Nouvelle stratégie réseau
Spécifier les conditions
Spécifiez les conditions qui déterminent si cette stratégie réseau est évaluée pour une demande de connexion. Au minimum, une condition est nécessaire.
Sélectionner une condition
Sélectionnez une condition
Groupes
Groupes Windov
La condition Grou
à l'un des groupes
Groupes d'ordin
La condition Grou
groupes sélection
Groupes d'utilisa
La condition Grou
groupes sélection
Restrictions relatives au
Restrictions rel
Les restrictions ré
connexion sont au
Policy Server)
Groupes Windows
Sélectionnez un groupe
Sélectionnez le type de cet objet :
un groupe
Types d'objets...
À partir de cet emplacement :
carbouffe.re
Emplacements...
Entrez le nom de l'objet à sélectionner (exemples) :
Carbouffe_RH
Vérifier les noms
Avancé...
OK
Annuler
Annuler
jouter...
Annuler
Ajouter...
Modifier...
Supprimer
Précédent
Suivant
Terminer
Annuler


Nouvelle stratégie réseau
Spécifier les conditions
Spécifiez les conditions qui déterminent si cette stratégie réseau est évaluée pour une demande de connexion. Au minimum, une condition est nécessaire.
Conditions :
Condition
Valeur
Groupes Windows
CARBOUFFE\Carbouffe_RH
Description de la condition :
La condition Groupes Windows spécifie que l'utilisateur ou l'ordinateur qui tente d'établir la connexion doit appartenir à l'un des groupes sélectionnés.
Ajouter...
Modifier...
Supprimer
Précédent
Suivant
Terminer
Annuler

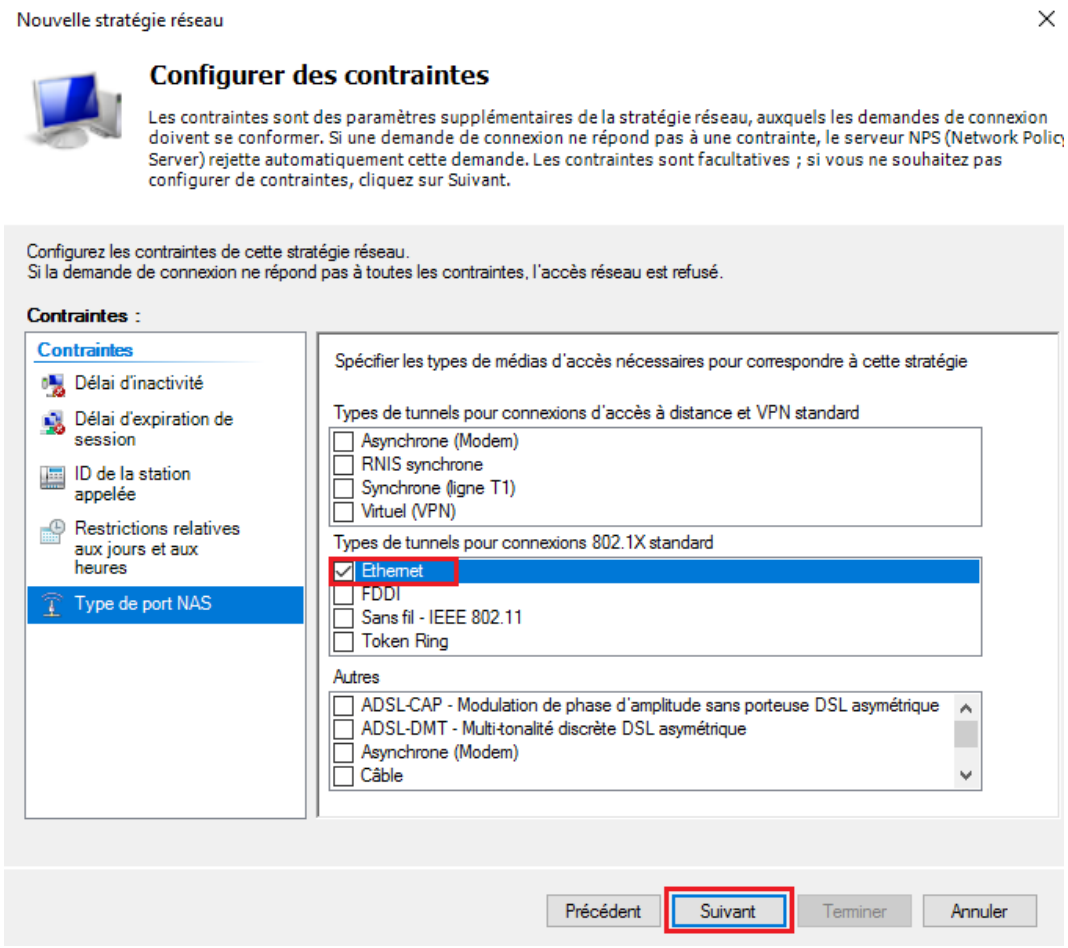
On peut définir des stratégies autorisant l'accès quand les conditions sont réunies ou, à l'inverse, interdisant l'accès lorsque les conditions sont réunies (un groupe d'utilisateur en congé par exemple):



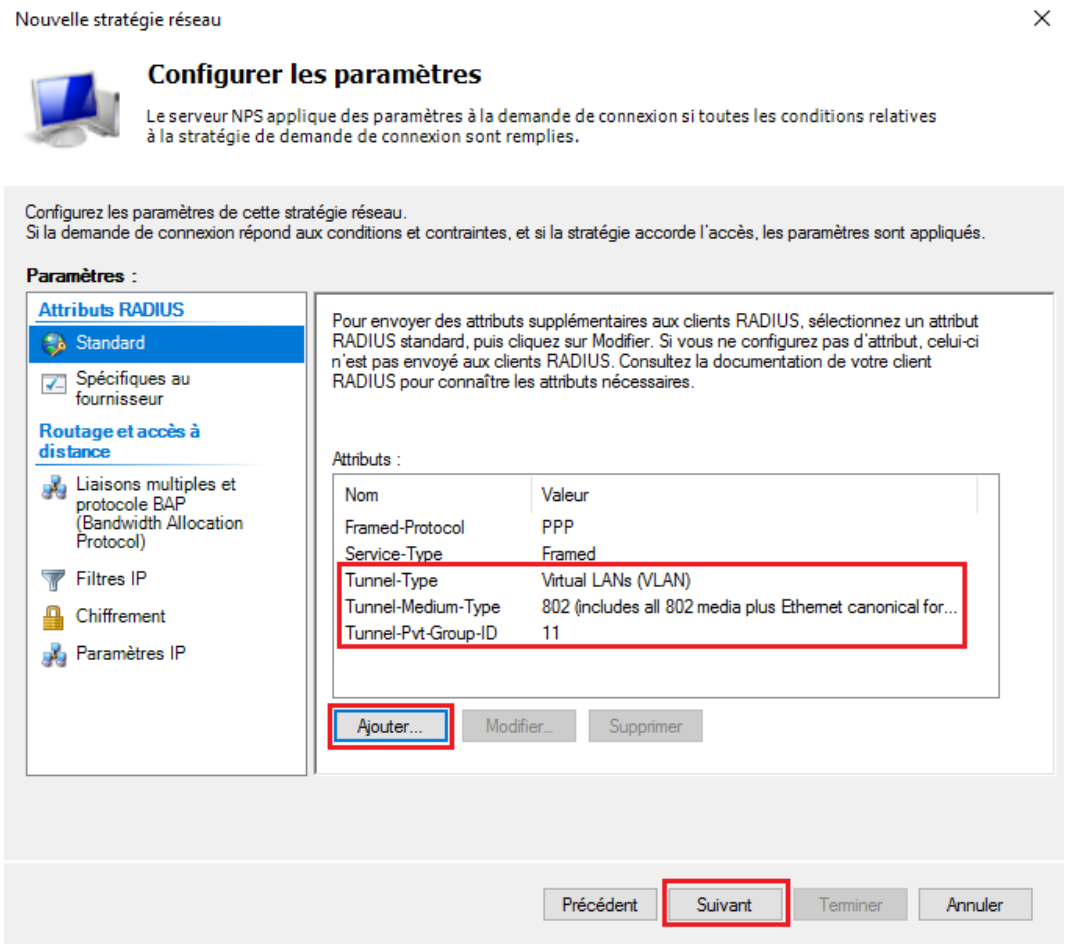
On déclare ensuite les types de protocoles EAP accepté : PEAP :



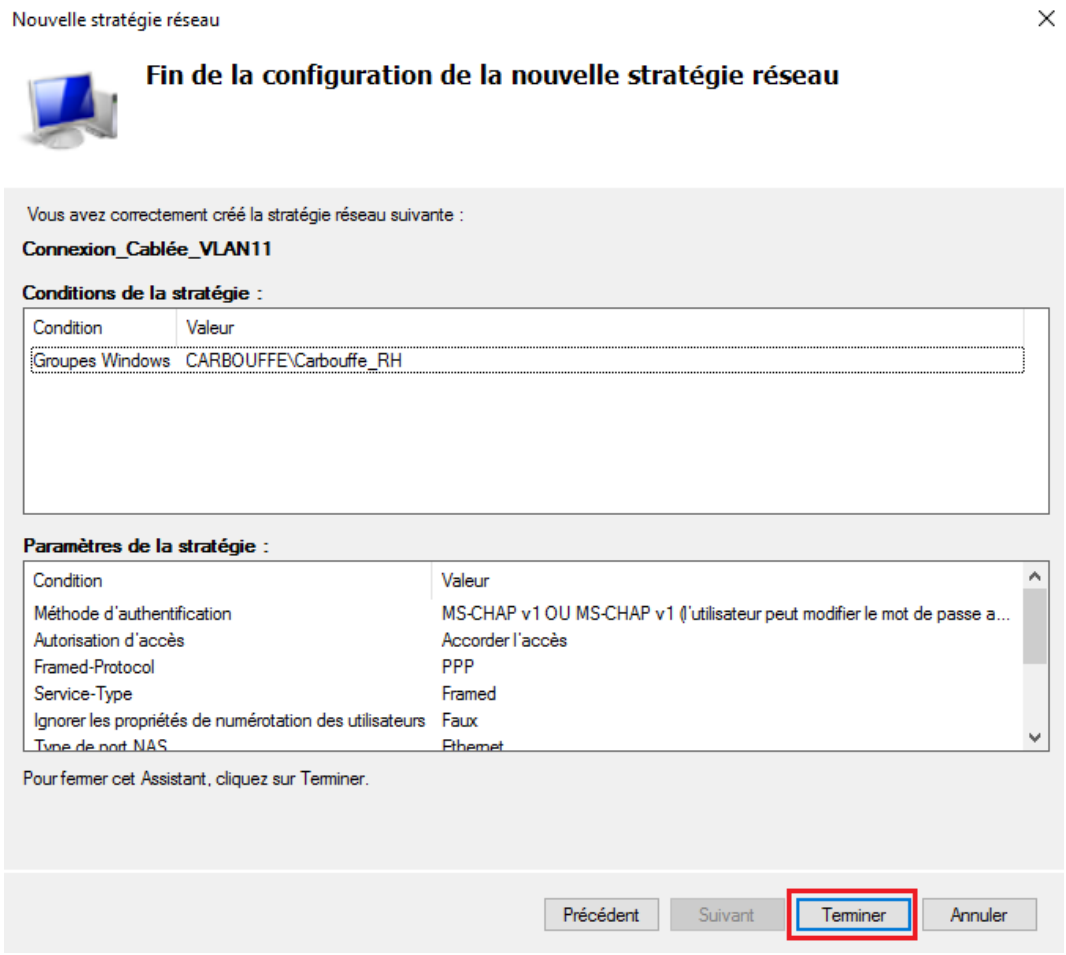
On accepte les Types de ports NAS Ethernet. C'est ici que se fait le lien avec la stratégie de demande de connexion :



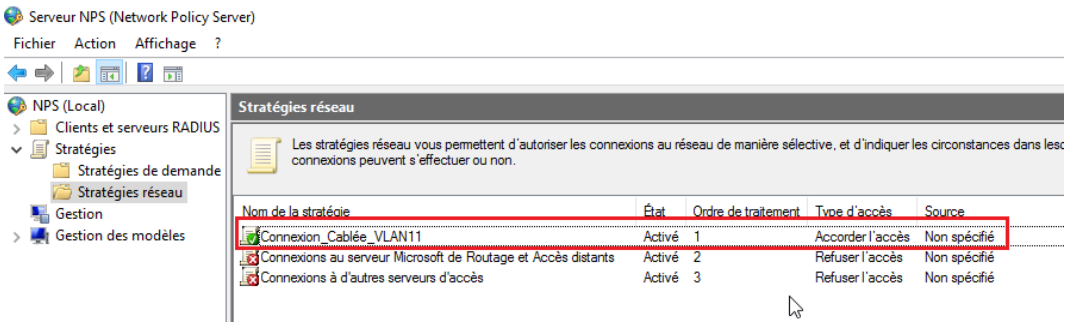
On va maintenant ajouter des attributs de contrôle de trafic en cliquant sur « Ajouter ». Dans notre objectif d'affectation dynamique de VLAN, on va modifier les attributs Tunnel-Type, Tunnel- Medium-Type et Tunnel-Pvt-Group-ID qui vont être envoyés au client Radius pour qu'il réalise l'affectation dynamique de VLAN :



Vérifiez votre configuration avec la console récapitulative de la stratégie et valider si cela vous convient :



Ce qui donnera :



Conclusion

On a donc défini une stratégie d'accès réseau Ethernet 802.1x plaçant dans le VLAN11 les membres authentifiés comme faisant partie du groupe "Carbouffe_RH", vérifiée par la collaboration du serveur NPS avec le client RADIUS, commutateur d'IP 172.18.154.79.

Pour cela le mécanisme de mise en place par NPS a été le suivant :

- Stratégie de demande de connexion associée aux connexions filaire Ethernet
- Stratégie réseau (accès au réseau)
- Précisant le groupe autorisé (Carbouffe_RH) → notion de conditions
- Précisant la méthode d'authentification (PEAP/MSCHAPV2) → notion de propriété
- Associant avec la stratégie de demande de connexion (NAS Ethernet) → notion de contrainte
- Précisant les paramètres renvoyés au client radius (VLAN) → notion d'attributs

Mode Opératoire Serveur Web sous Debian12

Dans ce mode opératoire, nous allons explorer la configuration d'un serveur Web "LAMP" sous Debian 12. Ce serveur sera prêt à héberger divers types de contenus tels que des sites Internet ou des applications.

Le terme "LAMP" fait référence à un ensemble de logiciels essentiels pour créer un serveur Web robuste. Il se compose de quatre composants principaux :

- **L pour Linux** : Cela désigne le système d'exploitation sur lequel le serveur est installé. Dans notre cas, nous utiliserons Debian 11.
- **A pour Apache** : Il s'agit du serveur Web qui gère les requêtes HTTP et sert les fichiers Web aux utilisateurs.
- **M pour MySQL/MariaDB** : Ce composant est un système de gestion de base de données relationnelle. Il stocke et organise les données nécessaires au fonctionnement des sites Web et des applications.
- **P pour PHP** : PHP est un langage de script côté serveur largement utilisé pour générer des contenus dynamiques sur les sites Web. Il est souvent utilisé en combinaison avec des bases de données pour créer des applications Web interactives.

En combinant ces quatre éléments, nous mettrons en place un environnement stable et fonctionnel pour répondre aux besoins d'hébergement du laboratoire.

Installation du serveur Apache

La première étape est la mise à jour des paquets. La commande « **apt update && apt upgrade** » permet de mettre à jour la liste des paquets disponibles et à installer les mises à jour disponibles pour les paquets installés sur un système Debian. Mettre à jour les paquets disponibles est essentiel pour garantir la sécurité, la stabilité et les performances du système. Les mises à jour peuvent inclure des correctifs de sécurité pour protéger le système contre les vulnérabilités connues, des améliorations de fonctionnalités pour optimiser les performances, ainsi que des corrections de bogues pour garantir la stabilité du système. En maintenant les logiciels à jour, on réduit les risques de failles de sécurité et on assure le bon fonctionnement global du système. Pour commencer, on va passer en mode super utilisateur (root) :

```
adm_n.wai-lune@webblpe5:~$ su -
Mot de passe :
root@webblpe5:~#
```

Utiliser l'utilisateur **root** plutôt que « sudo » est préférable dans certains cas car cela donne un accès direct et complet au système sans les restrictions de privilèges imposées par « sudo », mais cela nécessite une prudence accrue pour éviter les actions accidentelles ou malveillantes qui pourraient endommager le système.

Maintenant on peut utiliser la commande évoquer précédemment :

```
root@webblpe5:~# apt update && apt upgrade
```

Ensuite, on installe le paquet « apache2 » avec le paramètre « -y » pour une installation automatique et silencieuse :

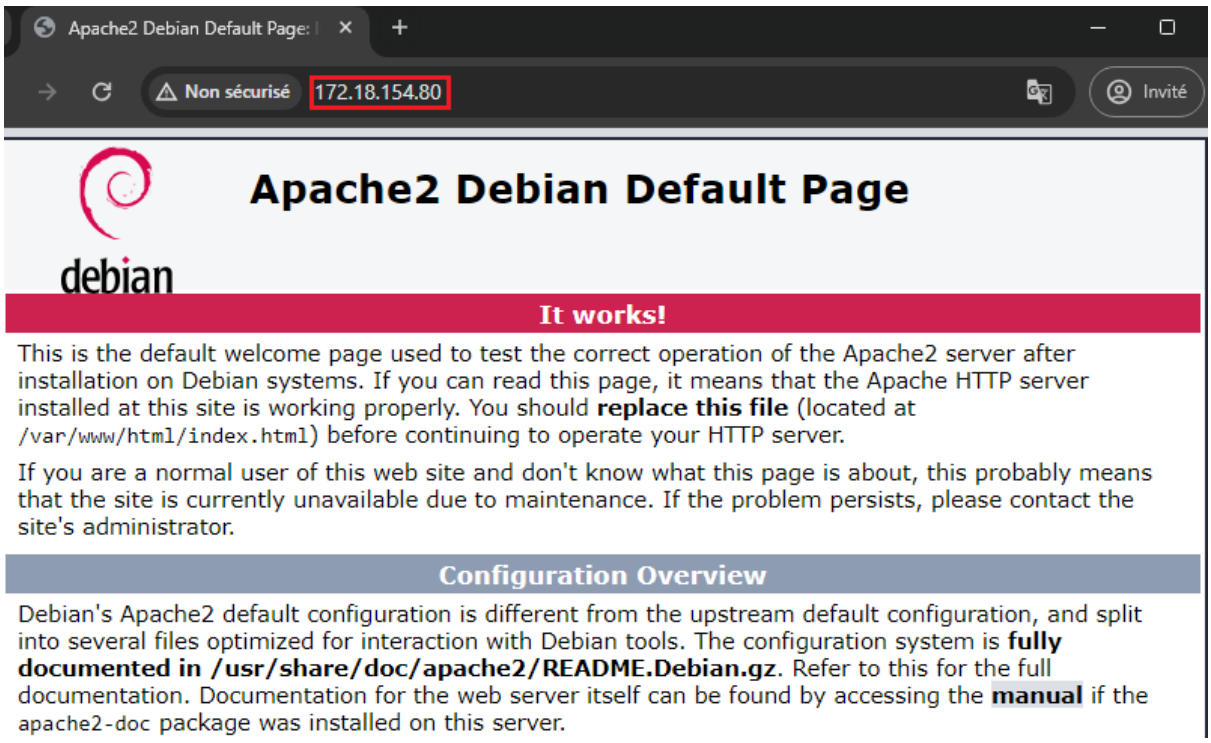
```
root@webblpe5:~# apt install apache2 -y
```

On vient utiliser la commande « systemctl enable apache2 ». Cette dernière configure le système pour démarrer automatiquement le service Apache (apache2) au démarrage du système. Cela garantit que le serveur web sera démarré dès que le système sera lancé, assurant ainsi sa disponibilité plus ou moins continue :

```
root@webblpe5:~# systemctl enable apache2
Synchronizing state of apache2.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable apache2
```

On peut d'or et déjà accéder à la page par défaut d'Apache. Pour cela, il suffit de récupérer l'adresse IP du serveur et d'y accéder par un navigateur Web. Par exemple :

```
root@webblpe5:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
2: ens192: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 00:50:56:a2:07:25 brd ff:ff:ff:ff:ff:ff
    altname enp11s0
    inet 172.18.154.80/21 brd 172.18.159.255 scope global ens192
       valid_lft forever preferred_lft forever
```



Il faut maintenant activer quelques modules d'Apache indispensables pour faire tourner un site Internet. Ces modules sont les suivants :

```
root@webblpe5:~# a2enmod rewrite
Enabling module rewrite.
To activate the new configuration, you need to run:
  systemctl restart apache2
```

```
root@webblpe5:~# a2enmod deflate
Considering dependency filter for deflate:
Module filter already enabled
Module deflate already enabled
```

```
root@webblpe5:~# a2enmod headers
Enabling module headers.
To activate the new configuration, you need to run:
  systemctl restart apache2
```

```
root@webblpe5:~# a2enmod ssl
Considering dependency setenvif for ssl:
Module setenvif already enabled
Considering dependency mime for ssl:
Module mime already enabled
Considering dependency socache_shmcb for ssl:
Module socache_shmcb already enabled
Module ssl already enabled
```

La commande « **a2enmode** » active les modules Apache « rewrite », « deflate », « headers » et « ssl ». Ces modules ajoutent des fonctionnalités supplémentaires au serveur Apache, telles que :

- **rewrite** : Réécrit les URLs, souvent utilisé pour la gestion des permaliens et les redirections.
- **deflate** : Compresse le contenu pour améliorer les performances en réduisant la taille des fichiers envoyés au client.
- **headers** : Permet la manipulation des en-têtes HTTP, utile pour configurer la sécurité, le cache et les contrôles d'accès.
- **ssl** : Fournit la prise en charge du protocole SSL/TLS pour des connexions sécurisées, permettant la configuration et la gestion des certificats SSL/TLS pour assurer la confidentialité et l'intégrité des données.

Après avoir activé ou désactivé un module, ou modifié la configuration d'Apache, il faut toujours redémarrer/rechargé le service apache2 :

```
root@webblpe5:~# systemctl restart apache2.service
```

On va également venir installer le paquet « apache2-utils » car ce dernier offre divers outils utiles pour la gestion, le dépannage et l'évaluation des performances du serveur Apache et notamment la gestion des fichiers de mots de passe utilisés pour l'authentification **HTTP** de base :

```
root@webblpe5:~# apt install apache2-utils -y
```

Installation du serveur MariaDB

MariaDB est une alternative populaire à MySQL, résultant d'un fork communautaire. L'avantage principal de MariaDB réside dans son statut open source et sa licence GPL, offrant ainsi une transparence et une liberté d'utilisation que MySQL, propriétaire chez Oracle, ne peut garantir malgré sa gratuité. MariaDB bénéficie d'un suivi de développement actif et d'une communauté engagée, ce qui en fait un système robuste et performant. Pour installer MariaDB sur notre serveur, on utilise la commande suivante :

```
root@webblpe5:~# apt install mariadb-server -y
```

Ensuite, une bonne pratique est d'exécuter le script « mariadb-secure-installation » afin de garantir une configuration de base sécurisée de MariaDB dès son installation, réduisant ainsi les risques potentiels liés à des paramètres par défaut moins sécurisés. Le script "mariadb-secure-installation" est un utilitaire fourni avec MariaDB qui permet de sécuriser une installation fraîche de MariaDB en suivant plusieurs étapes. Ces étapes incluent généralement :

1. Définition d'un nouveau mot de passe pour le compte "root" de la base de données.
2. Suppression des comptes d'utilisateurs anonymes.
3. Désactivation de la connexion root à distance pour des raisons de sécurité.
4. Suppression de la base de données de test, qui est généralement utilisée pour les tests de développement.
5. Rechargement des privilèges pour s'assurer que les modifications prennent effet immédiatement.

Pour l'exécuter on utilise la commande suivante :

```
root@webblpe5:~# mariadb-secure-installation
```

Il est ensuite possible de configurer les différents éléments énoncer précédemment. Pour cela, il faut répondre aux questions comme ci-dessous :

```
NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
```

```
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

You already have your root account protected, so you can safely answer 'n'.

Switch to unix_socket authentication [Y/n] n
```

```
You already have your root account protected, so you can safely answer 'n'.

Change the root password? [Y/n] n
```

```
By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] y
```

```
Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] y
```

```
By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] y
```

```
Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] y
```

```
All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
```

Pour se connecter à l'instance MariaDB, on utilise la syntaxe « **mariadb -u user -p** ». Le mot de passe de « user » vous sera ensuite demander afin de rentrer dans la console MariaDB, c'est là où vous pourrez exécuter vos requêtes SQL. S'il on est connecter avec le super utilisateur on peut y accéder simplement comme ci-dessous :

```
root@webblpe5:~# mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 38
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.
```

Installation de PHP

Maintenant, on va venir installer le langage de programmation PHP sur notre serveur. PHP est un langage de script côté serveur largement utilisé pour développer des applications web dynamiques. Une fois installé, PHP permet au serveur web d'interpréter et d'exécuter des scripts PHP, ce qui permet de créer des sites web interactifs et dynamiques. Ce dernier va venir se greffer sur notre serveur Apache, comme une extension, afin de pouvoir traiter les scripts intégrés aux pages «***.php*** **»**. Pour ce faire, on utilise la commande suivante :

```
root@webblpe5:~# apt install php -y
```

On peut vérifier la version de PHP qui vient d'être installer avec la commande suivante :

```
root@webblpe5:~# php -v
PHP 8.2.18 (cli) (built: Apr 11 2024 22:07:45) (NTS)
Copyright (c) The PHP Group
Zend Engine v4.2.18, Copyright (c) Zend Technologies
    with Zend OPcache v8.2.18, Copyright (c), by Zend Technologies
```

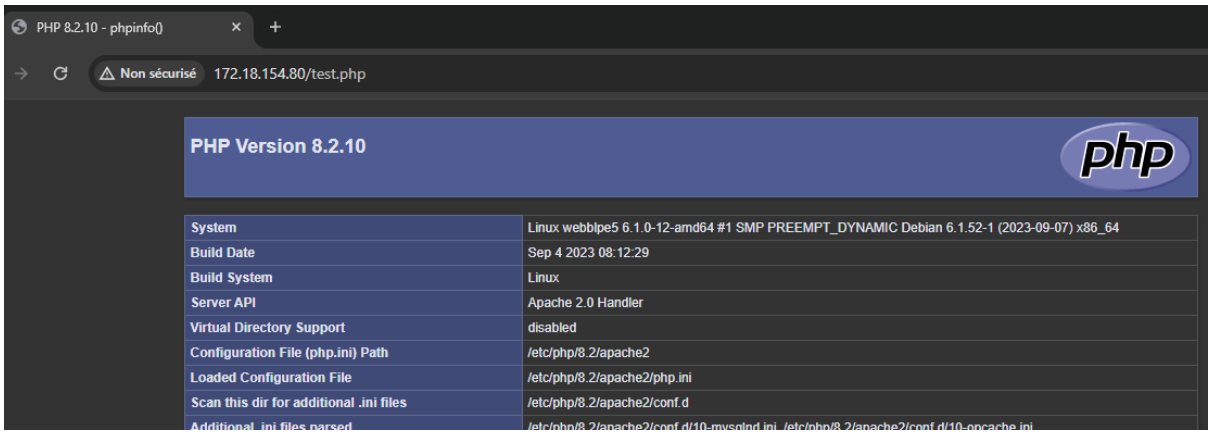
On va venir tester que notre moteur de script PHP est bien opérationnel en créant un fichier « **test.php »** dans le répertoire suivant : (nvim est un éditeur de texte tout comme nano utilisés dans les environnements Unix/Linux)

```
root@webblpe5:~# nvim /var/www/html/test.php
```

On va y définir la fonction « **phpinfo()** » qui génère et affiche des informations détaillées sur la configuration de PHP installée sur un serveur. Lorsque vous appelez cette fonction dans un script PHP et exécutez ce script dans un navigateur, vous obtenez une page HTML détaillant divers aspects de la configuration PHP, tels que les paramètres du serveur, les modules activés, les versions des logiciels, les chemins d'accès, les directives de configuration, etc :

```
<?php phpinfo(); ?>
```

Ce qui donne, en rajoutant « **/test.php** » à l'URL de la précédente page de notre navigateur :



La page générer fournit une multitude d'informations détaillées sur la configuration de PHP ainsi que sur le serveur Apache. Cependant, son accès devrait être restreint aux moments où ces données sont nécessaires. En d'autres termes, il est crucial de ne pas laisser cette page accessible à tout le monde, car elle peut contenir des informations sensibles sur votre configuration serveur. Il est donc recommandé de restreindre l'accès à cette page uniquement aux utilisateurs autorisés, afin de prévenir toute exposition non désirée de données sensibles.

Conclusion

En résumé, la mise en place réussie d'un environnement de serveur LAMP (Linux, Apache, MySQL/MariaDB, PHP) sur un système Debian constitue une étape cruciale dans la création d'une plateforme web robuste. Cette configuration permet désormais de fournir des services web, d'héberger des applications et de gérer des bases de données de manière sécurisée et efficace. Avec Linux comme fondation, Apache comme serveur web, MariaDB pour la gestion des données, et PHP pour la dynamique des applications web, ce serveur LAMP est parfaitement équipé pour répondre aux exigences de futures projets web, tout en offrant une flexibilité et une fiabilité plus ou moins optimales.

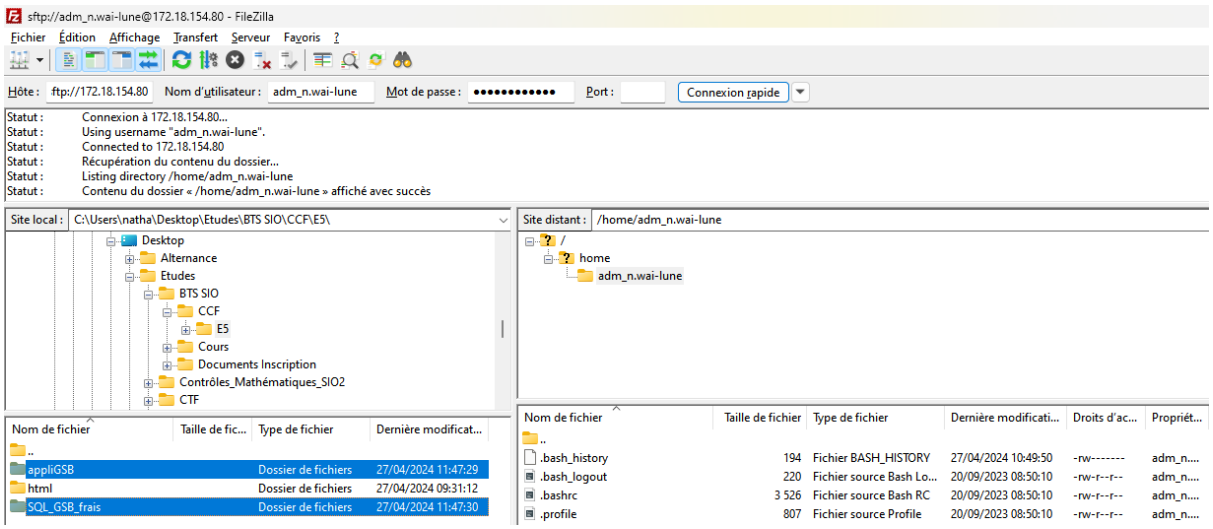
Mode Opératoire Virtual Host

Maintenant que nous avons réussi à mettre en place un environnement de serveur LAMP (Linux, Apache, MySQL/MariaDB, PHP) sur notre système Debian, nous sommes prêts à explorer une fonctionnalité essentielle pour la gestion des sites web : la configuration des Virtual hosts. Mais un Virtual host qu'est-ce que c'est ? Un Virtual host, ou hôte virtuel, est une méthode utilisée par le serveur web Apache pour héberger plusieurs sites web sur une seule machine physique. Chaque Virtual host peut avoir sa propre configuration distincte, permettant ainsi à plusieurs sites web de coexister sur le même serveur tout en étant isolés les uns des autres. Cette isolation permet aux administrateurs système de gérer efficacement plusieurs sites web avec des configurations et des contenus différents sur une seule infrastructure serveur.

Nous configurerons un hôte virtuel la solution Web ***gestionfraisintranet.gsb*** du laboratoire Galaxy Swiss Bourdin.

Il convient de noter qu'Apache est initialement configuré avec un hôte virtuel par défaut, qui pointe vers le répertoire « /var/www/html ». Bien que cela serve de point de départ pratique, cette configuration de base s'avère limitée pour héberger plusieurs sites web.

Dans des circonstances habituelles, la création d'un répertoire dans « /var/www » aurait été nécessaire. Cependant, dans ce contexte, ce répertoire m'a été fourni. Si vous êtes dans une situation similaire où vous avez déjà les dossiers de votre site, vous pouvez les importer en utilisant un client FTP tel que **FileZilla** :



Et les déplacer dans le bon répertoire :

```
root@webblpe5:~# ls -rtl
total 8
drwxr-xr-x 2 adm_n.wai-lune adm_n.wai-lune 4096 27 avril 12:08 SQL_GSB_frais
drwxr-xr-x 5 adm_n.wai-lune adm_n.wai-lune 4096 27 avril 12:08 appliGSB
```

```
root@webblpe5:~# mv /root/appliGSB/ /var/www/
```

Nous allons ultérieurement utiliser les fichiers situés dans le répertoire « **/root/SQL_GSB_frais** ». Ces fichiers sont essentiels au bon fonctionnement de la solution Web.

⚠ Je les ai dans le cadre de la mise en place de ma solution. Si vous suivez ce mode opératoire, à moins d'être dans le même contexte que moi, vous n'en aurez pas besoin ⚠

Pour créer l'hôte virtuel, on se déplace dans le répertoire « **/etc/apache2/sites-available** » comme ci-dessous :

```
root@webblpe5:~# cd /etc/apache2/sites-available/
root@webblpe5:/etc/apache2/sites-available# ls -rtl
total 12
-rw-r--r-- 1 root root 1286 27 avril 09:48 000-default.conf
-rw-r--r-- 1 root root 6195 27 avril 09:48 default-ssl.conf
```

Tous les nouveaux sites doivent être créer dans ce répertoire. Nous établirons un lien symbolique par la suite avec le répertoire « **/etc/apache2/sites-enabled** ». Pour créer un nouveau fichier de configuration on utilise la commande suivante :

```
root@webblpe5:/etc/apache2/sites-available# nvim gestionfraisintranet.gsb.conf
```

Voici un exemple de configuration d'un Virtual host Apache :

```
<VirtualHost *:80>
   ServerName example.com
   DocumentRoot /var/www/example
</VirtualHost>
```

Pour ma part je ne vais pas modifier le fichier que je viens de créer, du moins pour l'instant. La solution que je dois mettre en place doit utiliser le protocole HTTPS car ce dernier offre une sécurité supplémentaire par rapport à HTTP en chiffrant les données lors de leur transfert, ce qui les rend appropriées pour les sites Web nécessitant une protection des données sensibles. Pour cela, je vais créer un répertoire avec des permissions spécifiques et un propriétaire spécifique (vous pouvez évidemment faire de même) :

```
root@webblpe5:~# mkdir -p /etc/apache2/ssl
root@webblpe5:~# chmod 700 /etc/apache2/ssl/
root@webblpe5:~# chown -R root:root /etc/apache2/ssl/
```

Ces commandes sont utilisées pour créer un répertoire sécurisé et garantir que les fichiers de certificat et de clé privée sont accessibles uniquement par l'utilisateur et le groupe appropriés, tout en sécurisant les permissions pour prévenir les accès non autorisés.

Je vais ensuite générer une nouvelle clé privée RSA. En effet, une clé RSA est utilisée dans le processus SSL/TLS pour chiffrer les données, garantir leur intégrité et authentifier les parties impliquées dans la communication sécurisée sur Internet. Dans mon cas, j'utilise la commande ci-dessous pour générer une nouvelle clé privée RSA de 2048 bits et l'écrire dans le fichier spécifié :

```
root@webblpe5:~# openssl genrsa -out /etc/apache2/ssl/server.key 2048
```

Je fais ensuite une demande de signature de certificat (CSR - Certificate Signing Request), nécessaire pour obtenir un certificat SSL/TLS signé par une autorité de certification (CA). CSR est une demande formelle envoyée à une autorité de certification pour obtenir un certificat SSL/TLS. Elle contient des informations sur l'entité demandant le certificat ainsi que la clé publique correspondant à la clé privée qui sera utilisée pour chiffrer les données. Une fois que la CSR est signée par la CA, elle devient un certificat SSL/TLS valide qui peut être utilisé pour sécuriser les communications sur le site Web. Pour cela j'utilise la commande suivante et je complète le questionnaire :

```
root@webblpe5:~# openssl req -new -key /etc/apache2/ssl/server.key -out /etc/apache2/ssl/server.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:
```

```
Country Name (2 letter code) [AU]:FR
State or Province Name (full name) [Some-State]:Réunion
Locality Name (eg, city) []:Saint-Denis
Organization Name (eg, company) [Internet Widgits Pty Ltd]:Galaxy Swiss Bourdin
Organizational Unit Name (eg, section) []:GSB
Common Name (e.g. server FQDN or YOUR name) []:GSB
Email Address []:
```

```
Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:
An optional company name []:
```

Maintenant, je génère le certificat auto-signé. Générer un certificat SSL auto-signé peut être utile pour des besoins de développement, de test ou dans des environnements isolés où la confiance peut être établie différemment. Cependant, il est important de comprendre que les certificats auto-signés ne fournissent pas le même niveau de confiance et de sécurité que les certificats émis par une autorité de certification publique. Ils ne sont donc pas recommandés pour une utilisation en production sur des sites Web accessibles au public ou traitant des informations sensibles. Ils peuvent être utiles pour des cas d'utilisation spécifiques, mais leur utilisation doit être soigneusement évaluée en fonction des exigences de sécurité et de confiance de votre application ou service.

Pour générer le certificat, j'utilise la commande suivante :

```
root@webblpe5:~# openssl x509 -req -days 365 -in /etc/apache2/ssl/server.csr -signkey /etc/apache2/ssl/server.key -out /etc/apache2/ssl/server.crt
Certificate request self-signature ok
subject=C = FR, ST = R\C3\83\C2\A9union, L = Saint-Denis, O = Galaxy Swiss Bourdin, OU = GSB, CN = GSB
```

Je configure mes Virtuals Hosts de la manière suivante :

```
<VirtualHost *:80>
    ServerName gestionfraisintranet.gsb
    Redirect permanent / https://gestionfraisintranet.gsb
</VirtualHost>

<VirtualHost *:443>
    # The ServerName directive sets the request scheme, hostname and port that
    # the server uses to identify itself. This is used when creating
    # redirection URLs. In the context of virtual hosts, the ServerName
    # specifies what hostname must appear in the request's Host: header to
    # match this virtual host. For the default virtual host (this file) this
    # value is not decisive as it is used as a last resort host regardless.
    # However, you must set it for any further virtual host explicitly.
    #ServerName www.example.com

    ServerAdmin webmaster@localhost
    ServerName gestionfraisintranet.gsb
    DocumentRoot /var/www/appliGSB

    <Directory /var/www/appliGSB>
        Options Indexes FollowSymLinks
        AllowOverride All
        Require all granted
        DirectoryIndex cAccueil.php
    </Directory>

    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/server.crt
    SSLCertificateKeyFile /etc/apache2/ssl/server.key

    # Available loglevels: trace8, ..., trace1, debug, info, notice, warn,
    # error, crit, alert, emerg.
    # It is also possible to configure the loglevel for particular
    # modules, e.g.
    #LogLevel info ssl:warn

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

    # For most configuration files from conf-available/, which are
    # enabled or disabled at a global level, it is possible to
    # include a line for only one particular virtual host. For example the
    # following line enables the CGI configuration for this host only
    # after it has been globally disabled with "a2disconf".
    #Include conf-available/serve-cgi-bin.conf
</VirtualHost>
```

```
<VirtualHost 172.18.154.71:80>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/appliGSB

        <Directory /var/www/appliGSB>
                Options Indexes FollowSymLinks
                AllowOverride All
                Require all granted
                DirectoryIndex cAccueil.php
        </Directory>

        RewriteEngine On
        RewriteCond %{HTTPS} off
        RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>

<VirtualHost 172.18.154.71:443>
        ServerAdmin webmaster@localhost
        DocumentRoot /var/www/appliGSB

        <Directory /var/www/appliGSB>
                Options Indexes FollowSymLinks
                AllowOverride All
                Require all granted
                DirectoryIndex cAccueil.php
        </Directory>

        SSLEngine on
        SSLCertificateFile /etc/apache2/ssl/server.crt
        SSLCertificateKeyFile /etc/apache2/ssl/server.key

        ErrorLog ${APACHE_LOG_DIR}/error.log
        CustomLog ${APACHE_LOG_DIR}/access.log combined

</VirtualHost>
```

Le premier Virtual Host admet configuration Apache qui définit un hôte virtuel pour le trafic HTTP (port 80) du domaine **gestionfraisintranet.gsb**. Voici ce que chaque ligne de cette configuration fait :

- **<VirtualHost *:80>** : Déclare un hôte virtuel pour le trafic HTTP sur toutes les interfaces (*) et le port 80.
- **ServerName gestionfraisintranet.gsb** : Spécifie le nom du serveur pour cet hôte virtuel, dans ce cas, gestionfraisintranet.gsb.
- **Redirect permanent / https://gestionfraisintranet.gsb** : Cette directive indique à Apache d'effectuer une redirection permanente (statut 301) pour tout le trafic arrivant sur cet hôte virtuel. La redirection envoie le navigateur du client vers l'URL https://gestionfraisintranet.gsb, ce qui signifie que tout le trafic HTTP est automatiquement redirigé vers HTTPS pour une connexion sécurisée.

Le second admet une configuration Apache qui définit un hôte virtuel pour le trafic HTTPS (port 443) du domaine **gestionfraisintranet.gsb**. Voici une explication de chaque élément de cette configuration :

- **<VirtualHost *:443>** : Déclare un hôte virtuel pour le trafic HTTPS sur toutes les interfaces (`*`) et le port 443.
- **ServerName gestionfraisintranet.gsb** : Spécifie le nom du serveur pour cet hôte virtuel, dans ce cas, gestionfraisintranet.gsb.
- **ServerAdmin webmaster@localhost** : Définit l'adresse e-mail de l'administrateur du serveur.
- **DocumentRoot /var/www/appliGSB** : Définit le répertoire racine du site Web, c'est-à-dire l'emplacement où les fichiers du site sont stockés.
- **<Directory /var/www/appliGSB>** : Configure les options spécifiques au répertoire, telles que les permissions et les autorisations d'accès. Dans ce cas, cela permet l'accès au répertoire « /var/www/appliGSB » et spécifie « cAccueil.php » comme fichier index par défaut.
- **SSLEngine on** : Active le moteur SSL pour ce virtualhost, indiquant qu'il doit gérer le trafic HTTPS.
- **SSLCertificateFile /etc/apache2/ssl/server.crt** : Spécifie le chemin du fichier de certificat SSL utilisé pour ce virtualhost.
- **SSLCertificateKeyFile /etc/apache2/ssl/server.key** : Spécifie le chemin du fichier de clé privée correspondant au certificat SSL.
- **ErrorLog ${APACHE_LOG_DIR}/error.log** : Spécifie le fichier de journal des erreurs pour cet hôte virtuel.
- **CustomLog ${APACHE_LOG_DIR}/access.log combined** : Spécifie le fichier de journal des accès pour cet hôte virtuel.

Le troisième admet une configuration Apache qui définit un hôte virtuel pour le trafic HTTP (port 80) provenant de l'adresse IP **172.18.154.71** (hébergeur). Voici une explication de chaque élément de cette configuration :

- **<VirtualHost 172.18.154.71:80>** : Déclare un hôte virtuel pour le trafic HTTP sur l'adresse IP 172.18.154.71 et le port 80.
- **ServerAdmin webmaster@localhost** : Définit l'adresse e-mail de l'administrateur du serveur.
- **DocumentRoot /var/www/appliGSB** : Définit le répertoire racine du site Web, c'est-à-dire l'emplacement où les fichiers du site sont stockés.
- **<Directory /var/www/appliGSB>** : Configure les options spécifiques au répertoire, telles que les permissions et les autorisations d'accès. Dans ce cas, cela permet l'accès au répertoire « /var/www/appliGSB » et spécifie « cAccueil.php » comme fichier index par défaut.
- **RewriteEngine On** : Active le module de réécriture d'URL pour ce virtualhost.
- **RewriteCond %{HTTPS} off** : Définit une condition pour la règle de réécriture suivante : si la connexion n'est pas sécurisée (HTTPS est désactivé).
- **RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI} [L,R=301]** : Redirige toutes les requêtes HTTP vers HTTPS en utilisant une redirection permanente (statut 301). Cela garantit que tout le trafic HTTP est automatiquement redirigé vers HTTPS pour une connexion sécurisée.
- **ErrorLog ${APACHE_LOG_DIR}/error.log** : Spécifie le fichier de journal des erreurs pour cet hôte virtuel.
- **CustomLog ${APACHE_LOG_DIR}/access.log combined** : Spécifie le fichier de journal des accès pour cet hôte virtuel.

Le quatrième admet une configuration Apache qui définit un hôte virtuel pour le trafic HTTPS (port 443) provenant de l'adresse IP **172.18.154.71**. Voici une explication de chaque élément de cette configuration :

- **<VirtualHost 172.18.154.71:443>** : Déclare un hôte virtuel pour le trafic HTTPS sur l'adresse IP 172.18.154.71 et le port 443.
- **ServerAdmin webmaster@localhost** : Définit l'adresse e-mail de l'administrateur du serveur.
- **DocumentRoot /var/www/appliGSB** : Définit le répertoire racine du site Web, c'est-à-dire l'emplacement où les fichiers du site sont stockés.
- **<Directory /var/www/appliGSB>** : Configure les options spécifiques au répertoire, telles que les permissions et les autorisations d'accès. Dans ce cas, cela permet l'accès au répertoire « /var/www/appliGSB » et spécifie « cAccueil.php » comme fichier index par défaut.
- **SSLEngine on** : Active le moteur SSL pour ce virtualhost, indiquant qu'il doit gérer le trafic HTTPS.
- **SSLCertificateFile /etc/apache2/ssl/server.crt** : Spécifie le chemin du fichier de certificat SSL utilisé pour ce virtualhost.
- **SSLCertificateKeyFile /etc/apache2/ssl/server.key** : Spécifie le chemin du fichier de clé privée correspondant au certificat SSL.

- **ErrorLog ${APACHE_LOG_DIR}/error.log** : Spécifie le fichier de journal des erreurs pour cet hôte virtuel.
- **CustomLog ${APACHE_LOG_DIR}/access.log combined** : Spécifie le fichier de journal des accès pour cet hôte virtuel.

Vous pouvez ensuite enregistrer votre fichier de configuration et utiliser la commande suivante pour simplifier le processus d'activation des sites web sur Apache en créant les liens symboliques appropriés et en permettant à Apache de charger les fichiers de configuration des sites web lors de son démarrage :

```
root@webblpe5:/etc/apache2/sites-available# ls -rtl
total 16
-rw-r--r-- 1 root root 1286 27 avril 09:48 000-default.conf
-rw-r--r-- 1 root root 6195 27 avril 09:48 default-ssl.conf
-rw-r--r-- 1 root root 1837 27 avril 12:58 gestionfraisintranet.gsb.conf
```

```
root@webblpe5:/etc/apache2/sites-available# a2ensite gestionfraisintranet.gsb.conf
Enabling site gestionfraisintranet.gsb.
To activate the new configuration, you need to run:
  systemctl reload apache2
```

Assurez-vous de recharger le service Apache2. Maintenant, vous pouvez voire votre fichier dans le répertoire « /etc/apache2/sites-enabled » :

```
root@webblpe5:~# cd /etc/apache2/sites-enabled/
root@webblpe5:/etc/apache2/sites-enabled# ls -rtl
total 0
lrwxrwxrwx 1 root root 35  2 oct.   2023 000-default.conf -> ../sites-available/000-default.conf
lrwxrwxrwx 1 root root 35  2 oct.   2023 default-ssl.conf -> ../sites-available/default-ssl.conf
lrwxrwxrwx 1 root root 48 27 avril 13:20 gestionfraisintranet.gsb.conf -> ../sites-available/gestionfraisintranet.gsb.conf
```

Importation fichier SQL

Je vais maintenant importer les fichiers présents dans le répertoire « **/root/SQL_GSB_frais** » sur mon serveur MariaDB. L'importation de fichiers SQL est un outil essentiel pour gérer efficacement les données dans une base de données, que ce soit pour la migration, la sauvegarde, la synchronisation ou le déploiement de votre application. Pour cela, j'utilise la commande suivante :

```
root@webblpe5:~# mysql -u votre_utilisateur -p votre_base_de_donnees < /chemin/vers/votre/fichier.sql
```

Il faudra évidemment remplacez « votre_utilisateur » par votre nom d'utilisateur MySQL, « votre_base_de_donnees » par le nom de votre base de données et « /chemin/vers/votre/fichier.sql » par le chemin du fichier SQL que vous souhaitez importer. Dans mon cas, j'ai utilisé les commandes suivantes :

```
root@webblpe5:~# ls -rtl
total 4
drwxr-xr-x 2 adm_n.wai-lune adm_n.wai-lune 4096 27 avril 12:08 SQL_GSB_frais
```

```
root@webblpe5:~# cd SQL_GSB_frais/
root@webblpe5:~/SQL_GSB_frais# ls -rtl
total 8
-rw-r--r-- 1 adm_n.wai-lune adm_n.wai-lune 3340 27 avril 12:08 gsb_frais_structure.sql
-rw-r--r-- 1 adm_n.wai-lune adm_n.wai-lune 3745 27 avril 12:08 gsb_frais_insert_tables_statiques.sql
```

```
root@webblpe5:~/SQL_GSB_frais# mysql -u root < /root/SQL_GSB_frais/gsb_frais_structure.sql
```

```
root@webblpe5:~/SQL_GSB_frais# mysql -u root < /root/SQL_GSB_frais/gsb_frais_insert_tables_statiques.sql
```

Je peux vérifier si l'importation c'est bien effectuer via les commandes suivantes :

```
root@webblpe5:~# mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 41
Server version: 10.11.6-MariaDB-0+deb12u1 Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| gsb_frais          |
| information_schema |
| mysql              |
| ocs                |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0,001 sec)
```

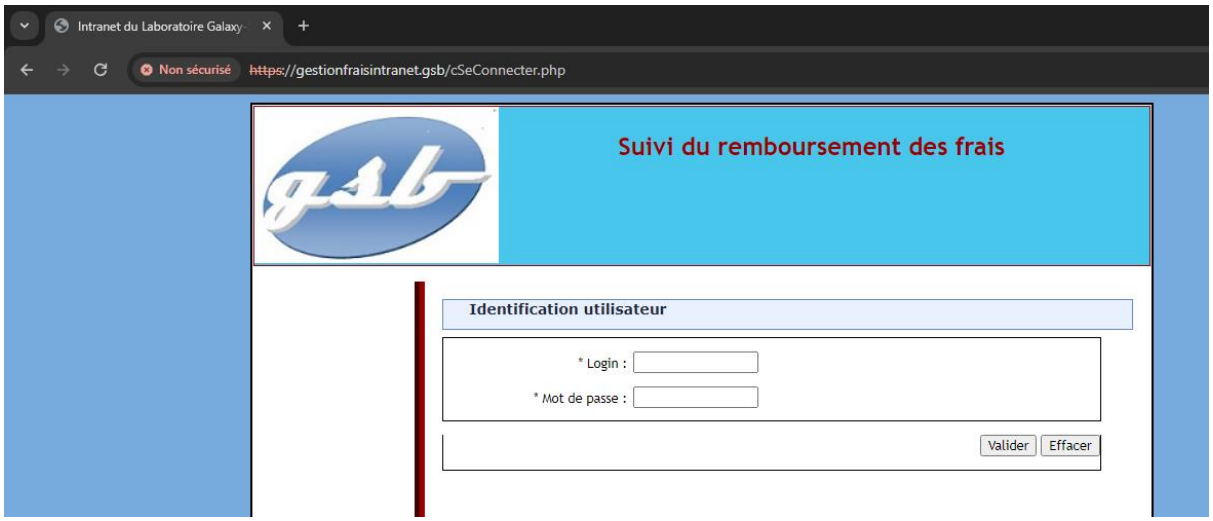
```
MariaDB [(none)]> USE gsb_frais;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [gsb_frais]> SHOW TABLES;
+------------------------+
| Tables_in_gsb_frais    |
+------------------------+
| Etat                   |
| FicheFrais             |
| FraisForfait           |
| LigneFraisForfait      |
| LigneFraisHorsForfait  |
| Visiteur               |
+------------------------+
6 rows in set (0,001 sec)
```

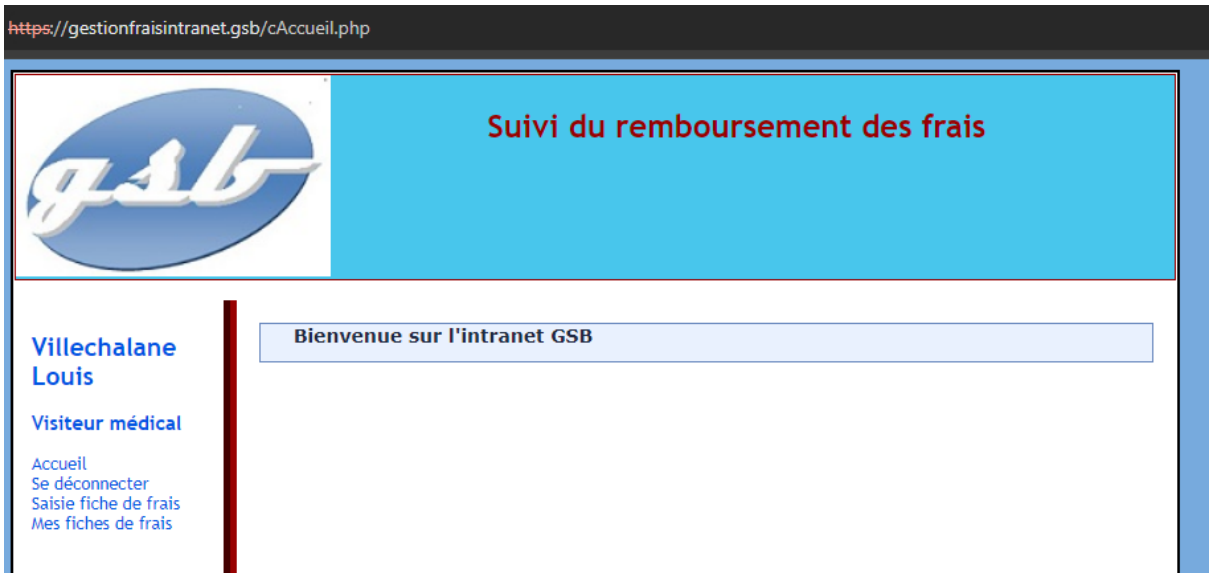
```
MariaDB [gsb_frais]> SELECT login, mdp from Visiteur;
+--------------+-------+
| login        | mdp   |
+--------------+-------+
| lvillachane  | jux7g |
```

Avant de tester l'accès à la solution Web :

```
root@webblpe5:~# systemctl restart mariadb.service
root@webblpe5:~# systemctl restart apache2.service
```

Ce qui donne :





Conclusion

En conclusion, la mise en place réussie d'un virtual host dans Apache constitue une étape cruciale pour rendre une solution web accessible de manière fiable et sécurisée. Grâce à cette configuration, le serveur est en mesure de diriger le trafic vers le bon répertoire de contenu, offrant ainsi aux utilisateurs une expérience en ligne fluide et optimale.

Mode Opératoire Haute disponibilité

Ce mode opératoire décrit la mise en œuvre d'une infrastructure à haute disponibilité pour une solution Web, en utilisant un cluster maitre-maitre basé sur les technologies Corosync et Pacemaker, installé sur des systèmes Debian 12 (dans mon cas 2 serveurs) déjà configurer en tant que serveur LAMP.

Un cluster est un ensemble de serveurs, appelés nœuds, qui travaillent collectivement pour améliorer la disponibilité, la fiabilité, et les performances des applications. Cette configuration est essentielle pour assurer une continuité de service, car en cas de défaillance d'un nœud, les autres peuvent immédiatement prendre le relais pour maintenir les opérations, réduisant ainsi les interruptions. Les clusters utilisent des "heartbeats" ou battements de cœur, qui sont des signaux réguliers envoyés entre les nœuds pour vérifier leur état actif. Corosync joue un rôle crucial dans cette communication, en gérant les échanges d'informations pour maintenir la cohésion et la synchronisation du cluster. Pacemaker complète cette configuration en agissant comme un gestionnaire de ressources qui orchestre le démarrage, l'arrêt, et la surveillance des services sur les nœuds, s'assurant que les applications fonctionnent là où elles sont le plus nécessaires selon les politiques prédéfinies. Ensemble, Corosync et Pacemaker fournissent une solution robuste pour les environnements nécessitant une haute disponibilité et une gestion efficace des failovers.

Quand on gère un service au sein d'un cluster (un groupe de serveurs), cela signifie généralement que le service est installé sur chaque serveur du groupe mais est seulement actif sur un serveur principal, appelé le nœud maître. Si le nœud maître a un problème et ne fonctionne plus, le service démarre automatiquement sur un autre serveur, nommé nœud esclave, qui prend alors le relais. Ce passage du service du nœud maître au nœud esclave s'appelle un basculement. Parfois, une fois le nœud maître réparé, le service peut automatiquement revenir à ce premier serveur, mais cela ne se fait pas tout seul et nécessite une configuration spécifique.

Installation de Corosync et Pacemaker

La première étape est de mettre à jour le système pour s'assurer que tous les paquets installés sont à la dernière version disponible. Cela prépare le système pour l'installation de nouveaux logiciels et garantit que toutes les dépendances nécessaires sont présentes et à jour. On utilise la commande suivante sur les serveurs qui vont faire partie du cluster :

```
root@webblpe5-cl1:~# apt update && apt upgrade
```

```
root@webblpe5-cl2:~# apt update && apt upgrade
```

Une fois le système mis à jour, on installe Corosync et Pacemaker, qui sont essentiels pour la création et la gestion d'un cluster. Ces outils gèrent la disponibilité des services et la communication entre les nœuds. Pour installer ces paquets, on utilise la commande :

```
root@webblpe5-cl1:~# apt install corosync pacemaker crmsh
```

```
root@webblpe5-cl2:~# apt install corosync pacemaker crmsh
```

Après l'installation, il est important de vérifier que Corosync, le moteur du cluster, fonctionne correctement. Cette vérification permet de s'assurer que le logiciel est actif et prêt à gérer la communication entre les nœuds. Utilisez la commande suivante pour vérifier le statut :

```
root@webblpe5-cl1:~# systemctl status corosync.service
● corosync.service - Corosync Cluster Engine
     Loaded: loaded (/lib/systemd/system/corosync.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-04-30 12:40:27 +04; 19h ago
       Docs: man:corosync
             man:corosync.conf
             man:corosync_overview
   Main PID: 7172 (corosync)
      Tasks: 9 (limit: 2307)
     Memory: 136.5M
        CPU: 7min 10.575s
     CGroup: /system.slice/corosync.service
             └─7172 /usr/sbin/corosync -f

avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [SERV  ] Service engine loaded: corosync cluster quorum service v0.1 [3]
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [QB    ] server name: quorum
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [TOTEM ] Configuring link 0
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [TOTEM ] Configured link number 0: local addr: 127.0.0.1, port=5405
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [KNET  ] link: Resetting MTU for link 0 because host 1 joined
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [QUORUM] Sync members[1]: 1
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [QUORUM] Sync joined[1]: 1
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [TOTEM ] A new membership (1.5) was formed. Members joined: 1
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [QUORUM] Members[1]: 1
avril 30 12:40:27 webblpe5-cl1 corosync[7172]:   [MAIN  ] Completed service synchronization, ready to provide service.
```

```
root@webblpe5-cl2:~# systemctl status corosync.service
● corosync.service - Corosync Cluster Engine
     Loaded: loaded (/lib/systemd/system/corosync.service; enabled; preset: enabled)
     Active: active (running) since Tue 2024-04-30 12:39:50 +04; 19h ago
       Docs: man:corosync
             man:corosync.conf
             man:corosync_overview
   Main PID: 52851 (corosync)
      Tasks: 9 (limit: 2307)
     Memory: 136.6M
        CPU: 7min 16.951s
     CGroup: /system.slice/corosync.service
             └─52851 /usr/sbin/corosync -f

avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [SERV  ] Service engine loaded: corosync cluster quorum service v0.1 [3]
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [QB    ] server name: quorum
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [TOTEM ] Configuring link 0
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [TOTEM ] Configured link number 0: local addr: 127.0.0.1, port=5405
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [KNET  ] link: Resetting MTU for link 0 because host 1 joined
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [QUORUM] Sync members[1]: 1
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [QUORUM] Sync joined[1]: 1
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [TOTEM ] A new membership (1.5) was formed. Members joined: 1
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [QUORUM] Members[1]: 1
avril 30 12:39:50 webblpe5-cl2 corosync[52851]:   [MAIN  ] Completed service synchronization, ready to provide service.
```

Corosync utilise une clé d'authentification pour sécuriser les communications entre les nœuds du cluster. Cette étape consiste à créer une clé cryptographique partagée, qui doit être réalisée sur un seul serveur et ensuite copiée sur tous les autres nœuds du cluster pour garantir une communication sécurisée. On génère cette clé avec :

```
root@webblpe5-cl1:~# corosync-keygen
```

Après la génération, on doit copier la clé sur les autres nœuds en utilisant un client FTP ou tout autre moyen sécurisé de transfert de fichiers. On peut voire cette les dernières via la commande suivante :

```
root@webblpe5-cl1:~# ls -l /etc/corosync/
total 16
-r-------- 1 root root  256 30 avril 12:43 authkey
-rw-r--r-- 1 root root 1011 30 avril 13:02 corosync.conf
```

```
root@webblpe5-cl2:~# ls -l /etc/corosync/
total 16
-r-------- 1 root root  256 30 avril 12:44 authkey
-rw-r--r-- 1 root root 1011 30 avril 12:55 corosync.conf
```

Une bonne pratique est de sauvegarder le fichier d'origine et d'en créer un nouveau. Pour cela on utilise les commandes suivantes :

```
root@webblpe5-cl2:~# cd /etc/corosync/
root@webblpe5-cl2:/etc/corosync# mv corosync.conf corosync.conf.ori
```

```
root@webblpe5-cl1:~# cd /etc/corosync/
root@webblpe5-cl1:/etc/corosync# mv corosync.conf corosync.conf.ori
```

```
root@webblpe5-cl1:~# cd /etc/corosync/
root@webblpe5-cl1:/etc/corosync# nvim corosync.conf
```

```
root@webblpe5-cl2:~# cd /etc/corosync/
root@webblpe5-cl2:/etc/corosync# nvim corosync.conf
```

Voici les directives que j'ai mis en place sur mes deux serveurs :

```
totem {
        version: 2
        crypto_cipher: aes256
        crypto_hash: sha1
        clear_node_high_bit: yes
        interface {
                ringnumber: 0
                bindnetaddr: 172.18.152.0
                mcastaddr: 239.255.1.250
                mcastport: 5405
                ttl: 1
        }
        auto_increment_increment: 2
        auto_increment_offset: 1

}
```

Cette section du fichier « **corosync.conf** » configure les paramètres du sous-système « **totem** », qui est au cœur de la communication et de la gestion des membres dans un cluster Corosync. Voici la signification de chaque paramètre dans cette configuration :

- **version** : Spécifie la version du protocole Totem à utiliser. Ici, la version est « **2** », ce qui correspond au protocole utilisé pour la communication et la coordination au sein du cluster.
- **crypto_ciphe** : Définit le type de chiffrement utilisé pour sécuriser les communications entre les nœuds du cluster. « **aes256** » indique que le chiffrement AES 256 bits est utilisé, offrant un haut niveau de sécurité.
- **crypto_hash** : Spécifie l'algorithme de hachage utilisé pour l'intégrité des messages. « **sha1** » est utilisé ici, bien que des options plus sécurisées comme SHA-256 puissent être préférables dans des environnements nécessitant une sécurité accrue
- **clear_node_high_bit** : Ce paramètre, lorsqu'il est défini sur « **yes** », permet d'éviter les conflits d'identifiant de nœud en clairant le bit le plus significatif des adresses IP des nœuds.
- **interface** : Cette section configure les détails de l'interface réseau utilisée pour la communication du cluster :
- **ringnumber** : Identifie le numéro de l'anneau de communication dans le cluster. « **0** » est généralement utilisé pour le premier et souvent unique anneau.
- **bindnetaddr** : Adresse du réseau sur lequel Corosync doit opérer. Ici, « **172.18.152.0** » indique l'adresse réseau sur laquelle le cluster doit communiquer.
- **mcastaddr** : Adresse multicast utilisée pour les communications de cluster. « **239.255.1.250** » est l'adresse spécifiée pour la diffusion des messages à tous les membres du cluster.
- **mcastport** : Port utilisé pour les communications multicast. « **5405** » est le port sur lequel les nœuds écoutent et transmettent les messages multicast.
- **ttl** : Time To Live (TTL) pour les paquets multicast, défini ici à « **1** », ce qui signifie que les paquets ne sont pas destinés à quitter le réseau local.
- **auto_increment_increment = 2** : Dans un environnement de réplication, ce paramètre contrôle l'incrémentation des valeurs auto-incrémentées. Cela aide à éviter les conflits de clés lors de l'insertion de données dans des tables avec des colonnes auto-incrémentées sur différents serveurs.
- **auto_increment_offset = 1/2** : Définit la valeur de départ pour l'incrémentation automatique dans un environnement de réplication maître-maître. Chaque serveur doit avoir un offset différent pour éviter les conflits de clés.

La configuration de « **totem** » est essentielle pour assurer une communication efficace et sécurisée au sein du cluster, et chaque paramètre doit être ajusté en fonction des besoins spécifiques et de l'environnement réseau de votre cluster. Cette configuration est vitale pour assurer une communication efficace et sécurisée au sein du cluster.

La section « **logging** » dans un fichier de configuration Corosync sert à contrôler la manière dont les événements sont enregistrés (log) dans le système.

```
logging {
        fileline: off
        to_logfile: yes
        to_syslog: no
        logfile: /var/log/corosync/corosync.log
        debug: off
        timestamp: on
        logger_subsys {
                subsys: QUORUM
                debug: off
        }
}
```


```
logging {
        fileline: off
        to_logfile: yes
        to_syslog: no
        logfile: /var/log/corosync/corosync.log
        debug: off
        timestamp: on
        logger_subsys {
                subsys: QUORUM
                debug: off
        }
}
```


- **fileline** : Lorsque cette option est définie sur « **off** », elle indique que les informations sur le fichier et la ligne du code source ne seront pas incluses dans les logs. Cela aide à réduire la quantité d'informations enregistrées, rendant les logs plus propres mais moins détaillés pour le débogage.
- **to_logfile** : Réglé sur « **yes** », cela indique que Corosync doit écrire les logs dans un fichier spécifique. C'est une pratique courante pour permettre une analyse postérieure plus facile des logs.
- **to_syslog** : Réglé sur « **no** », cela signifie que Corosync n'enregistrera pas les événements dans le journal système (syslog), ce qui est utile pour éviter le bruit dans le syslog général du système.
- **dogfile** : Spécifie le chemin d'accès au fichier de log où Corosync écrira ses messages. Ici, « **/var/log/corosync/corosync.log** » est utilisé, ce qui est un emplacement standard pour les logs des applications sous Linux.
- **debug** : Lorsqu'il est défini sur « **off** », cela indique que le mode débogage est désactivé, ce qui signifie que les informations de débogage ne seront pas enregistrées dans les logs. Cela aide à maintenir la propreté des logs et à réduire l'espace disque utilisé.

- Timestamp : Réglé sur « **on** », indique que chaque entrée de log doit inclure un horodatage. Cela est crucial pour suivre quand les événements se produisent et pour le dépannage.
- **logger_subsys** :
    - **subsys** : Spécifie le sous-système de Corosync pour lequel ces paramètres de logging s'appliquent. « **QUORUM** » est un composant essentiel de Corosync qui gère la cohérence et le quorum du cluster.
    - **debug** : Tout comme l'option globale de debug, ici elle est définie sur `off` pour le sous-système « **QUORUM** », ce qui signifie que les informations de débogage détaillées pour le sous-système quorum ne seront pas enregistrées.

Cette configuration assure que les logs sont gérés de manière à fournir suffisamment d'informations pour le suivi opérationnel et le dépannage de base, sans surcharger le système de fichiers avec des données de débogage détaillées ou redondantes.

La section « **QUORUM** » est fondamentale pour garantir que le cluster fonctionne de manière stable et cohérente, en évitant les problèmes qui peuvent survenir lors de défaillances partielles ou de communications interrompues entre les nœuds du cluster. Dans la section « **QUORUM** » :

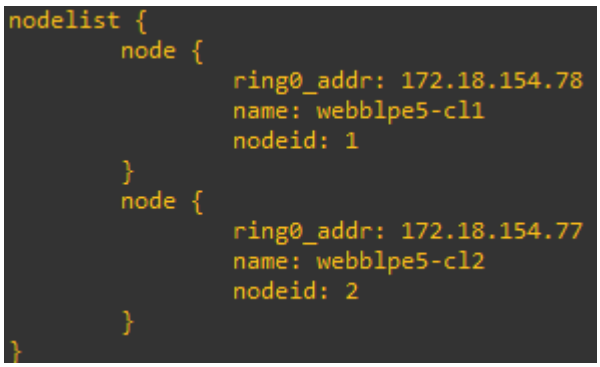
```
quorum {
        provider: corosync_votequorum
        expected_votes: 2
        two_nodes: 1
}
```


```
quorum {
        provider: corosync_votequorum
        expected_votes: 2
        two_nodes: 1
}
```


- **provider** : Définit le mécanisme utilisé pour le calcul du quorum. **corosync_votequorum** est un module de Corosync qui gère le quorum basé sur le nombre de votes que chaque nœud possède dans le cluster.
- **expected_votes** : Nombre total de votes attendus pour que le cluster soit en quorum. Ici, il est défini à 2, ce qui signifie que deux votes (normalement provenant de deux nœuds) sont nécessaires pour atteindre le quorum. Cela est typique dans une configuration où chaque nœud est configuré pour avoir un vote.
- **two_nodes** : Une option spécifique pour les clusters à deux nœuds. Quand cette option est réglée sur 1, Corosync ajuste le comportement du quorum pour permettre à un cluster de deux nœuds de continuer à fonctionner même si un seul nœud est en ligne. Sans cette option, un cluster à deux nœuds perdrait le quorum si un nœud tombe en panne, car la majorité absolue ne peut plus être atteinte.

La section « **nodelist** » est utilisée pour spécifier explicitement les membres du cluster :

```
nodelist {
        node {
                ring0_addr: 172.18.154.78
                name: webblpe5-cl1
                nodeid: 1
        }
        node {
                ring0_addr: 172.18.154.77
                name: webblpe5-cl2
                nodeid: 2
        }
}
```



- **node** : Cela définit un nœud spécifique dans le cluster.
- **ring0_addr** : L'adresse IP utilisée pour la communication de cluster sur l'anneau 0. Corosync peut gérer plusieurs anneaux pour la redondance ; ici, seul l'anneau 0 est configuré.
- **name** : Nom symbolique du nœud, utilisé pour l'identification facile du nœud dans la gestion et le suivi.
- **nodeid** : Un identifiant numérique unique pour chaque nœud dans le cluster. Les IDs de nœud sont importants pour identifier de manière unique chaque nœud au sein du cluster.

Pacemaker dépend de Corosync pour fonctionner, mais la réciproque n'est pas vraie. Si Corosync est arrêté, cela entraînera également l'arrêt de Pacemaker, et démarrer Pacemaker déclenchera automatiquement le démarrage de Corosync. On va donc venir redémarrer Corosync avec la commande suivante :

```
root@webblpe5-cl1:/etc/corosync# systemctl restart corosync.service
```

```
root@webblpe5-cl2:/etc/corosync# systemctl restart corosync.service
```

Après avoir redémarré Corosync, vous pouvez vérifier sa configuration en utilisant la commande suivante :

```
root@webblpe5-cl1:~# corosync-cfgtool -s
Local node ID 1, transport knet
LINK ID 0 udp
        addr    = 172.18.154.78
        status:
                nodeid:          1:     localhost
                nodeid:          2:     connected
```

```
root@webblpe5-cl2:~# corosync-cfgtool -s
Local node ID 2, transport knet
LINK ID 0 udp
        addr    = 172.18.154.77
        status:
                nodeid:          1:     connected
                nodeid:          2:     localhost
```

La commande **crm**, accessible via l'outil **crmsh**, est essentielle pour la gestion des clusters sous Pacemaker. Elle permet de manipuler la configuration du cluster et de gérer ses ressources de manière interactive ou par le biais de scripts. L'une des grandes forces de **crmsh** est sa capacité à propager automatiquement les changements de configuration à tous les nœuds du cluster, ce qui signifie que toute modification effectuée sur un nœud est répercutée sur tous les autres. Cela simplifie considérablement la gestion du cluster, car il n'est pas nécessaire de répéter les commandes sur chaque nœud individuellement. Ci-dessous, le retour de **crm status** sur mes deux serveurs :

```
root@webblpe5-cl1:~# crm status
Status of pacemakerd: 'Pacemaker is running' (last updated 2024-05-01 09:32:09 +04:00)
Cluster Summary:
  * Stack: corosync
  * Current DC: webblpe5-cl2 (version 2.1.5-a3f44794f94) - partition with quorum
  * Last updated: Wed May  1 09:32:09 2024
  * Last change:  Wed May  1 09:09:36 2024 by root via cibadmin on webblpe5-cl1
  * 2 nodes configured
  * 0 resource instances configured

Node List:
  * Online: [ webblpe5-cl1 webblpe5-cl2 ]

Full List of Resources:
  * No resources
```

```
root@webblpe5-cl2:~# crm status
Status of pacemakerd: 'Pacemaker is running' (last updated 2024-05-01 09:31:56 +04:00)
Cluster Summary:
  * Stack: corosync
  * Current DC: webblpe5-cl2 (version 2.1.5-a3f44794f94) - partition with quorum
  * Last updated: Wed May  1 09:31:56 2024
  * Last change:  Wed May  1 09:09:36 2024 by root via cibadmin on webblpe5-cl1
  * 2 nodes configured
  * 0 resource instances configured

Node List:
  * Online: [ webblpe5-cl1 webblpe5-cl2 ]

Full List of Resources:
  * No resources
```

On vient utiliser la commande suivant pour vérifier la la validité du fichier de configuration XML généré :

```
root@webblpe5-cl1:~# crm_verify -L -V
(unpack_resources)      error: Resource start-up disabled since no STONITH resources have been defined
(unpack_resources)      error: Either configure some or disable STONITH with the stonith-enabled option
(unpack_resources)      error: NOTE: Clusters with shared data need STONITH to ensure data integrity
crm_verify: Errors found during check: config not valid
```

```
root@webblpe5-cl2:~# crm_verify -L -V
(unpack_resources)      error: Resource start-up disabled since no STONITH resources have been defined
(unpack_resources)      error: Either configure some or disable STONITH with the stonith-enabled option
(unpack_resources)      error: NOTE: Clusters with shared data need STONITH to ensure data integrity
crm_verify: Errors found during check: config not valid
```

STONITH, qui signifie "**S**hoot **T**he **O**ther **N**ode **I**n **T**he **H**ead", est un mécanisme utilisé pour arrêter définitivement les nœuds défaillants du cluster. Ce processus implique l'arrêt total du serveur affecté en coupant son alimentation électrique, souvent via un onduleur. Cette méthode est particulièrement importante dans les environnements où plusieurs serveurs partagent un même disque, car elle empêche un serveur considéré comme défaillant de continuer à écrire sur le disque partagé, ce qui pourrait compromettre ou corrompre les données.

Pour des mesures de simplification, je vais désactiver STONITH par la commande suivante :

```
root@webblpe5-cl1:~# crm configure property stonith-enabled=false
```

```
root@webblpe5-cl2:~# crm configure property stonith-enabled=false
```

La vérification ne renvoie plus d'erreur :

```
root@webblpe5-cl1:~# crm_verify -L -V
root@webblpe5-cl1:~#
```

```
root@webblpe5-cl2:~# crm_verify -L -V
root@webblpe5-cl2:~#
```

Configuration des ressources Pacemaker

Une ressource Pacemaker représente un service ou une application gérée par Pacemaker dans le cadre d'un cluster. Cette ressource peut être un service réseau, un système de fichiers, une application, ou tout autre composant dont le fonctionnement continu est crucial pour l'entreprise. La gestion de ces ressources par Pacemaker est fondamentale pour assurer la haute disponibilité des services.

En haute disponibilité, les ressources sont généralement configurées pour s'exécuter sur plusieurs nœuds d'un cluster, permettant ainsi au service de rester disponible même en cas de défaillance d'un ou plusieurs nœuds. Pacemaker surveille l'état de ces ressources, gérant leur arrêt et leur démarrage, et en cas de défaillance d'un nœud, il redémarre les ressources affectées sur un autre nœud pré-configuré. Ce mécanisme garantit que le service reste disponible sans interruption significative, ce qui est crucial pour les opérations d'entreprise critiques.

Les ressources dans Pacemaker peuvent être configurées de manière très détaillée, incluant des paramètres pour le démarrage, l'arrêt, et le monitoring. Les ressources sont définies par des primitives dans la configuration de Pacemaker, chaque primitive faisant appel à un script d'application spécifique qui contrôle la ressource. Ces scripts peuvent suivre des standards tels que OCF (Open Cluster Framework) ou LSB (Linux Standard Base), qui assurent l'interopérabilité et la conformité dans des environnements de cluster diversifiés.

Ci-dessous, les ressources que j'ai déployées présenter via la commande « **crm configure show** » (disponible sur les deux serveurs) :

```
root@webblpe5-cl1:~# crm configure show
node 1: webblpe5-cl1 \
        attributes standby=off
node 2: webblpe5-cl2 \
        attributes standby=off
primitive IPFailover IPaddr2 \
        params ip=172.18.154.76 cidr_netmask=21 nic=ens192 \
        op monitor interval=10s timeout=10s start-delay=30s \
        op_params stop-delay=40s \
        op_params flabel=VIP
primitive serviceMySQL lsb:mariadb \
        op monitor interval=30s
primitive serviceWeb lsb:apache2 \
        op monitor interval=60s \
        op start timeout=40s interval=0 \
        op stop timeout=40s interval=0
group servWebBlpE5 IPFailover serviceWeb \
        meta migration-threshold=5
clone mysqlClone serviceMySQL
location cli-prefer-servWebBlpeE5 servWebBlpE5 role=Started inf: webblpe5-cl1
property cib-bootstrap-options: \
        have-watchdog=false \
        dc-version=2.1.5-a3f44794f94 \
        cluster-infrastructure=corosync \
        cluster-name=debian \
        stonith-enabled=false
```

1. **Configuration des Nœuds**

Les commandes pour les nœuds « **webblpe5-cl1** » et « **webblpe5-cl2** » définissent le mode de disponibilité de chaque nœud. En désactivant le mode standby (« **standby=off** »), les nœuds sont toujours actifs et prêts à prendre en charge les ressources du cluster sans être mis en attente.

2. **Ressources Primitives**

Ces ressources sont les services ou applications individuelles que le cluster doit gérer pour assurer leur disponibilité continue.

- **IPFailover** : Une ressource IPaddr2 configurée pour assurer un basculement d'IP (failover) en cas de défaillance du nœud principal. Les paramètres incluent l'adresse IP, le masque de sous-réseau, l'interface réseau, et des opérations de monitoring et de démarrage avec des intervalles et délais spécifiques.
- **serviceMySQL** : Une ressource pour le service de base de données MariaDB, avec une opération de monitoring configurée pour s'exécuter toutes les 30 secondes, permettant de vérifier régulièrement l'état du service.
- **serviceWeb** : Une ressource pour le service web Apache avec des opérations de monitoring plus détaillées, incluant des délais pour les opérations de démarrage et d'arrêt, assurant que le service est démarré et arrêté proprement.

3. **Groupement de Ressources**

Le groupement de l'**IPFailover** et du **serviceWeb** en un seul groupe « **servWebBlpE5** » permet de gérer ces ressources comme une unité unique. Cela signifie que lorsqu'une action est nécessaire, comme un démarrage ou un arrêt, elle est exécutée sur l'ensemble du groupe, garantissant que l'IP failover et le service web sont toujours localisés sur le même nœud.

4. **Clonage de Ressources**

Le clonage de la ressource MySQL avec « **mysqlClone** » permet de faire fonctionner cette ressource sur plusieurs nœuds, augmentant la disponibilité et la redondance. Cela signifie que **MariaDB** peut être actif sur plus d'un nœud à la fois, ce qui est utile pour les configurations de lecture-écriture réparties ou simplement pour garantir une disponibilité continue en cas de défaillance d'un nœud.

5. **Contraintes de Localisation**

La contrainte de localisation **« cli-prefer-servWebBlpeE5** » indique une préférence pour exécuter le groupe de ressources « **servWebBlpE5** » sur le nœud « **webblpe5-cl1** ». Cela influence le planificateur de Pacemaker pour favoriser ce nœud lors du démarrage des ressources, à moins que des conditions du cluster ne dictent autrement.

6. **Propriétés du Cluster**

Les propriétés définissent les paramètres globaux du cluster, tels que la désactivation de STONITH (« **stonith-enabled=false** »).

En faisant de nouveau un « **crm status** », on obtient le résultat suivant :

```
root@webblpe5-cl1:~# crm status
Status of pacemakerd: 'Pacemaker is running' (last updated 2024-05-01 09:43:45 +04:00)
Cluster Summary:
  * Stack: corosync
  * Current DC: webblpe5-cl2 (version 2.1.5-a3f44794f94) - partition with quorum
  * Last updated: Wed May  1 09:43:46 2024
  * Last change:  Wed May  1 09:40:43 2024 by hacluster via crmd on webblpe5-cl2
  * 2 nodes configured
  * 4 resource instances configured

Node List:
  * Online: [ webblpe5-cl1 webblpe5-cl2 ]

Full List of Resources:
  * Resource Group: servWebBlpE5:
    * IPFailover        (ocf:heartbeat:IPaddr2):         Started webblpe5-cl1
    * serviceWeb        (lsb:apache2):   Started webblpe5-cl1
  * Clone Set: mysqlClone [serviceMySQL]:
    * Started: [ webblpe5-cl1 webblpe5-cl2 ]
```

```
root@webblpe5-cl2:~# crm status
Status of pacemakerd: 'Pacemaker is running' (last updated 2024-05-01 09:44:17 +04:00)
Cluster Summary:
  * Stack: corosync
  * Current DC: webblpe5-cl2 (version 2.1.5-a3f44794f94) - partition with quorum
  * Last updated: Wed May  1 09:44:17 2024
  * Last change:  Wed May  1 09:40:43 2024 by hacluster via crmd on webblpe5-cl2
  * 2 nodes configured
  * 4 resource instances configured

Node List:
  * Online: [ webblpe5-cl1 webblpe5-cl2 ]

Full List of Resources:
  * Resource Group: servWebBlpE5:
    * IPFailover        (ocf:heartbeat:IPaddr2):         Started webblpe5-cl1
    * serviceWeb        (lsb:apache2):   Started webblpe5-cl1
  * Clone Set: mysqlClone [serviceMySQL]:
    * Started: [ webblpe5-cl1 webblpe5-cl2 ]
```

Attention, j'ai au préalable modifier quelques éléments pour que ma ressources « **mysqlClone** » et « **serviceMySQL** » fonctionne. Toujours dans l'optique d'une infrastructure HA multi-maitre, j'ai configuré le fichier « **/etc/mysql/mariadb.conf.d/50-server.cnf** » respectivement comme ci-dessous :

```
server-id               = 100
log_bin                 = /var/log/mysql/mysql-bin.log
expire_logs_days        = 10
max_binlog_size         = 100M
master-retry-count = 20
binlog_do_db = gsb_frais
replicate-do-db = gsb_frais
log_slave_updates = 1
```

```
server-id               = 101
log_bin                 = /var/log/mysql/mysql-bin.log
expire_logs_days        = 10
max_binlog_size         = 100M
master-retry-count = 20
binlog_do_db = gsb_frais
replicate-do-db = gsb_frais
log_slave_updates = 1
```

Ces paramètres sont utilisés pour configurer divers aspects du serveur de bases de données, en particulier ceux relatifs à la réplication et à la gestion des logs binaires. Voici une explication détaillée de chaque paramètre :

- **server-id = 100/101** : Ce paramètre assigne un identifiant unique au serveur au sein d'un environnement de réplication. Chaque serveur participant à la réplication doit avoir un ID unique. Cela permet au système de réplication de distinguer les serveurs.
- **log_bin = /var/log/mysql/mysql-bin.log** : Active le logging binaire, qui est essentiel pour la réplication. Les logs binaires enregistrent toutes les modifications apportées à la base de données de manière à ce que ces modifications puissent être répliquées sur les serveurs esclaves.
- **expire_logs_days = 10** : Définit le nombre de jours après lesquels les fichiers de log binaires seront automatiquement supprimés. Cela aide à gérer l'espace disque en éliminant les vieux logs qui ne sont plus nécessaires.
- **max_binlog_size = 100M** : Définit la taille maximale d'un fichier log binaire. Une fois cette taille atteinte, un nouveau fichier log est créé.
- **master-retry-count = 20** : Définit le nombre de tentatives qu'un serveur esclave fera pour se reconnecter au serveur maître après qu'une connexion ait échoué.
- **binlog_do_db = gsb_frais** : Spécifie quelle base de données doit être loggée dans les logs binaires. Seules les modifications apportées à la base de données spécifiée seront enregistrées.
- **replicate-do-db = gsb_frais** : Indique au serveur esclave de répliquer uniquement les modifications de la base de données spécifiée. Cela peut être utilisé pour limiter la réplication à certaines bases de données
- **log_slave_updates = 1** : Quand il est activé (1), cet option fait en sorte que les serveurs esclaves loggent les modifications qu'ils reçoivent du maître dans leur propre log binaire. Cela est nécessaire dans les configurations de chaînage de réplication ou pour les sauvegardes.

Il y a également quelque petit réglage a faire dans la console MariaDB pour la configuration de la réplication en mode multi-maître :

1. Arrêt des esclaves

Pour éviter des changements pendant la configuration, j'ai commencé par arrêter les processus de réplication sur les deux serveurs.

```
root@webblpe5-cl1:~# mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 32
Server version: 10.11.6-MariaDB-0+deb12u1-log Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> stop slave;
Query OK, 0 rows affected, 1 warning (0,000 sec)
```

```
root@webblpe5-cl2:~# mariadb
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 33
Server version: 10.11.6-MariaDB-0+deb12u1-log Debian 12

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> stop slave;
Query OK, 0 rows affected, 1 warning (0,000 sec)
```

2. Verrouillage des tables

J'ai ensuite verrouillé les tables pour assurer la cohérence des données durant le processus de sauvegarde.

```
MariaDB [(none)]> flush tables with read lock;
Query OK, 0 rows affected (0,004 sec)
```

```
MariaDB [(none)]> flush tables with read lock;
Query OK, 0 rows affected (0,005 sec)
```

3. Sauvegarde de la base de données

Avec les tables verrouillées, j'ai réalisé une sauvegarde des données à l'aide de

```
root@webblpe5-cl2:~# mysqldump -h 172.18.154.78 -u adm_gsb1 --databases gsb_frais -p > sauvBases.sql
Enter password:
root@webblpe5-cl2:~# ls
sauvBases.sql
```

4. Importation de la sauvegarde

J'ai importé la sauvegarde dans le second serveur pour synchroniser les bases de données.

```
root@webb1pe5-cl2:~# mysql -u root < sauvBases.sql
```

5. Vérification du statut du maître

Avant de procéder à la sauvegarde, j'ai vérifié le statut du maître sur chaque serveur pour obtenir le nom du fichier binaire et la position du log, informations nécessaires pour configurer correctement les esclaves par la suite.

```
MariaDB [(none)]> show master status;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000001 |      328 | gsb_frais    |                  |
+------------------+----------+--------------+------------------+
1 row in set (0,001 sec)
```

```
MariaDB [(none)]> show master status;
+------------------+----------+--------------+------------------+
| File             | Position | Binlog_Do_DB | Binlog_Ignore_DB |
+------------------+----------+--------------+------------------+
| mysql-bin.000001 |    10321 | gsb_frais    |                  |
+------------------+----------+--------------+------------------+
1 row in set (0,001 sec)
```

6. Modification des paramètres de maître

Avec les informations récupérées de l'étape du « **SHOW MASTER STATUS** », j'ai configuré chaque serveur pour reconnaître son nouveau maître.

```
MariaDB [(none)]> change master to
    -> master_host='172.18.154.77',
    -> master_user='adm_gsb2',
    -> master_password='          ',
    -> master_log_file='mysql-bin.000001',
    -> master_log_pos=10321;
Query OK, 0 rows affected, 1 warning (0,013 sec)
```

```
MariaDB [(none)]> change master to
    -> master_host='172.18.154.78',
    -> master_user='adm_gsb1',
    -> master_password='          ',
    -> master_log_file='mysql-bin.000001',
    -> master_log_pos=328;
Query OK, 0 rows affected, 1 warning (0,019 sec)
```

7. Redémarrage des esclaves

Les esclaves ont été redémarrés pour activer la réplication avec la nouvelle configuration.

```
MariaDB [(none)]> start slave;
Query OK, 0 rows affected (0,005 sec)
```

```
MariaDB [(none)]> start slave;
Query OK, 0 rows affected, 1 warning (0,001 sec)
```

8. Déverrouillage des tables

Les tables ont été déverrouillées une fois toutes les opérations de configuration terminées, permettant la reprise des opérations normales.

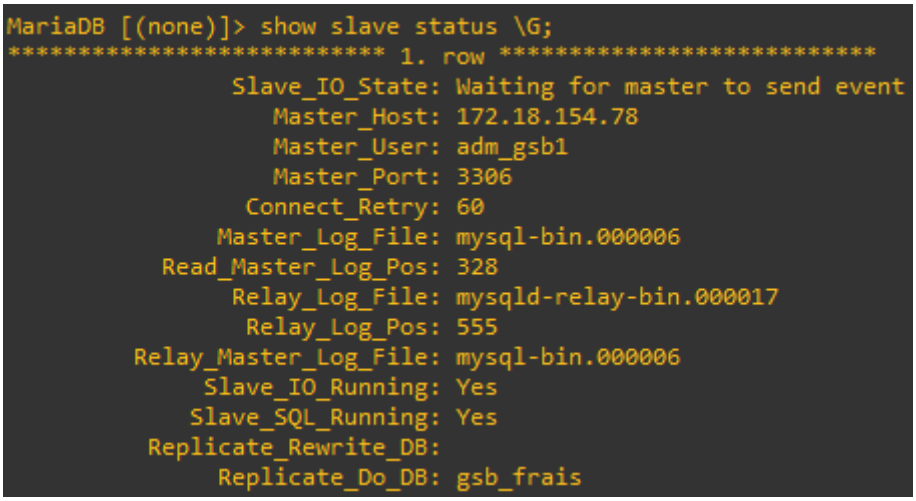
```
MariaDB [(none)]> unlock tables;
Query OK, 0 rows affected (0,000 sec)
```

```
MariaDB [(none)]> unlock tables;
Query OK, 0 rows affected (0,001 sec)
```

9. Vérification finale du statut des esclaves

J'ai vérifié le statut des esclaves pour confirmer que la réplication était opérationnelle après les modifications.

```
MariaDB [(none)]> show slave status \G;
*************************** 1. row ***************************
               Slave_IO_State: Waiting for master to send event
                  Master_Host: 172.18.154.77
                  Master_User: adm_gsb2
                  Master_Port: 3306
                Connect_Retry: 60
              Master_Log_File: mysql-bin.000005
          Read_Master_Log_Pos: 342
               Relay_Log_File: mysqld-relay-bin.000017
                Relay_Log_Pos: 641
        Relay_Master_Log_File: mysql-bin.000005
             Slave_IO_Running: Yes
            Slave_SQL_Running: Yes
          Replicate_Rewrite_DB:
                Replicate_Do_DB: gsb_frais
```



Conclusion

La mise en œuvre réussie de cette infrastructure à haute disponibilité nécessite une planification minutieuse et une exécution précise. Le cluster maître-maître, en utilisant Corosync et Pacemaker, offre une solution robuste pour les environnements nécessitant une haute disponibilité et une gestion efficace des failovers. Cette configuration aide non seulement à minimiser les interruptions de service mais assure également la sécurité et l'intégrité des données à travers la réplication continue.