TP Sécurité des réseaux

BRIZAI Olivier
THORAVAL Maxime

13 février 2011

Table des matières

1 Introduction

Le but de ce TP est de mettre en place un réseau sécurisé à l'aide d'un firewall CISCO ASA.
Ci-dessous, le réseau que nous souhaitons obtenir (les règles de filtrage ne sont pas représentées).

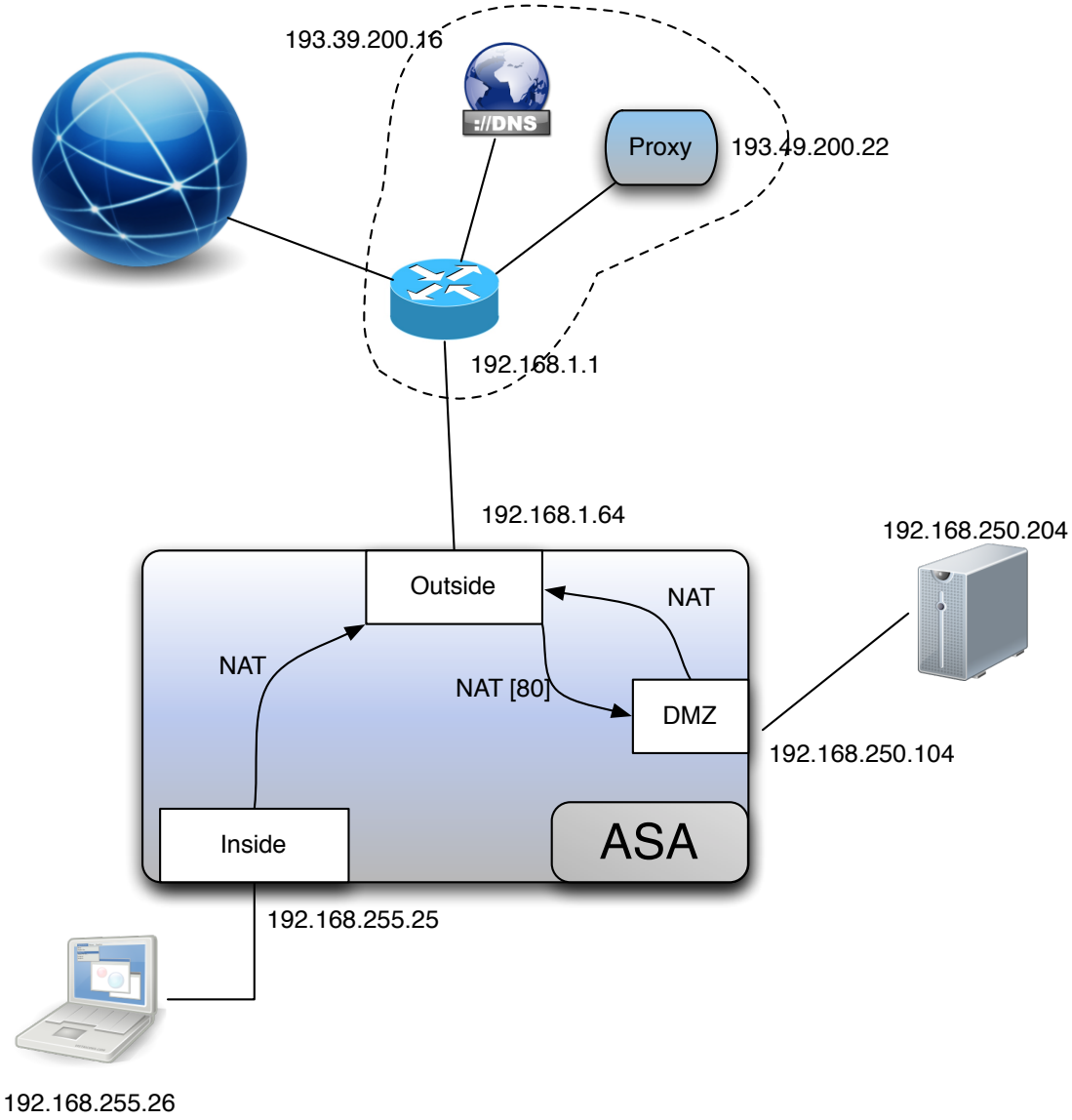


FIGURE 1 – Réseau à obtenir

2 Installation et Configurations préliminaires

2.1 Client

Dans un premier temps, nous avons installé Ubuntu 9.04 (version client) sur le PC relié à l'interface *inside*. Ceci effectué, nous réalisons les démarches suivantes, c'est à dire mise en place de Java ainsi que l'installation du paquet « Minicom ».
Nous lançons ensuite la commande **minicom -s** et définissons les divers paramètres afin de configurer le port console. Puis, nous définissons l'adresse *inside* de l'ASA. Nous pouvons maintenant, à partir de celle-ci, accéder à l'interface d'administration de l'ASA au sein de notre navigateur.
La figure ci-dessous présente l'accueil de celle-ci.

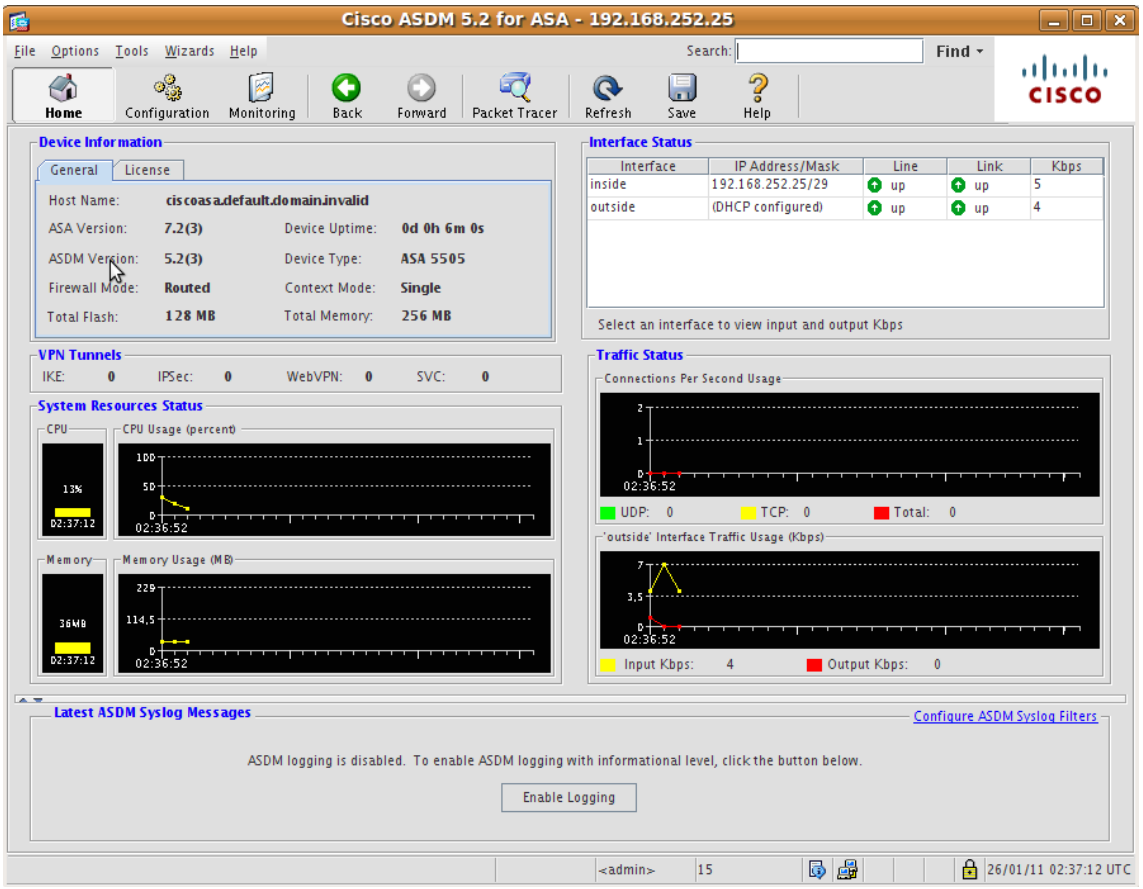


FIGURE 2 – Interface de configuration

Nous avons ensuite utilisé le « Wizard »de l'application pour mettre en place un certain nombres de paramètres tel que adresses IP (inside, outside, dmz) ou encore la répartition des interfaces du firewall (cf. figure ci-dessous).

Name	Switch Ports	Security Level	IP Address	Subnet Mask	VLAN
inside	Ethernet0/1, Ethernet0/2, Ethernet0/3, Ethernet0/4, Ethernet0/5, Ethernet0/6	100	192.168.252.25	255.255.255.248	vlan1
outside	Ethernet0/0	0	192.168.1.64	255.255.255.0	vlan2
dmz	Ethernet0/7	50	192.168.250.104	255.255.255.0	vlan3

FIGURE 3 – Configuration des interfaces

Enfin, nous mettons en place une route statique sur l'interface *outside* afin que

tous les paquets provenant des autres interfaces soit envoyés par défaut à la passerelle de l'ENSICAEN.

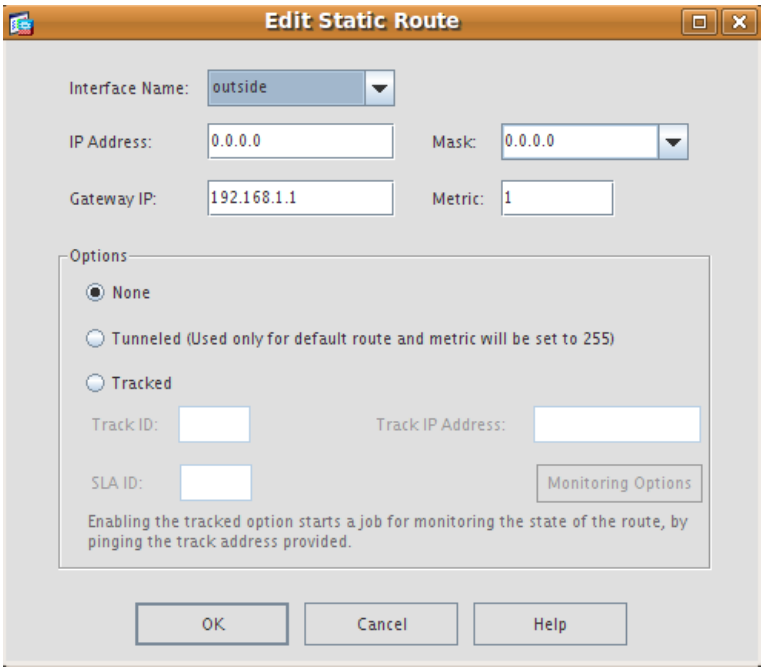


FIGURE 4 – Route statique

2.2 Serveur

En parallèle, nous installons « Ubuntu 9.0.4 server » sur le PC relié à l'interface *dmz* du firewall. Lors de l'installation, nous indiquons que nous souhaitons avoir par défaut les services suivants : un serveur SSH et un serveur web (LAMP).

L'installation terminée, nous allons maintenant configurer les informations réseau de notre serveur. Nous renseignons son adresse IP (192.168.250.204), le masque associé et enfin le routeur (ici il s'agit de l'adresse de l'interface *dmz* de notre firewall).
Afin de mettre en place ces informations, nous allons modifier le fichier */etc/network/interfaces* de la sorte :

```
1 auto eth0
2 iface eth0 inet static
3     address 192.168.250.204
4     netmask 255.255.255.0
5     gateway 192.168.250.104
```

3 Configuration Inside

Dans cette partie, nous avons configuré notre firewall afin de permettre certaines actions au sous réseau relié à l'interface *inside*.

3.1 Configuration du NAT

Dans un premier temps, il nous a fallu configurer une règle de NAT afin de traduire l'adresse privée de l'interface *inside* en l'adresse publique de l'interface *outside*. Nous devons effectuer cette étape afin de réduire les adresses IP utilisées, d'une part dans le but de ralentir la pénurie d'adresse IPv4, mais aussi pour que la passerelle de l'ENSICAEN n'est qu'une adresse à gérer (celle définie à l'interface *outside*).

Un NAT a pour effet de remplacer les adresses sources des paquets provenant d'un réseau (ou PC) par celle souhaitée (ici remplacement de celles du sous-réseau *inside* par *outside*). Pour les paquets retours (exemple paquet acquitant la réception), le firewall va pouvoir le transmettre au bon destinataire grâce à une sauvegarde de la transaction.

Ci-dessous, la configuration de notre NAT, pour le sous-réseau de lié à notre interface *inside* (192.168.252.24), nous lions l'adresse de l'interface *outside*.

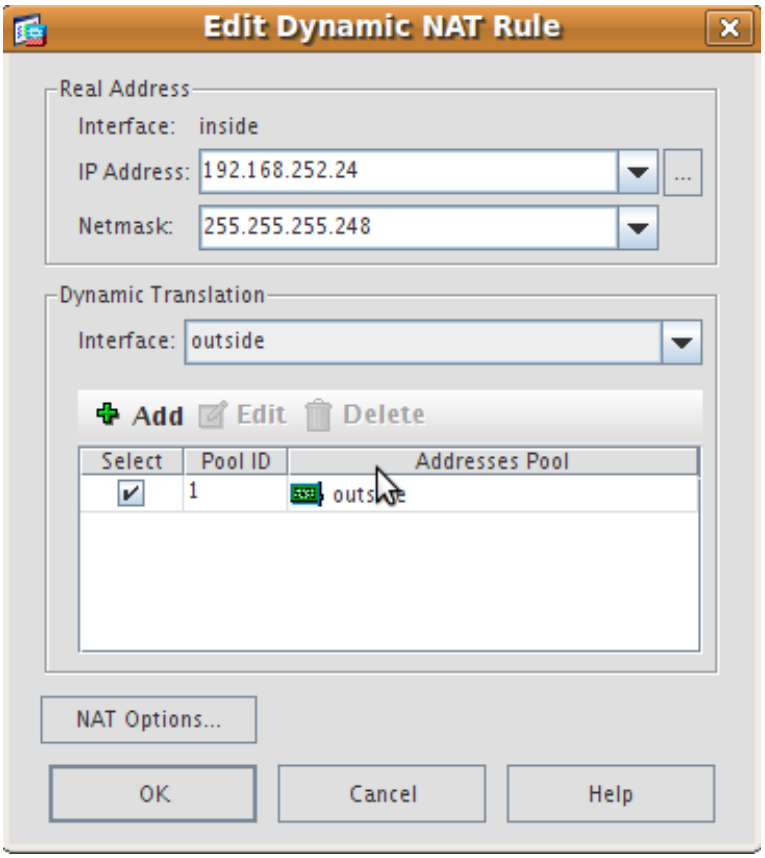


FIGURE 5 – Configuration du NAT

3.2 Règles de filtrage

Notre NAT crée, nous allons maintenant mettre en place des règles de filtrage afin de ne laisser passer que les paquets liés à des services définis. Il faut savoir que lorsque des paquets TCP et UDP sont envoyés, une connexion est établie. Cela permet de n'avoir à définir que les règles de sortie, celles d'entrées étant liées. Nous pourrons remarquer que le port source des règles est toujours définis sur « Any », en effet, l'application effectuant la demande n'utilise pas forcément le port dédié.

Chaque règle appliquée ici autorise les services à tout le sous réseau connecté à l'interface *inside*. Il aurait, par exemple, pu être possible de réduire l'accès au service SSH qu'à certaines machines mais nous pensons que ce n'est pas nécessaire dans le cadre de notre TP.

Dans un premier temps, nous autorisations les flux TCP et UDP sur le port 53 (DOMAIN) qui sont à destination de 193.49.200.16 (adresse du serveur DNS de l'ENSICAEN).

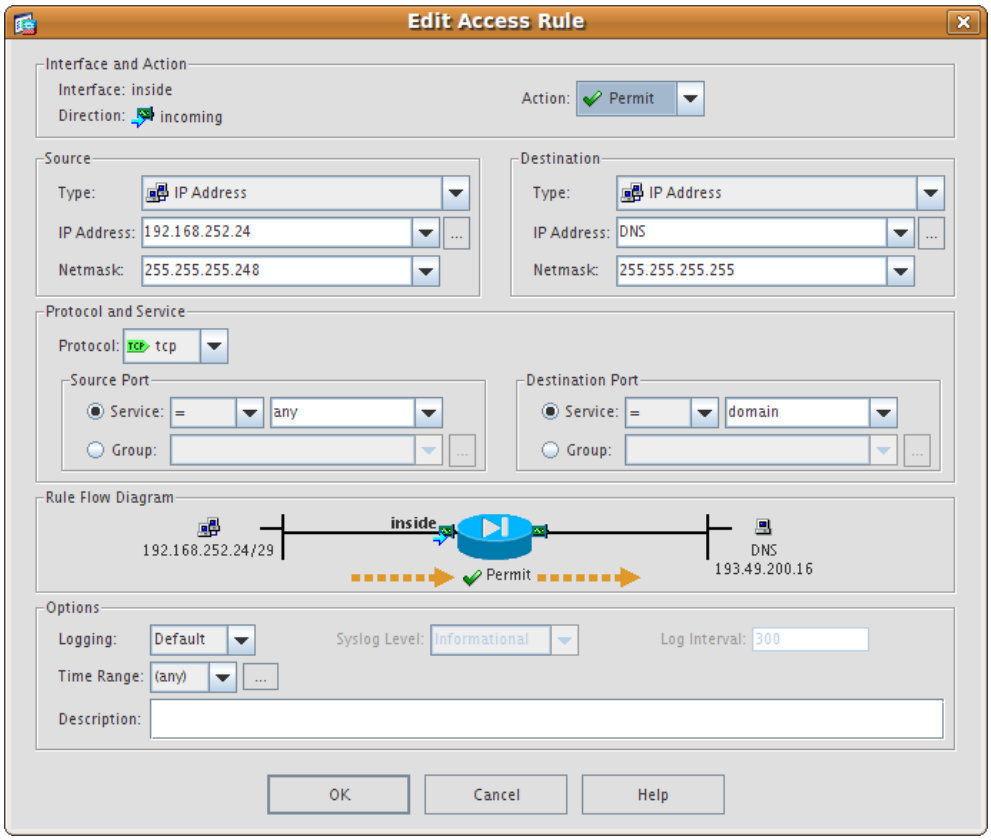


FIGURE 6 – Règle TCP d'accès au DNS

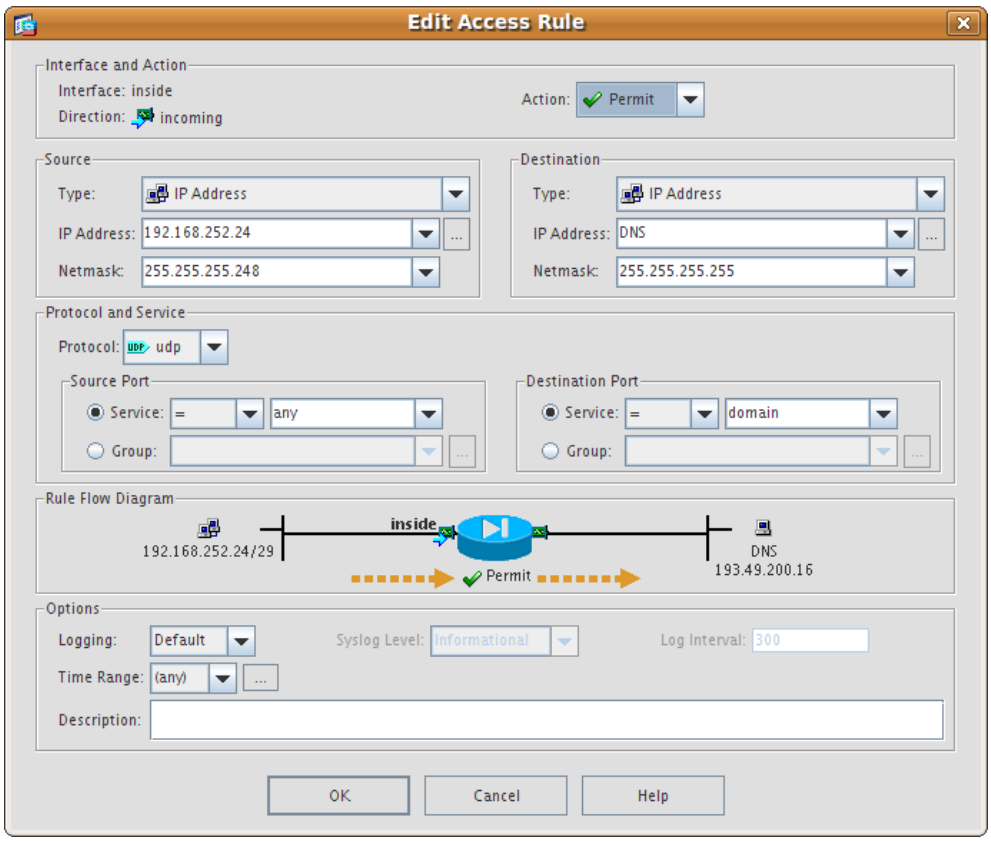


FIGURE 7 – Règle UDP d'accès au DNS

Maintenant, nous créons la règle autorisant le flux SSH (TCP sur le port 22). Nous ne nous soucions pas de la cible de la demande.

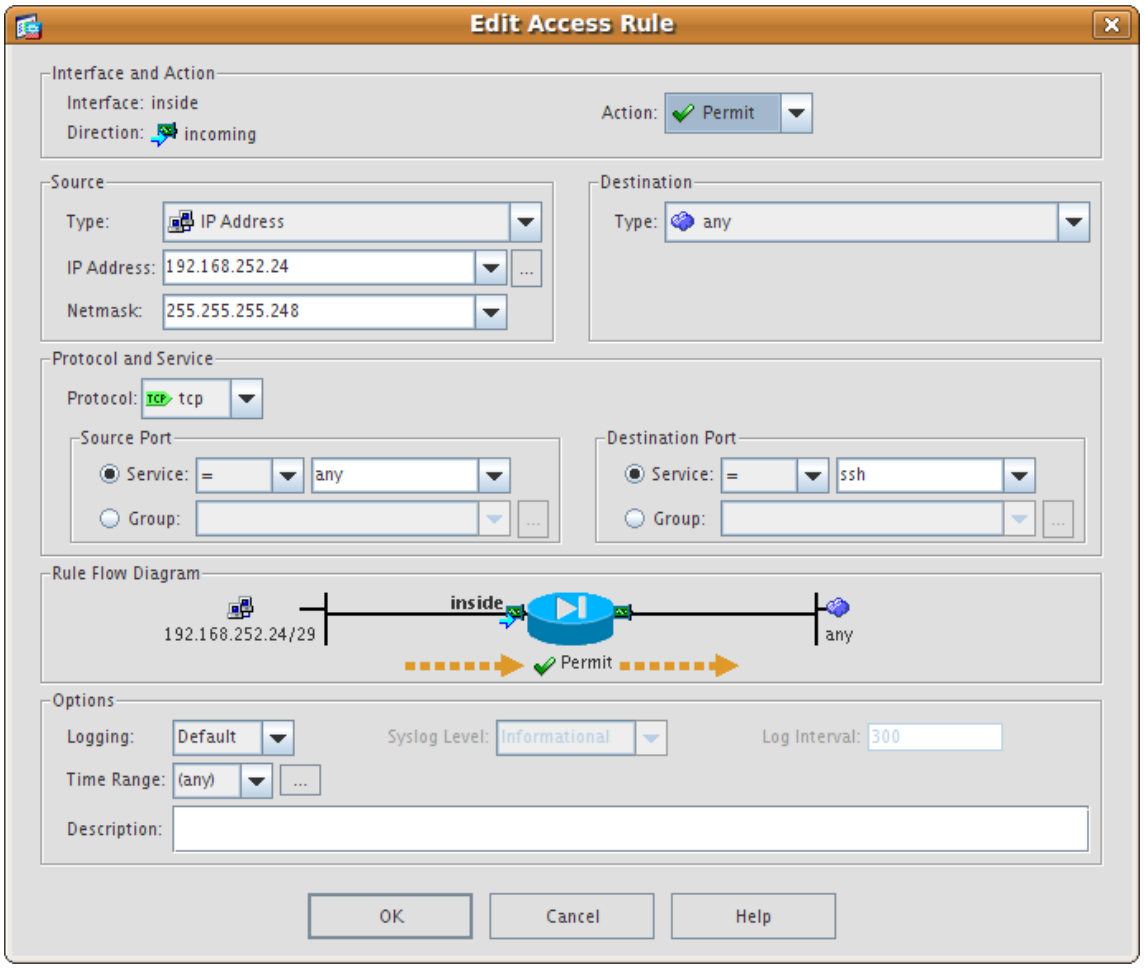


FIGURE 8 – Règle SSH

Puis la règle autorisant le flux HTTP (TCP sur le port 80) à destination de n'importe quelle machine.

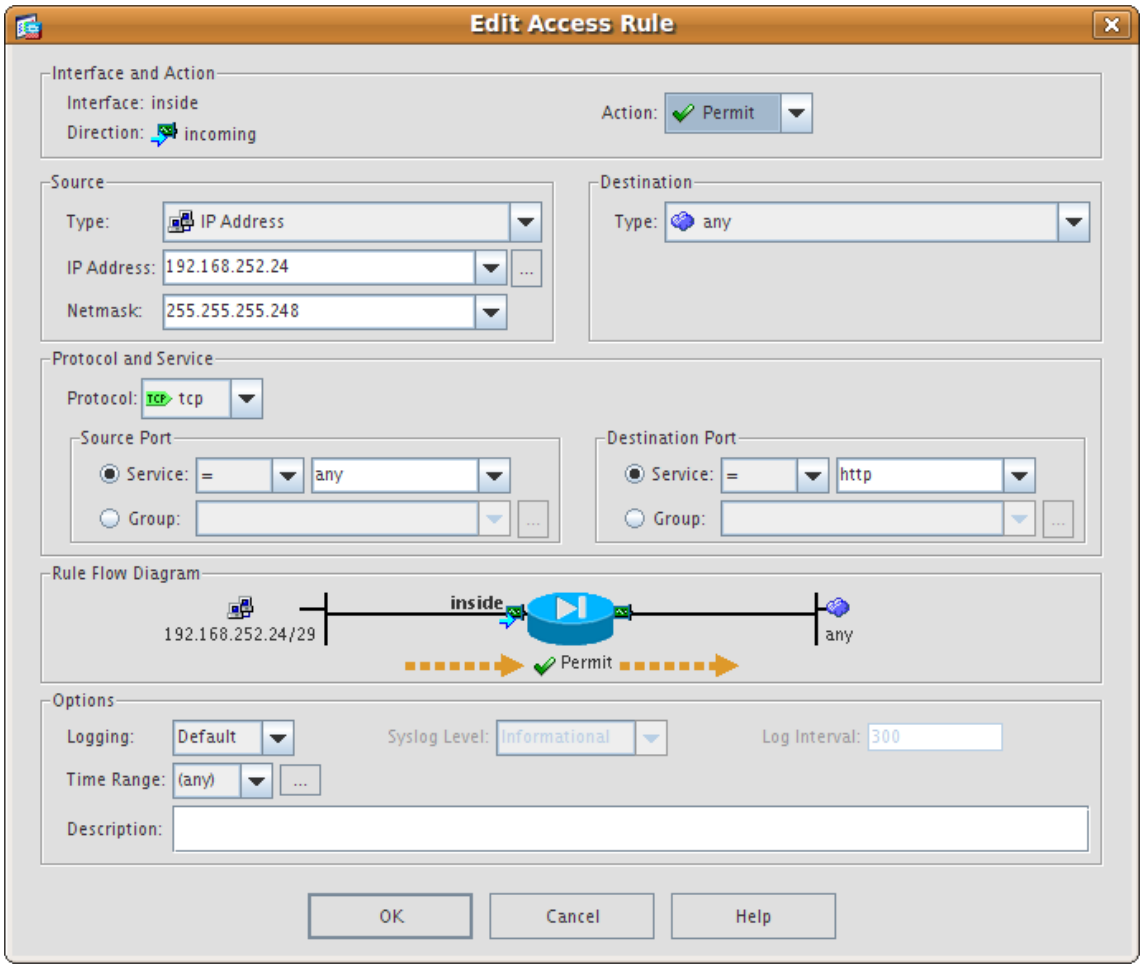


FIGURE 9 – Règle HTTP

Enfin, nous autorisons le flux à destination d'un proxy (TCP sur le port 3128 = port du proxy de l'école). Bien entendu, nous nous restreignons à l'adresse du proxy de l'ENSICAEN.

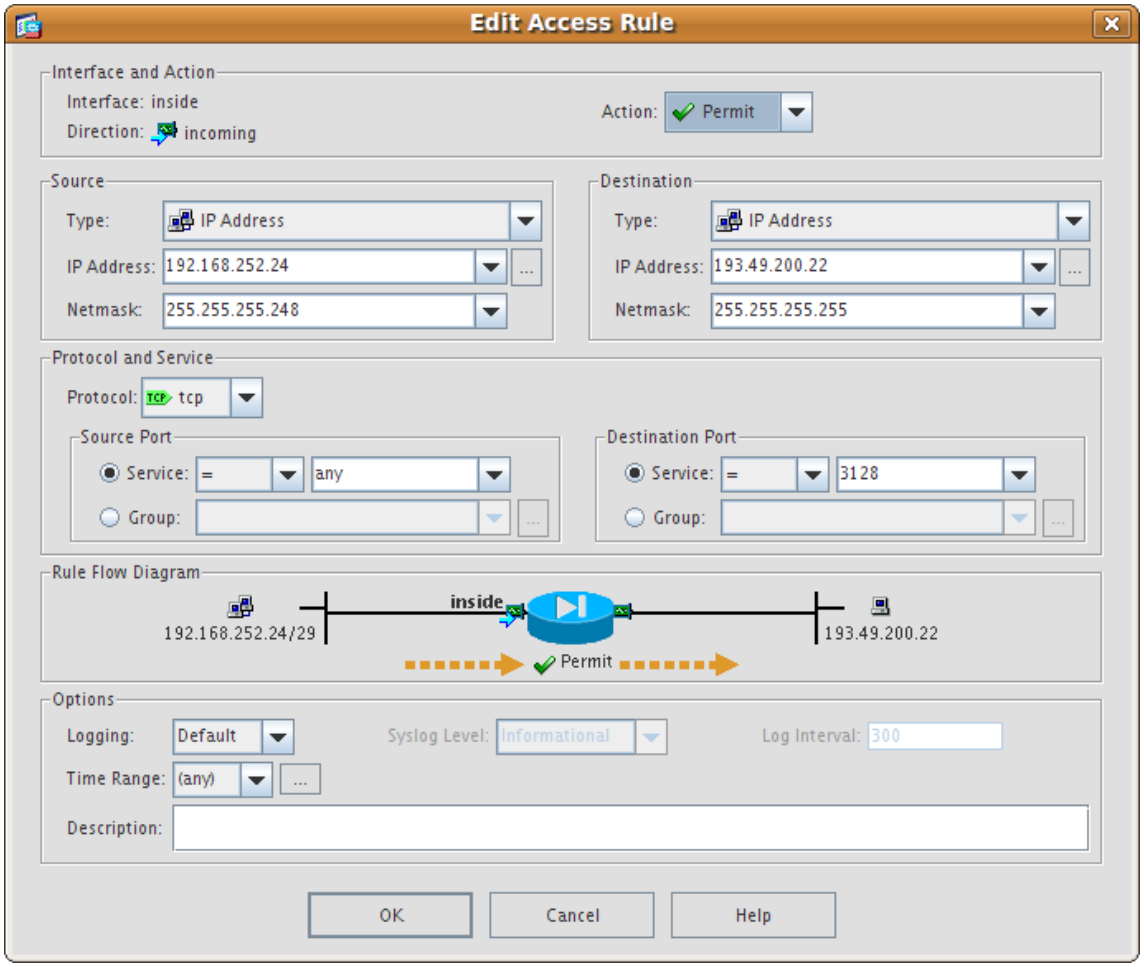


FIGURE 10 – Règle Proxy

Problèmes rencontrés

Ping

Nous sommes maintenant censé pouvoir accéder au routeur de l'école (adresse 192.168.1.1). Pour le vérifier, nous lançons la commande **ping** sur son adresse. On remarque que nous n'avons pas de retour de cette commande. Afin de vérifier l'erreur, nous allons regarder le *monitoring* de notre firewall. Ceci va nous permettre de suivre son activité. Après analyse des traces, nous avons pu comprendre l'échec de la commande **ping**. En effet, elles nous informent que les paquets de type ICMP ne sont pas autorisés à destination de l'interface *inside*. Afin de résoudre ce problème, nous devons rajouter une nouvelle règle de filtrage que nous avons défini de la manière ci-dessous.

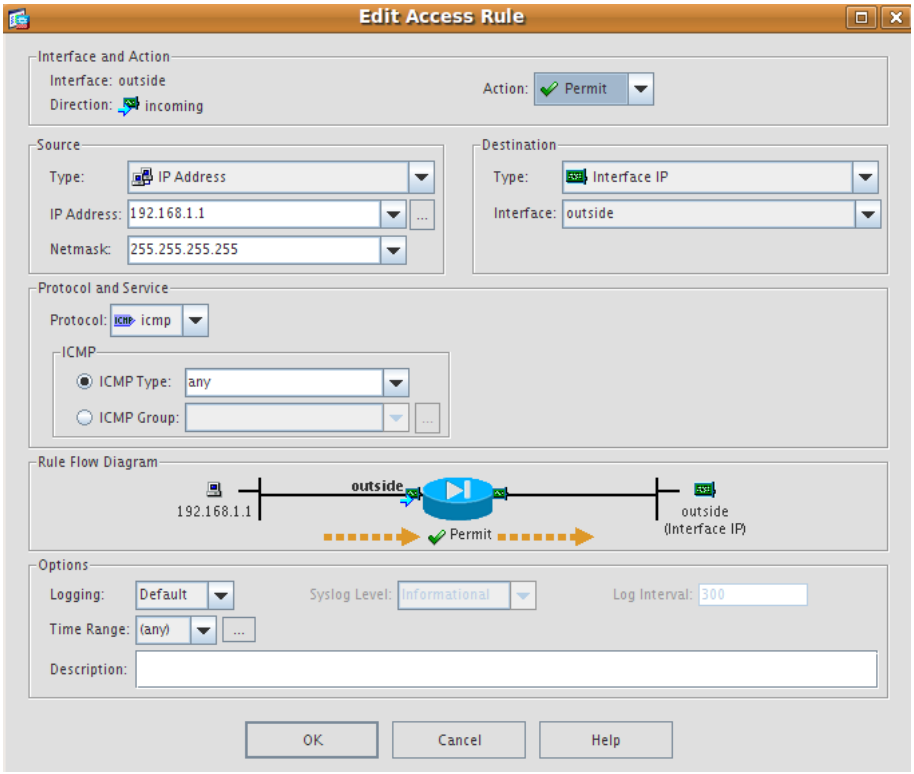


FIGURE 11 – Filtrage ICMP pour autoriser le retour de ping

Cette règle mise en place, nous lançons une nouvelle fois la commande **ping**. Comme visible sur la figure ci-dessous, il n'y a plus d'échec.

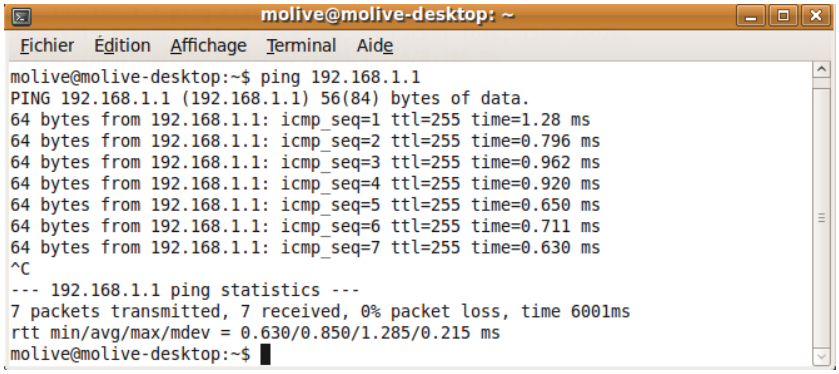


FIGURE 12 – Résultat ping

Utilisation du DNS

La règle du DNS étant active, nous considérions que l'ordinateur branché sur *inside* pouvait accéder à son service. Pour le vérifier, nous avons essayer plusieurs commandes listées ci-dessous

```
1 ping google.fr
2 host google.fr
3 nslookup google.fr
```

Chacune de ses commandes ne fonctionnaient pas, en effet, elles indiquaient qu'elle n'arrivait pas à récupérer l'adresse IP lié au nom de domaine. Etant donnée que c'est au DNS de nous fournir ces informations, nous avons conclu qu'il y avait un problème de configuration. Pour tester si notre règle de filtrage fonctionnait, nous avons effectué ceci

```
1         telnet 193.49.200.16 53
2         Trying 193.49.200.16...
3         Connected to ns.ecole.ensicaen.fr.
4         Escape character is '^]'.
```

Nous testons la connexion au serveur DNS sur le port 53 (comme définis dans nos règles). Nous remarquons que nous avons pu nous y connecter. (les messages suivants sont dû au fait que nous utilisions **telnet** pour nous connecter). Nos règles sont donc fonctionnelles. Afin de vérifier l'utilisation du DNS, nous avons fait ceci

```
1  nslookup
2  > server 193.49.200.16
3  Default server: 193.49.200.16
4  Address: 193.49.200.16#53
5  > google.fr
6  Server:         193.49.200.16
7  Address:        193.49.200.16#53
8
9  Non-authoritative answer:
10 Name:   google.fr
11 Address: 209.85.229.99
12 Name:   google.fr
13 Address: 209.85.229.147
14 Name:   google.fr
15 Address: 209.85.229.104
```

Dans un premier temps, nous indiquons à **nslookup** l'adresse IP du serveur DNS. Puis, re-testons avec le nom de domaine *google.fr*. Nous pouvons voir qu'il y a un retour, le DNS est donc utilisable. Après recherche, il se trouve que c'est dans le système Linux en lui-même que nous avions oublié d'indiquer l'adresse IP du serveur DNS au moment de rentrer celle de la machine.

4 Configuration DMZ

Maintenant que la configuration du PC client est mis en place et fonctionnelle, nous allons configurer notre serveur. Celui-ci doit être accessible de l'extérieur en HTTP et SSH, mais aussi par notre PC client.

4.1 Configuration du NAT

Comme pour l'interface *inside*, nous allons définir une NAT afin de traduire les adresses du sous réseau. Etant donné que nous n'avons qu'un serveur de connecté sur l'interface *dmz*, nous donnons son adresse et non celle du sous réseau.

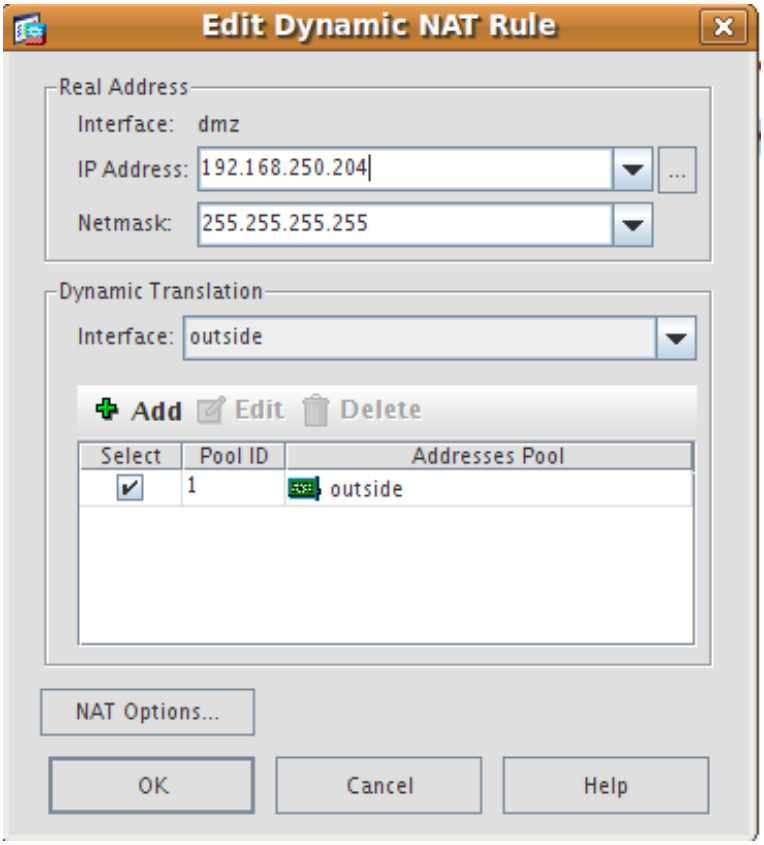


FIGURE 13 – NAT pour l'interface *dmz*

4.2 Règles de filtrage

Nous allons aussi autoriser quelques services à notre serveur. Pour toutes nos règles, nous allons limiter l'adresse IP source à celle de notre serveur, en effet, c'est la seule machine du sous-réseau.

Dans un premier temps, l'accès au serveur DNS. Nous autorisons le flux TCP et UDP sur le port 53 spécifiquement pour notre serveur (192.168.250.204).

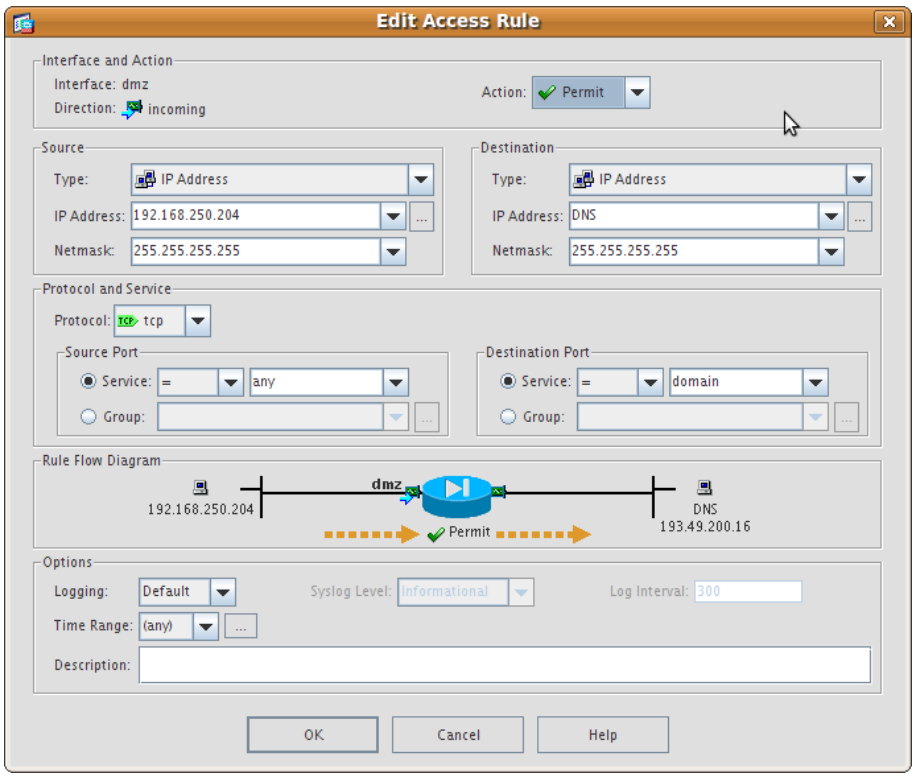


FIGURE 14 – Règle TCP pour l'accès au DNS

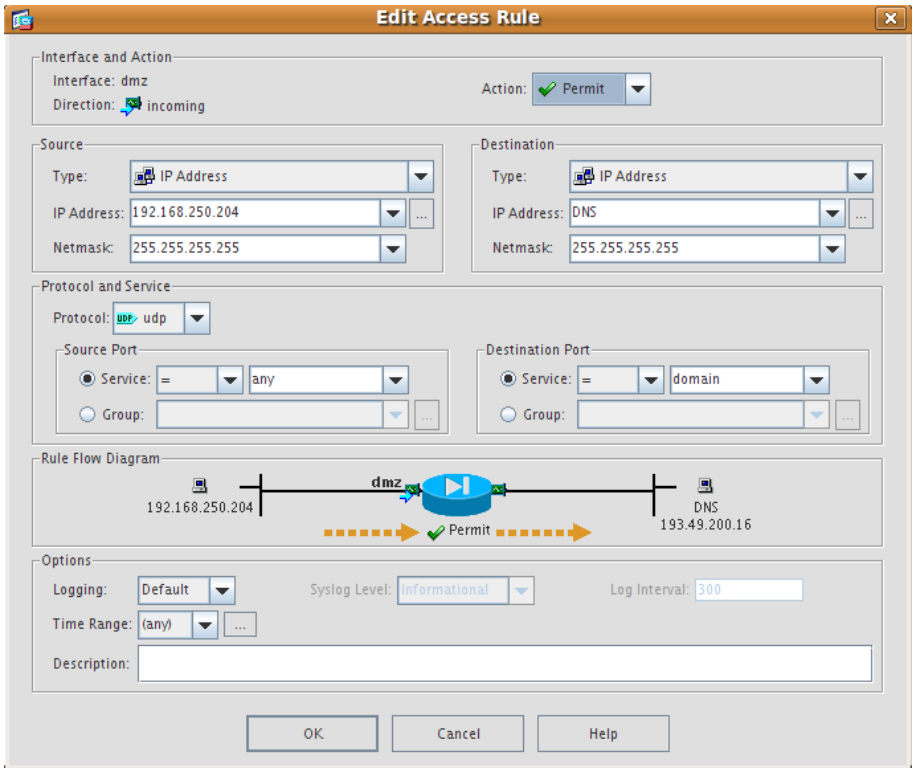


FIGURE 15 – Règle UDP pour l'accès au DNS

Nous devons aussi indiquer au système l'adresse du serveur DNS. Pour ce faire, nous éditons le fichier */etc/resolv.conf* du PC serveur de cette manière.

```
1 nameserver 193.49.200.16
```

Nous autorisons aussi le flux HTTP et SSH en sortie, tout comme pour l'interface *inside*.

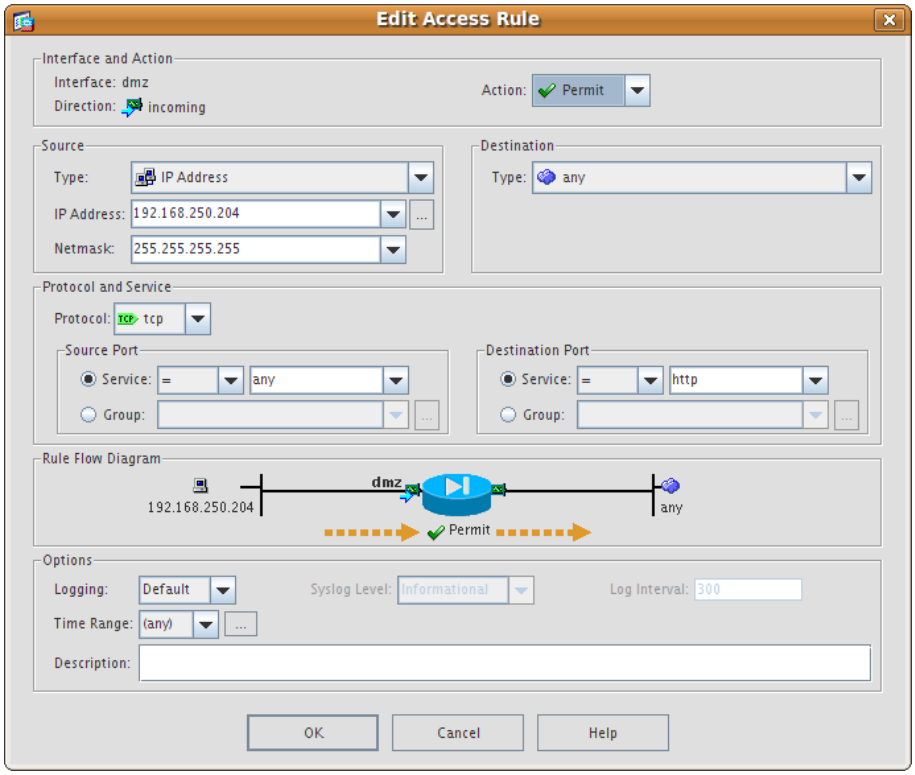


FIGURE 16 – Règle HTTP

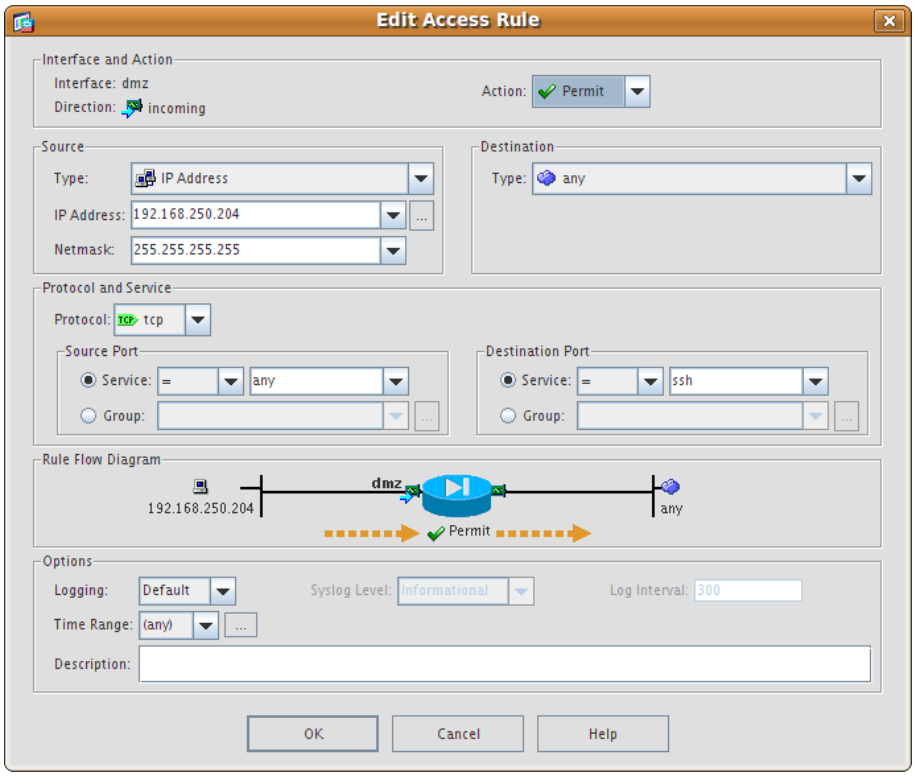


FIGURE 17 – Règle SSH

Nous donnons aussi l'autorisation d'envoyer des paquets en direction du proxy

de l'ENSICAEN.

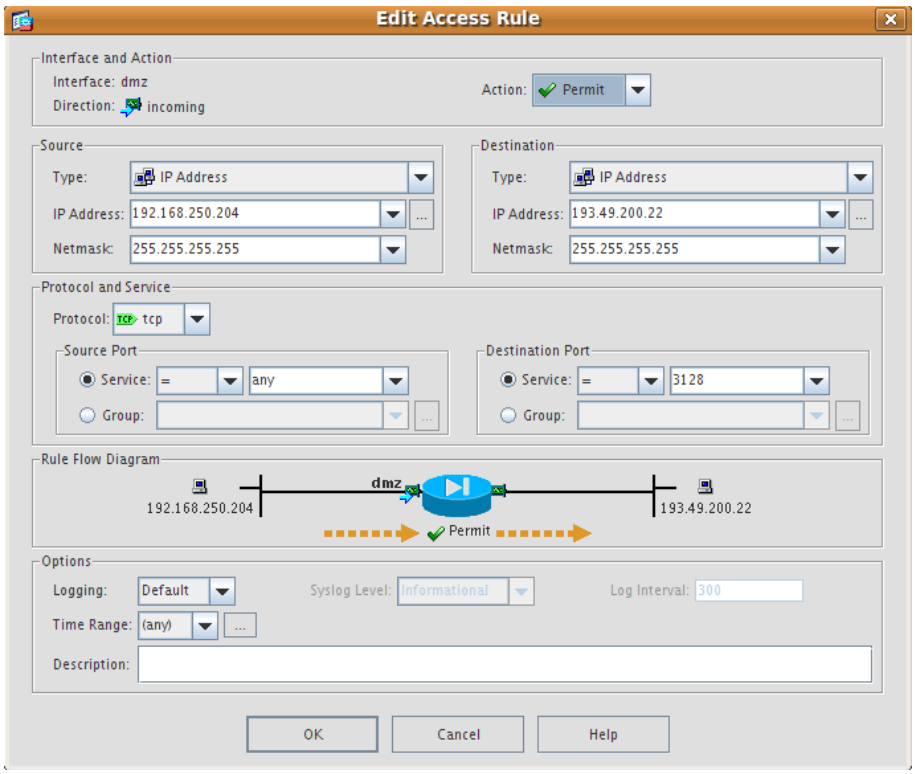


FIGURE 18 – Règle d'accès au proxy

Notre serveur est maintenant capable de discuter avec l'extérieur. Notre but est de pouvoir y accéder à partir de notre client. Pour cela, nous utiliserons la technologie SSH. Avant de tester celle-ci, nous allons voir si notre serveur est accessible. Pour cela, nous allons utiliser la commande **ping** une nouvelle fois à partir de notre PC client. Nous remarquons que cela ne fonctionne pas, nous ne pouvons y accéder. Les log nous indique qu'il y a un problème de translation. Ceci est dû au NAT crée entre l'interface *inside* et *outside*. Afin de résoudre ceci, nous allons y créer une exception indiquant qu'il ne faut pas traduire les messages en provenance du sous-réseau *inside* à destination du serveur (192.168.250.204).

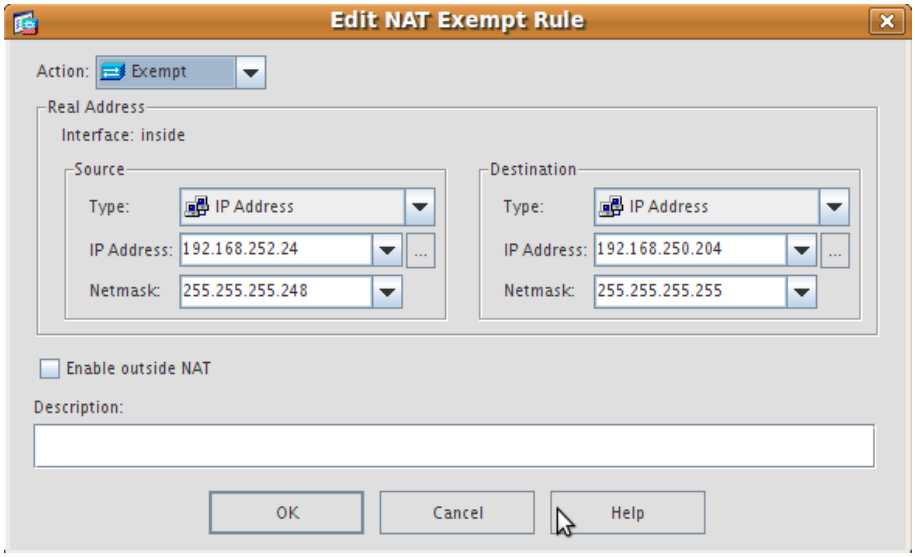


FIGURE 19 – Exception du NAT

4.3 HTTP et SSH à partir du réseau ENSICAEN

Nous pouvons maintenant accéder à notre serveur à partir de notre PC que ce soit en SSH ou en HTML. Cependant, cet accès est restreint à l'interface *inside*, en effet, nous souhaiterions que des personnes reliées au serveur de l'ENSICAEN puisse accéder à nos pages web ou encore se connecter en SSH.

Dans un premier temps, nous acceptons les flux HTTP. Nous aurions pu mettre l'adresse IP de notre serveur en tant que destination, mais sachant qu'il n'y a pas de serveur HTTP sur notre PC client, les utilisateurs n'ont aucun intérêt à aller l'interroger.

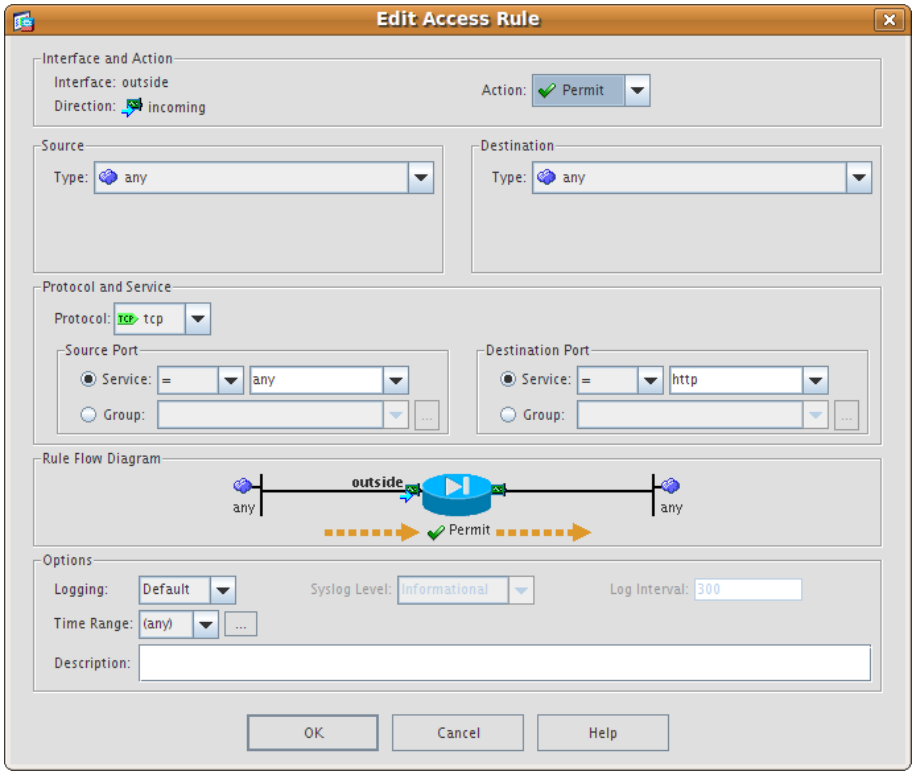


FIGURE 20 – Filtrage HTTP sur outside

Nous configurons maintenant afin qu'il puisse y avoir des demandes de connexion SSH à partir de l'extérieur. Pour plus de sécurité, nous n'avons indiqué que l'IP du serveur. Nous aurions aussi pu ne pas le renseigner, limitant l'accès SSH à notre client *inside*.

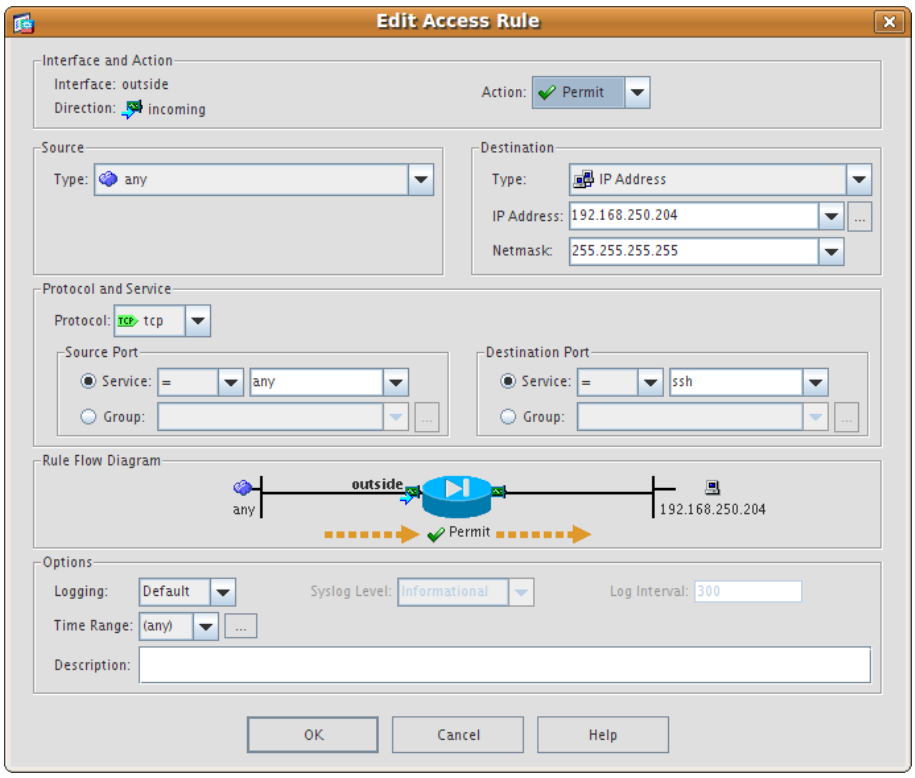


FIGURE 21 – Filtrage SSH

Il nous fallait indiquer au firewall que les flux HTTP et SSH rentrant dans l'interface *dmz* doivent obligatoirement être redirigé à *outside* en utilisant son adresse. Ci-dessous, les deux NAT statiques définis pour effectuer ceci.

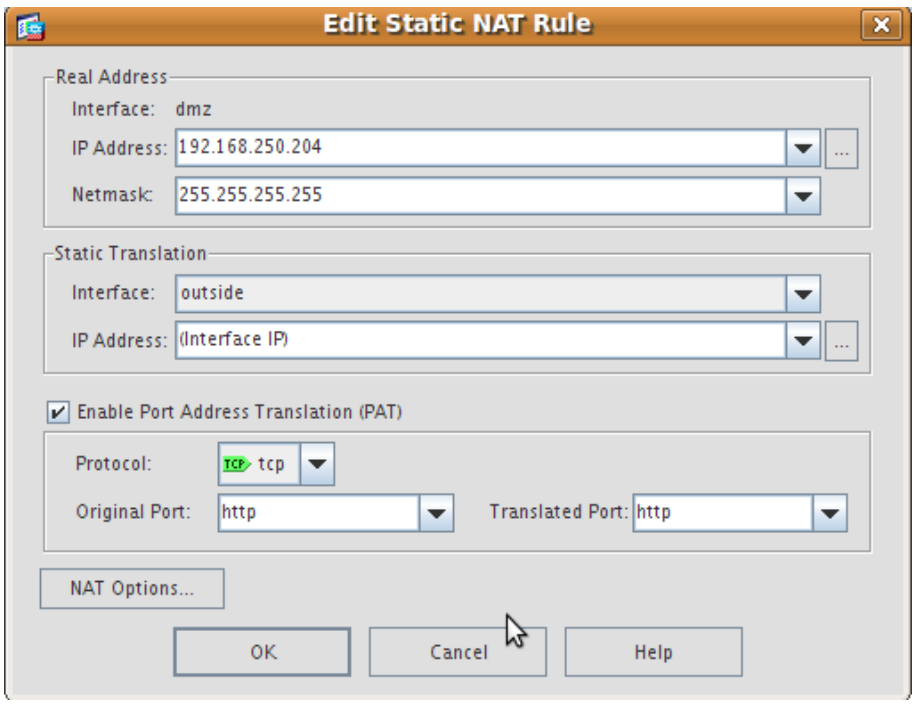


FIGURE 22 – Route statique pour HTTP

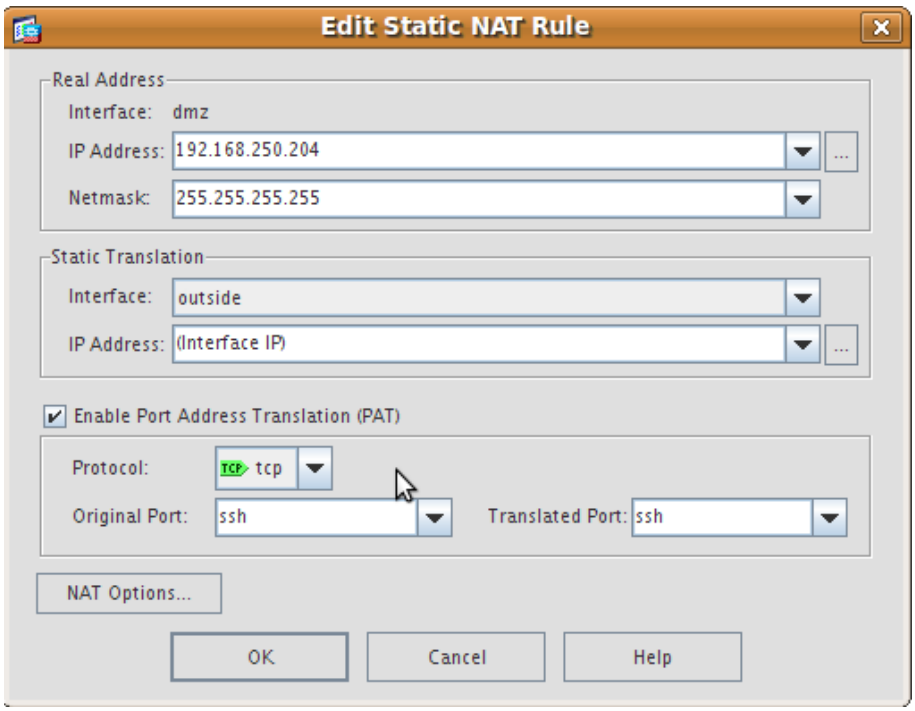


FIGURE 23 – Route statique pour SSH

A partie de ce moment, il était possible aux personnes présentes dans le réseau ENSICAEN d'accéder à notre serveur HTTP mais aussi à se connecter en SSH. Vous pourrez trouver en Annexe notre fichier de configuration.

5 VPN

Suite aux configurations que nous venons d'effectuer, nous avons permis l'accès à notre serveur HTTP et SSH présents dans notre *dmz*. Cependant, nous souhaitons maintenant permettre l'accès à un serveur de fichier présent sur notre ordinateur relié à la *inside*. Pour ce faire, nous allons mettre en place une passerelle VPN sur l'ASA.

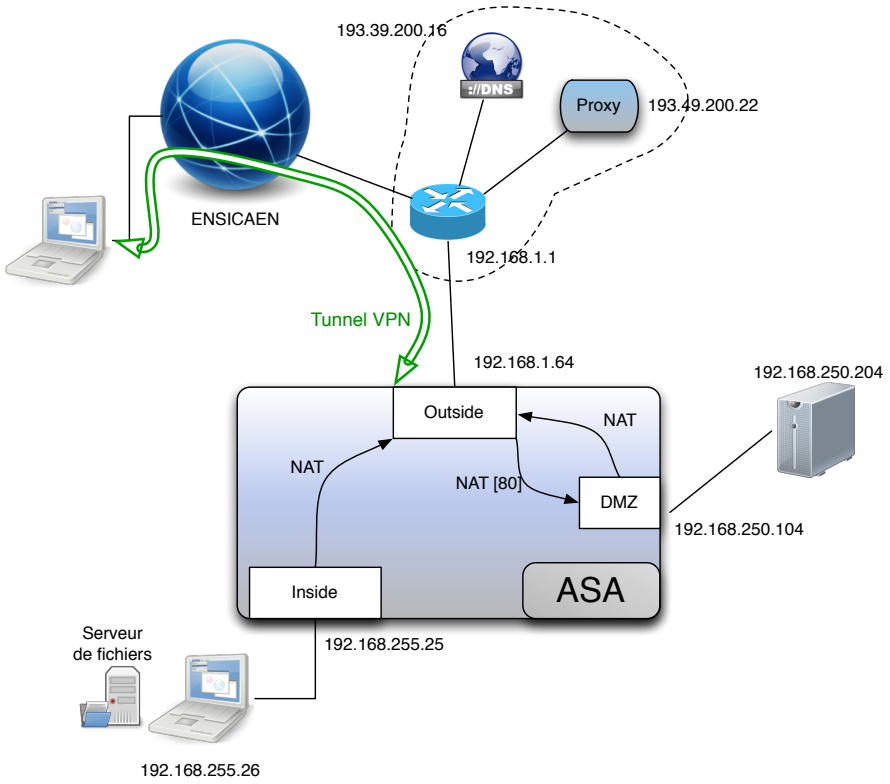
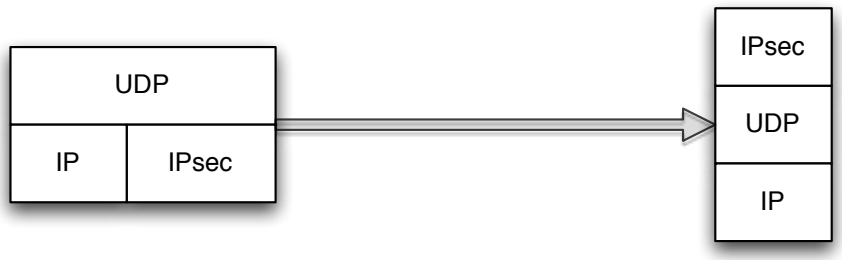


FIGURE 24 – Architecture du réseau avec mise en place du VPN

Afin de sécuriser nos transactions, nous allons utiliser le standard IPsec. N'étant pas forcément géré par tous les clients VPN, nous allons faire que sa couche soit placée au dessus de UDP (voir ci-dessous).



5.1 Présentation

Le principe du VPN est d'indiquer au PC de l'utilisateur quel chemin prendre pour appeler un serveur cible. Sa mise en place se déroule en plusieurs étapes :

1. Nous installons le VPN sur notre ASA (configuration détaillée par la suite)
2. L'utilisateur lance son client VPN et indique les différentes informations (pre_shared_key, login, password, ...).
3. La connexion de l'utilisateur entraîne l'envoie d'informations de la part de l'ASA (adresse IP allouée à la machine, masque, DNS, passerelle, ...).
4. L'ordinateur de l'utilisateur se charge ensuite de gérer la mise en place du VPN (exemple : simulation d'une nouvelle interface réseau avec modification des routes). Cela va créer un tunnel entre l'utilisateur et l'ASA. Il va être utilisé spécifiquement lorsqu'il y aura appel au serveur de fichiers.

Une autre information importante indiquant pourquoi nous utilisons IPsec sur UDP et non TCP. IPsec se charge de vérifier si les paquets ont bien été reçus, on dit qu'il est en mode connecté. TCP effectue les mêmes vérification alors que UDP est en mode déconnecté. Autrement dit, si nous mettons en place IPsec avec TPC, il y aurait une double vérification des paquets, ce qui est une perte de temps inutile.

5.2 Configuration

Nous allons maintenant configurer notre VPN. Pour cela, nous utilisons le « VPN Wizard » de l'interface de l'ASA.

Dans un premier temps, nous renseignons le type de VPN : Remote Access. En effet, c'est un utilisateur distant (hors des interfaces *inside*, *dmz* de l'ASA) qui va l'utiliser.

FIGURE 25 – Type de VPN

Ensuite, nous indiquons le type de logiciel client l'utilisateur va utiliser pour se connecter. Dans notre cas, il s'agit d'un VPN classique.

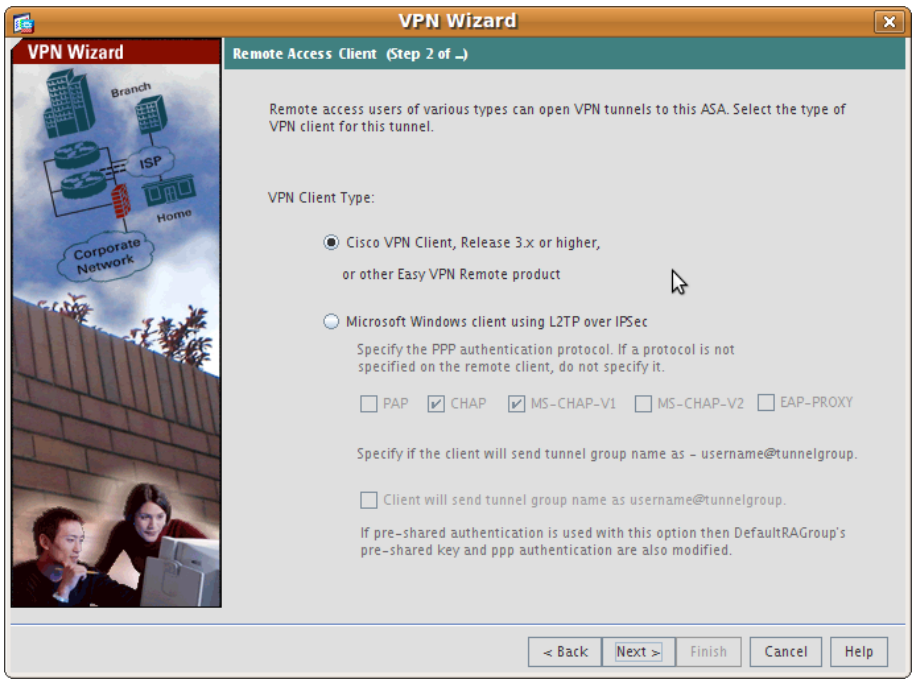


FIGURE 26 – Logiciel client

Nous configurons maintenant le type d'authentification utilisée. Nous aurions pu utiliser un certificat signé grâce au RSA. Dans notre cas, nous allons simplement partager un secret (authentification symétrique) qui sera une simple clé : « tpsecurite ». Nous définissons aussi un groupe, ceci permet de gérer les accès des personnes suivant le groupe auquel ils appartiennent.

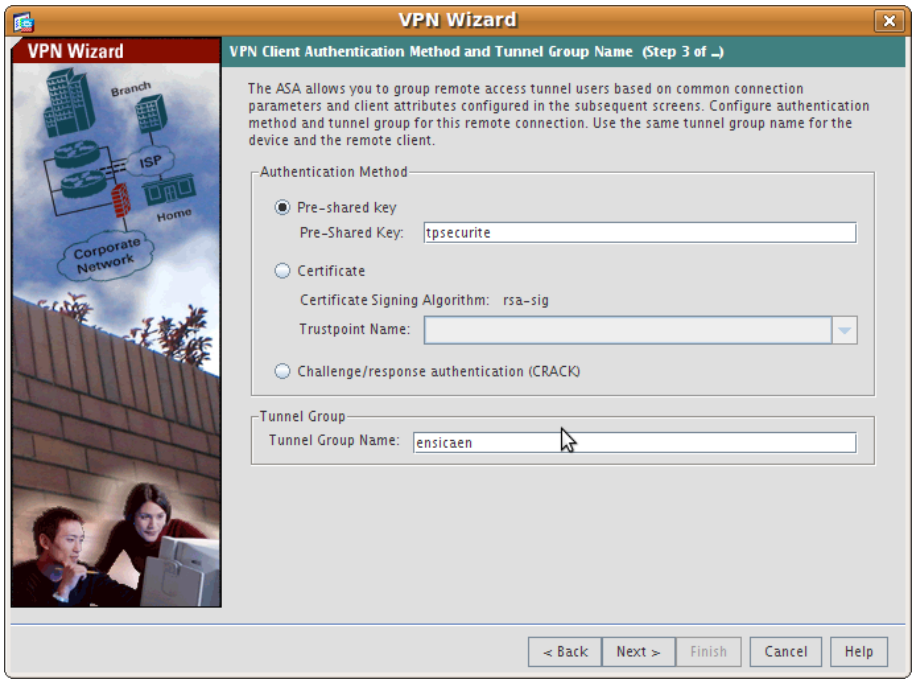


FIGURE 27 – Pre_shared_key et nom du groupe

Nous indiquons à partir de quelle source sont récupérées les données d'authentification du client. Dans notre cas, nous allons les créer au sein de l'ASA.

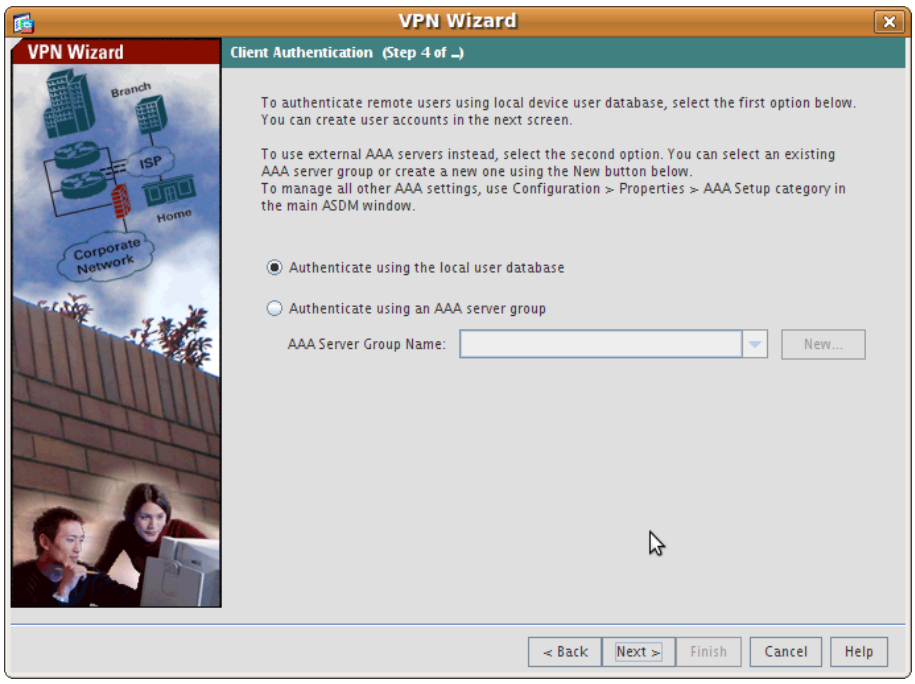


FIGURE 28 – Authentification du client

Nous ajoutons donc un nouvel utilisateur qui pourra utiliser le VPN. Son login et mot de passe sont simplement « ensicaen ».

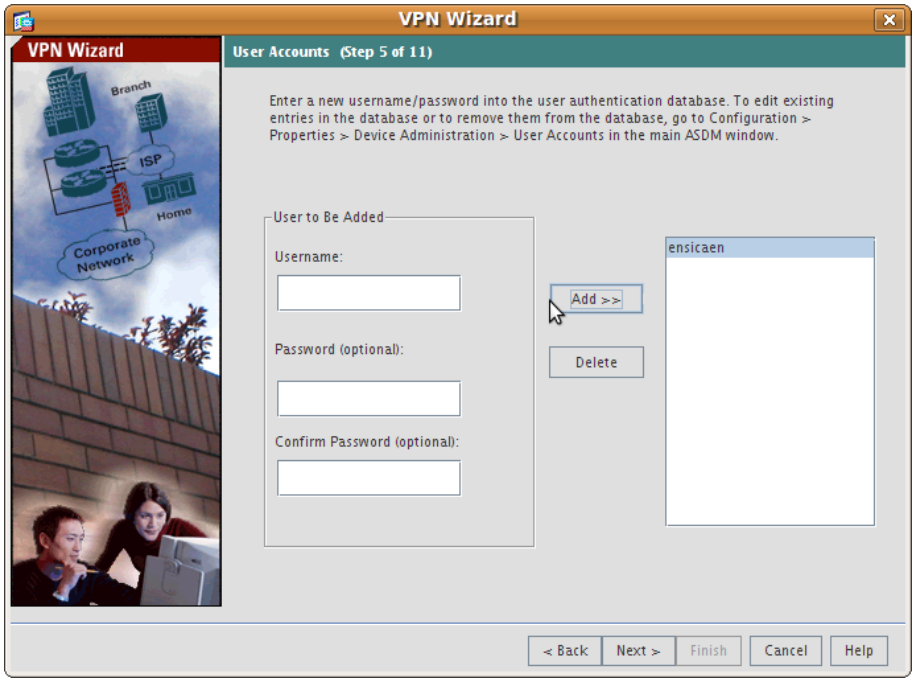


FIGURE 29 – Ajout d'un nouvel utilisateur

Nous devons aussi définir un pool d'adresses. Il s'agit en fait d'un espace d'adresses à partir duquel l'utilisateur s'en verra attribuer une. Comme nous n'avons pas encore définis avant, nous allons en créer un. Nous indiquons ainsi que le PC de l'utilisateur pourra se voir octroyer une adresse entre 192.168.64.1 et 192.168.64.10. Nous pouvons remarquer que le fait de commencer à la première adresse du réseau n'est pas naturel dans le monde du réseau. En effet, cette adresse est généralement attribué à la passerelle par défaut.

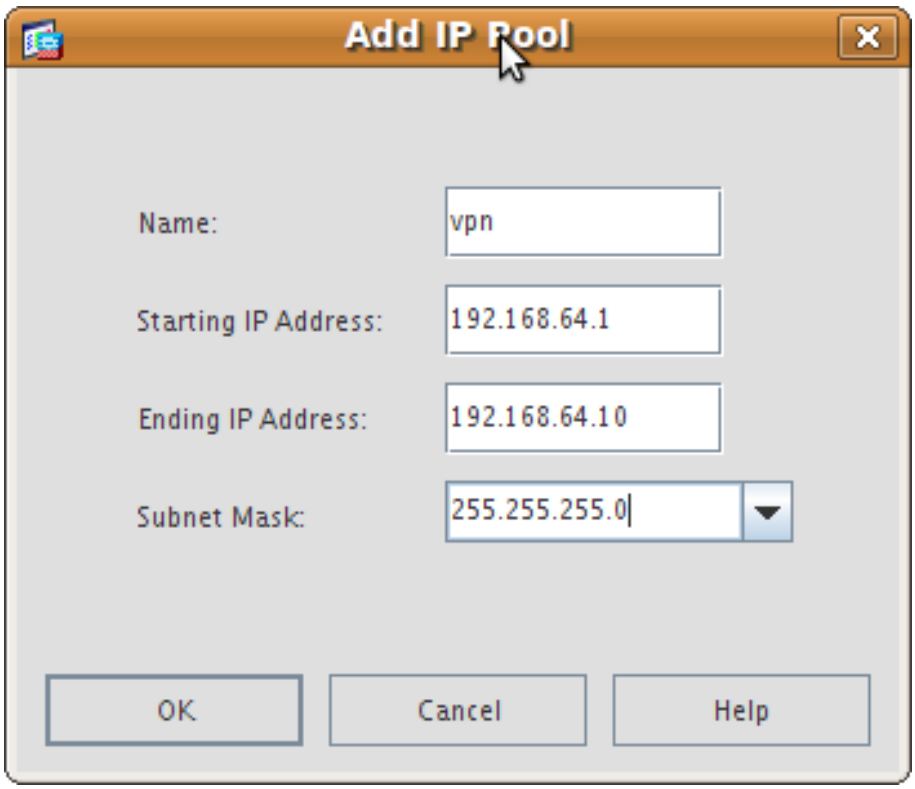


FIGURE 30 – Pool d'adresses définis

Une fois ce pool crée, nous l'attribuons au groupe que nous avons précédemment définis.

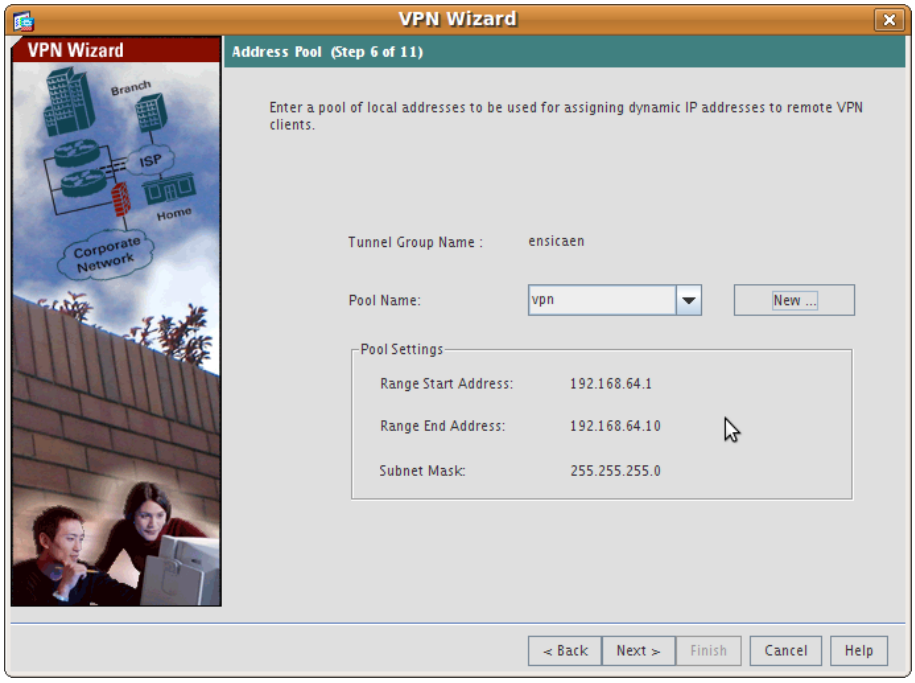


FIGURE 31 – Choix du pool d'adresses

Nous pouvons définir différentes informations plus ou moins utile suivant le contexte. Celles-ci seront transmises à l'utilisateur que son logiciel client VPN se chargera de prendre en compte. Dans notre cas, nous allons simplement définir l'adresse du serveur DNS.

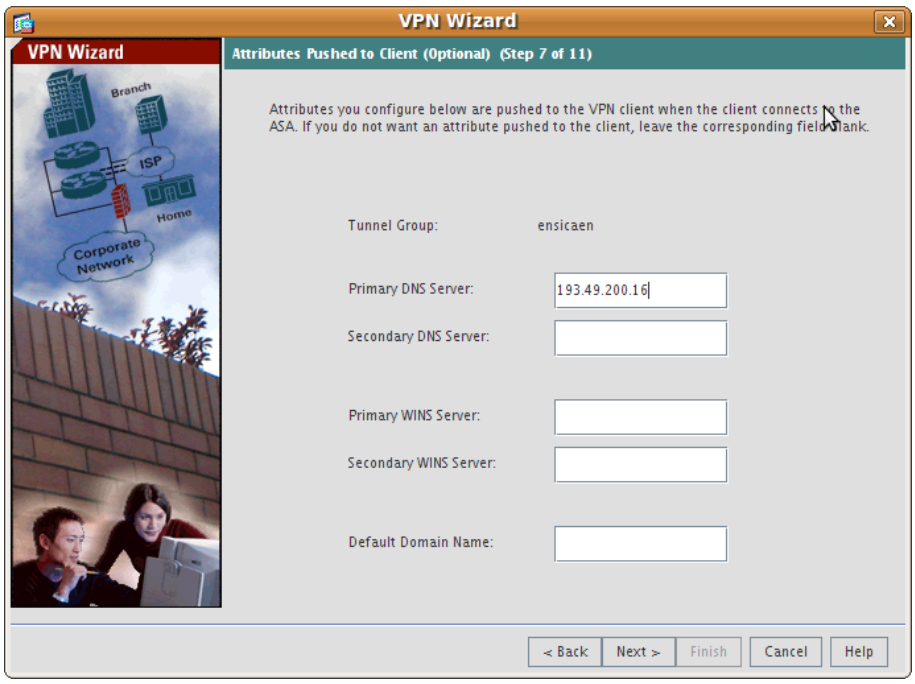


FIGURE 32 – Indication du serveur DNS

Maintenant que notre VPN est crée, il nous faut indiquer que les paquets reçus par l'utilisateur, connecté en VPN, seront transmis au sous-réseau relié à l'interface *inside*.

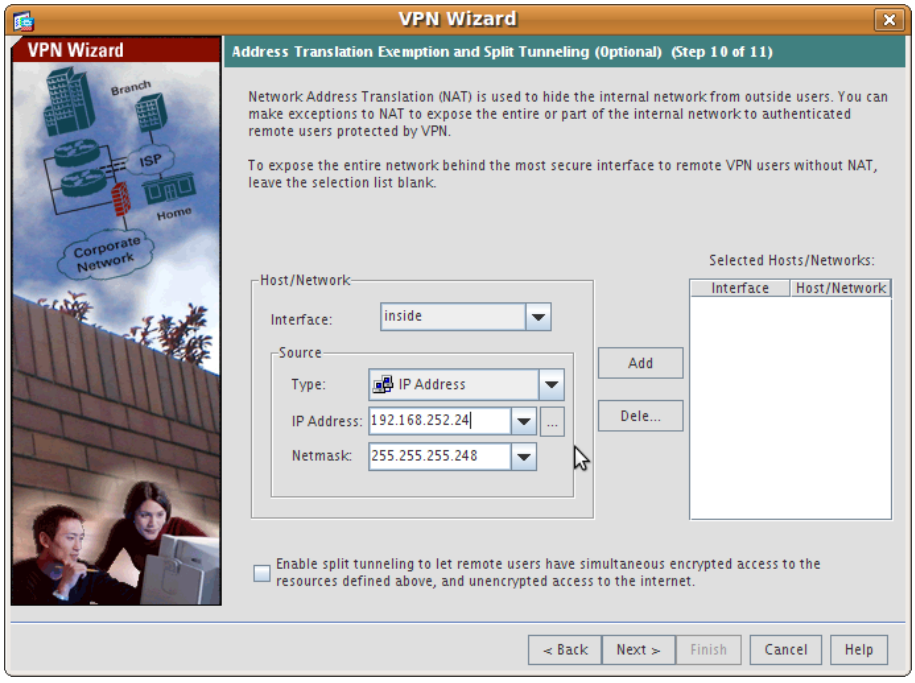


FIGURE 33 – Translation d'adresse vers la *inside*

Maintenant que la configuration est terminée, nous avons tester à partir d'un ordinateur distant. Il a fallu signaler le mode de fonctionnement utilisé pour IPsec, ainsi que la clé partagée. Une fois connecté, nous avons pu constater que le logiciel client avait mis en place une interface réseau. Celle-ci possédait l'adresse 192.168.64.1 et que la passerelle par défaut était 192.168.64.2 (non habituel en réseau). Nous ne pouvions cependant pas accéder à Internet.

Pour cela, nous avons mis en place une translation d'adresse sur l'interface *dmz* sauf que cette fois, nous autorisons l'utilisateur à accéder à Internet de façon non cryptée.

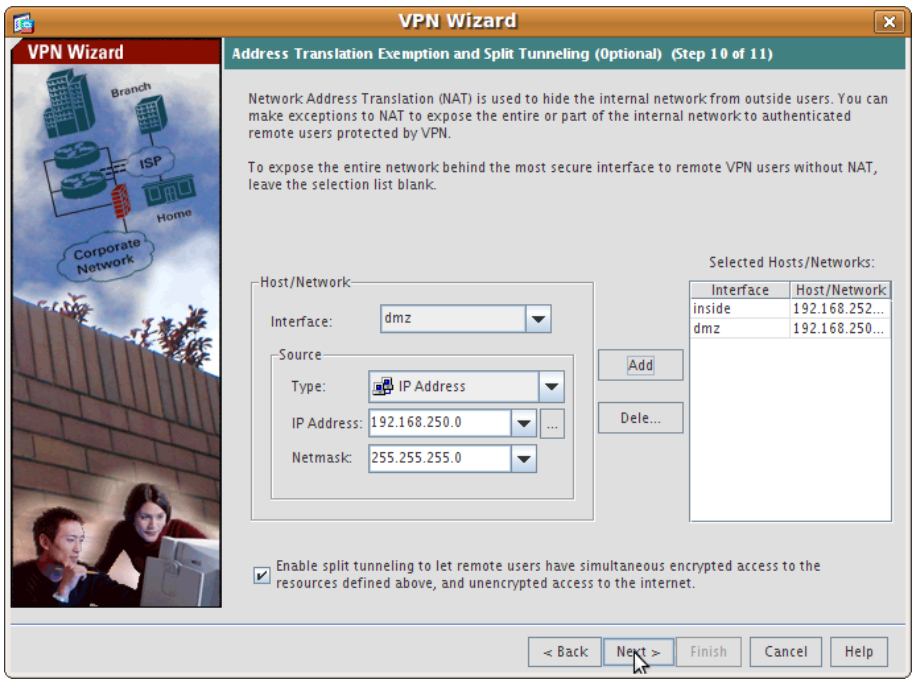


FIGURE 34 – Translation d'adresse vers la *dmz*

Après un nouveau test, cette fois, nous pouvions accéder à Internet. Nous avons pu constater que des routes ont été mises en place afin de définir où envoyer les paquets à destination de notre sous-réseau *inside*, ainsi qu'à destination de la *dmz*.

6 Outils

Dans cette section, nous allons voir différents outils permettant de mettre à jour des vulnérabilités du système ainsi que les attaques subis. N'ayant pas pu tester ces outils à l'ENSICAEN, nous les avons utiliser au sein d'un réseau privé. Tous les tests ont été effectués sous Mac OS X.

6.1 Wireshark

C'est un outil performant permettant de sniffer un réseau et d'afficher les paquets émis et reçu sous un format ergonomique. En effet, chaque paquet sont affichés et découpés selon les couches ISO avec présentation de nombre d'informations (flag, protocole, checksum, ...). De plus, celui-ci possède un outils de filtrage performant. Enfin, il inclut un grand nombre d'outils permettant l'analyse des paquets (encodage d'une conversation audio sur IP, possibilité de suivi d'une connexion TCP, ...).

6.2 nmap

Il s'agit d'un logiciel d'analyse de ports, services et un informations sur le système d'une ou plusieurs cibles. Il est ainsi possible de récupérer la liste des ports ouvert (et de ce fait les applications liées) pouvant être la cible de potentielles attaques. Plusieurs scénarios d'attaque sont disponibles, exemple attaque de tout un sous réseau afin d'en comprendre son architecture mais aussi d'y découvrir des vulnérabilités. Pour notre exemple, nous allons dans un premier temps récupérer la liste de tous les ordinateurs de notre réseau à l'aide de la commande ci-dessous :

```
1 nmap -PN -sL 192.168.1.0/24
2 -PN : Permet de considerer que toutes les cibles sont en
     lignes + certaines machines bloquent les pings, cet
     parametre permet de d'outrepasser cette limite
3 -sL : Indique que l'on souhaite juste lister toutes les
     machines du reseau
4 -192.168.1.0/24 : Reseau a analyser
```

Nous récupérons une liste de machines (nom d'hôte) ainsi que leur adresse IP associée.

Nous allons maintenant choisir une cible et essayer d'obtenir plus d'informations. Ci dessous la commande à utiliser ainsi que le résultat de celle-ci.

```
~ -> nmap -PN -A --script all 192.168.1.89

Starting Nmap 5.21 ( http://nmap.org ) at 2011-02-12 14:09 CET
Nmap scan report for [redacted] (192.168.1.89)
Host is up (0.039s latency).
Not shown: 993 filtered ports
PORT      STATE SERVICE     VERSION
135/tcp   open  msrpc       Microsoft Windows RPC
139/tcp   open  netbios-ssn
445/tcp   open  netbios-ssn
554/tcp   open  rtsp?
2869/tcp  open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-enum:
5357/tcp  open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_http-date: Sat, 12 Feb 2011 13:10:52 GMT; -6s from local time.
|_html-title: Service Unavailable
| http-headers:
|   Content-Type: text/html; charset=us-ascii
|   Server: Microsoft-HTTPAPI/2.0
|   Date: Sat, 12 Feb 2011 13:10:53 GMT
|   Connection: close
|   Content-Length: 326
|
|_  (Request type: GET)
|_http-enum:
10243/tcp open  http        Microsoft HTTPAPI httpd 2.0 (SSDP/UPnP)
|_html-title: Not Found
|_http-date: Sat, 12 Feb 2011 13:10:52 GMT; -6s from local time.
| http-headers:
|   Content-Type: text/html; charset=us-ascii
|   Server: Microsoft-HTTPAPI/2.0
|   Date: Sat, 12 Feb 2011 13:10:53 GMT
|   Connection: close
|   Content-Length: 315
|
|_  (Request type: GET)
|_http-enum:
Service Info: OS: Windows

Host script results:
|_nbstat: NetBIOS name: [redacted], NetBIOS user: <unknown>, NetBIOS MAC: [redacted]
|_smbv2-enabled: Server supports SMBv2 protocol
| smb-brute:
|_  No accounts found
|_smb-server-stats:
| smb-security-mode:
|   User-level authentication
|   SMB Security: Challenge/response passwords supported
|_  Message signing disabled (dangerous, but default)

Service detection performed. Please report any incorrect results at http://nmap.org/submit/ .
Nmap done: 1 IP address (1 host up) scanned in 187.55 seconds
```

FIGURE 35 – Commande nmap et résultat

- -A permet d'afficher des informations sur le système d'exploitation de la cible.
- –script all : Indique que nous souhaitons lancer la totalité des scripts sur la cible afin de récupérer un maximum d'informations
- 192.168.1.89 : Cible de l'attaque

Pour ce qui est des résultats, dans un premier nous avons des informations sur les ports de la cible. Nous pouvons voir la liste des ports ouverts ainsi que les applications liées. On peut distinguer plusieurs ports ouverts pour le protocole HTTP, ils sont donc listés avec les entêtes.
Ensuite, nous avons des informations sur le service smb (mode de sécurité, compte existant, ...).

6.3 Nessus

Cette application permet de mettre en évidence un certain nombre de failles sur un système. Il fonctionne à l'aide d'un serveur et d'un client. Nous lançons dans un premier temps le serveur et créons un compte client. Puis, nous lançons le client qui s'y connecte. C'est au sein de se dernier que nous pouvons définir un certain nombre de paramètres quand aux « attaques » à effectuer. Tel que ne tester que les attaques de serveur Apache mais aussi indiquer quelle sont les machines à analyser.

Remarque : Le scan Nessus n'est pas lié au scan nmap précédent (cible différente).

Nous lançons donc notre serveur Nessus

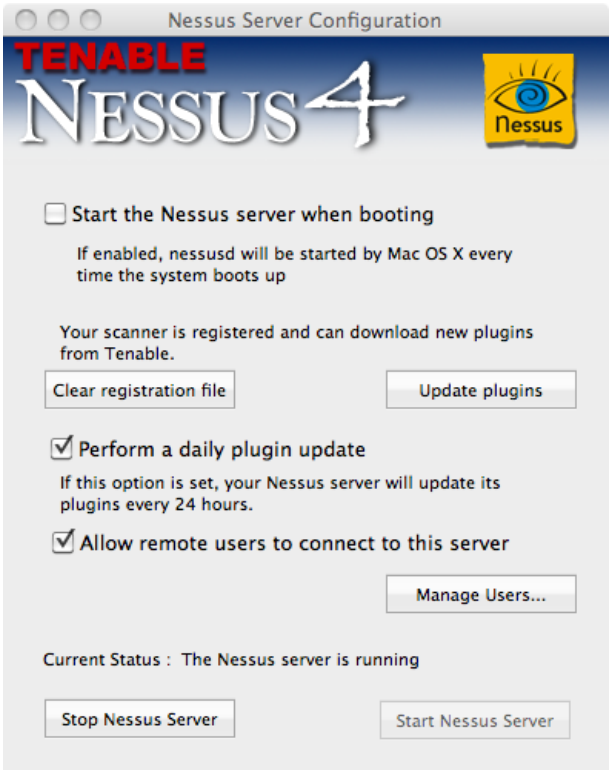


FIGURE 36 – Serveur Nessus

L'application cliente Nessus que nous avons utilisé fonctionne au sein d'un navigateur. Une fois rentrée l'adresse fournie, nous arrivons sur une page de login. Nous nous connectons à l'aide d'un utilisateur que nous avons précédemment renseigné au sein du serveur.

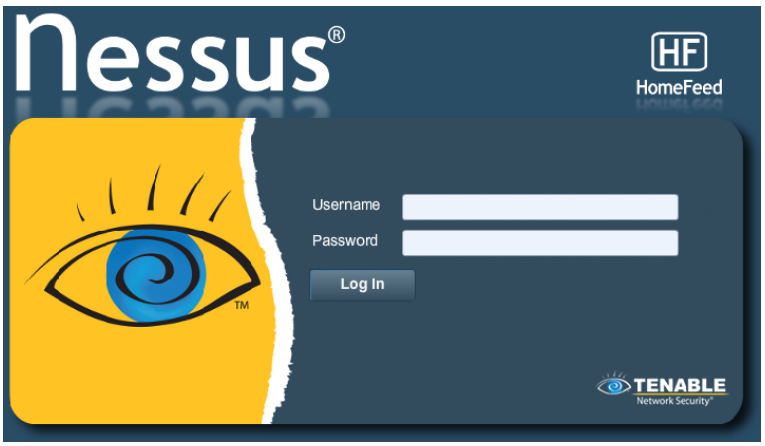


FIGURE 37 – Connexion à Nessus

Suite à cette connexion, nous accédons à l'accueil permettant de gérer et utiliser Nessus.

FIGURE 38 – Accueil Nessus

- Policy : Définition de règles (type d'attaques à effectuer, plugin à activer/désactiver, ...)
- Scans : Etat des scans en cours. Lors de la création d'un scan, nous lui lions une règle Policy.
- Reports : Contient les rapports de scans.

6.4 Ajout d'une policy

Dans un premier temps, nous allons créer une policy d'attaque. Celles-ci sont découpées en 4 parties :

- General : Permet de définir des paramètres générales sur la règle d'attaques (nom, description, ports à scanner, ...)
- Credentials : Permet de définir des comptes à tester sur la cible (compte SMB, SSH, ...)
- Plugins : Nessus fonctionne à l'aide de plugin que l'on peut activer ou désactiver suivant la cible à attaquer. Chaque plugin correspond à une attaque spécifique.
- Preferences : Permet de fournir des informations supplémentaire (identifiant base de données, s'il faut ou non scanner du matériel fragile,...)

Dans un premier temps, nous configurons la partie General. Nous indiquons que nous souhaitons tester tous les ports.

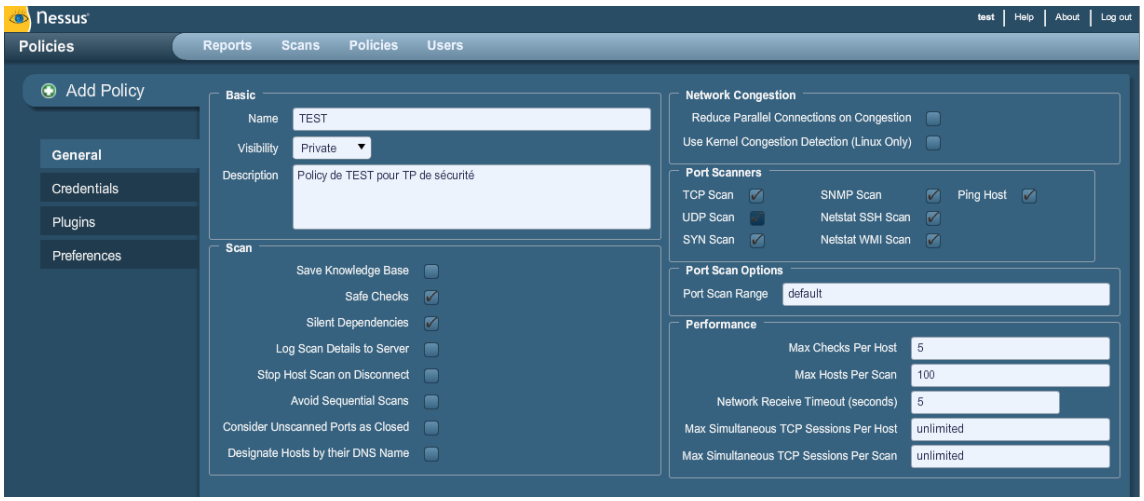


FIGURE 39 – Création d'une policy - General

Pour la partie crédentials, nous laissons les données par défaut.

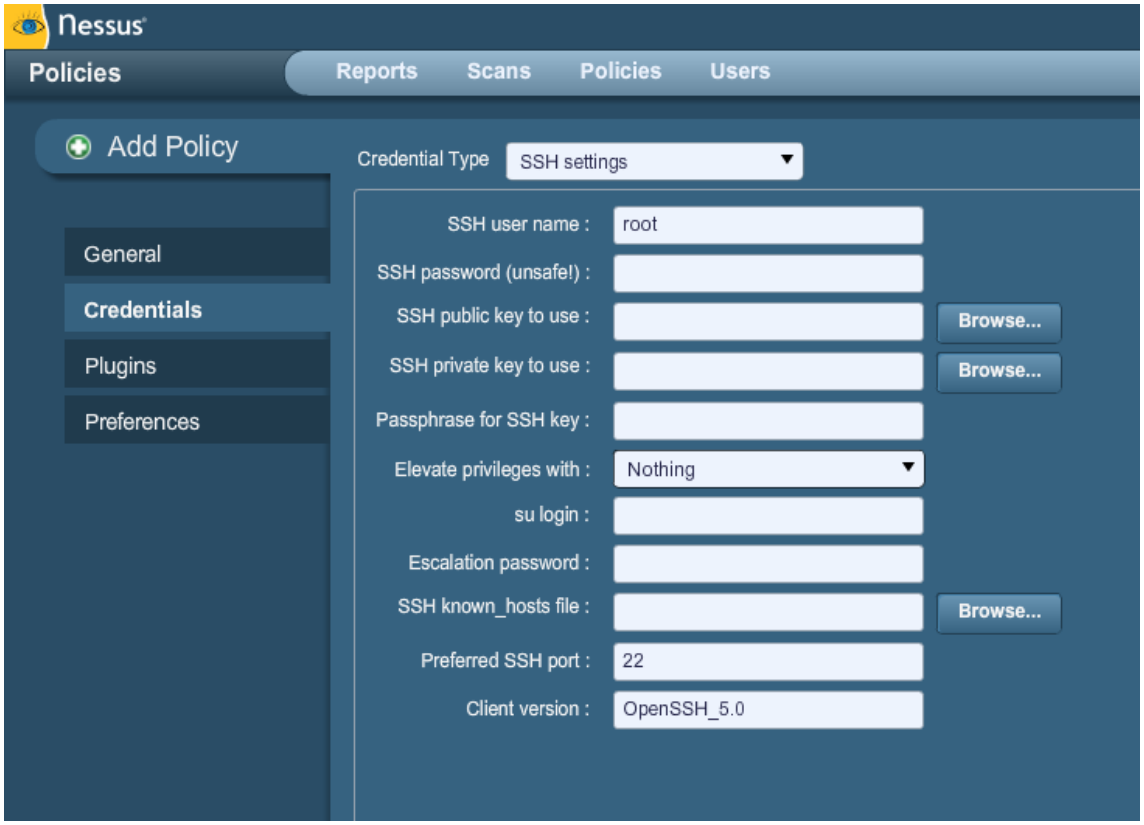


FIGURE 40 – Création d'une policy - Credentials

Nous ne faisons pas de restrictions au niveau des plugins et activons tout. Tous ne sont pas nécessaire en sachant que nous attaquons un ordinateur personnel et non un serveur Apache par exemple.

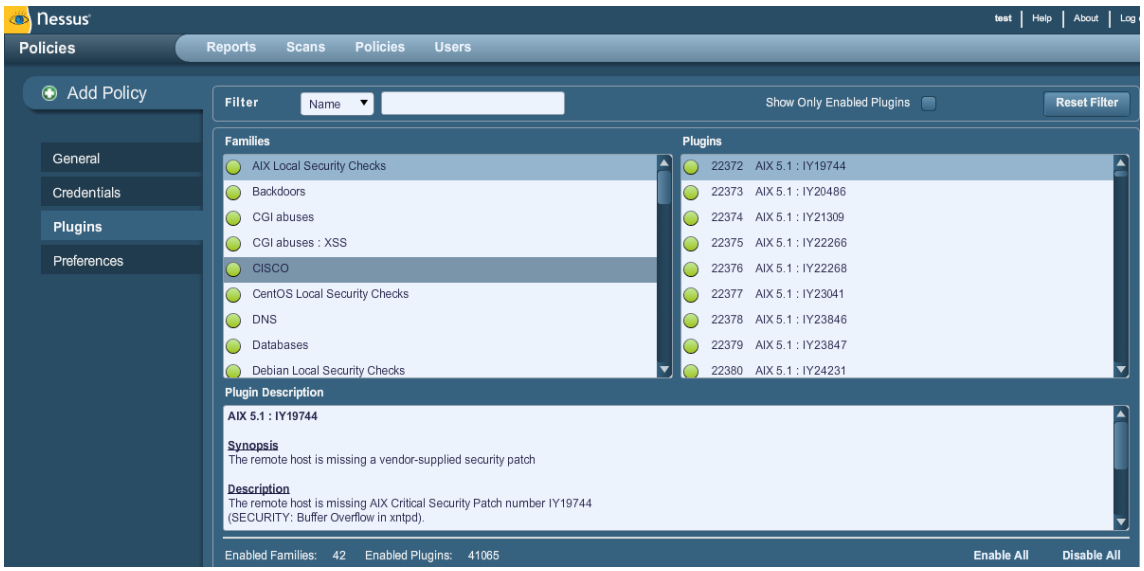


FIGURE 41 – Création d'une policy - Plugins

Enfin, nous laisson les préférences par défaut (pas d'utilisation de base de données, ...)

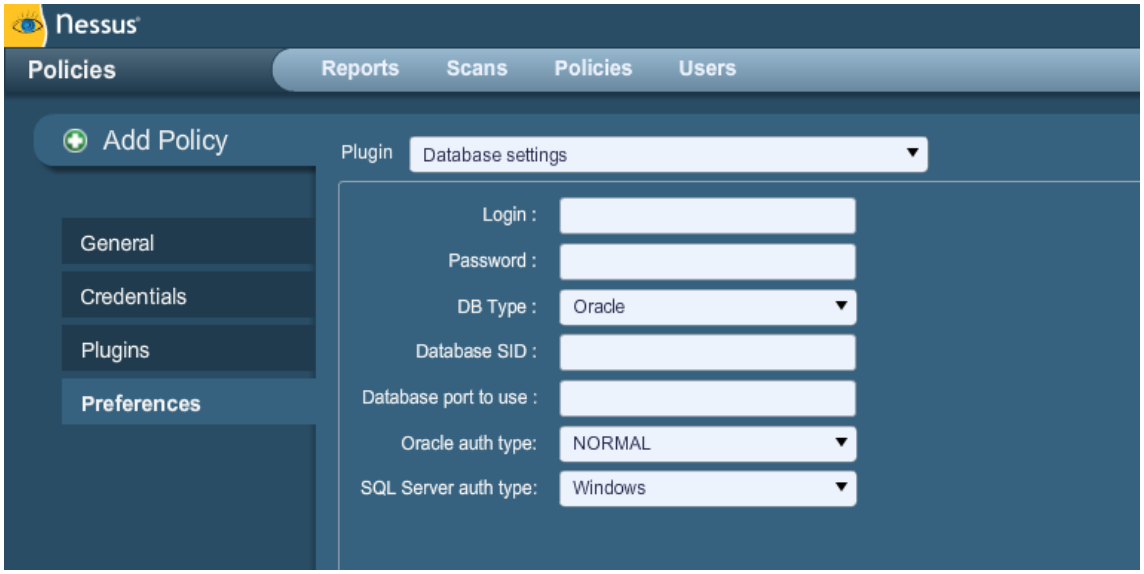


FIGURE 42 – Création d'une policy - Preferences

6.5 Ajout et lancement d'un scan

Notre policy crée, nous allons maintenant pouvoir lancer un scan l'utilisant. Lors de l'ajout, nous devons renseigner un nom, la date de lancement, la policy à utiliser et, enfin, les cibles de ce scan.

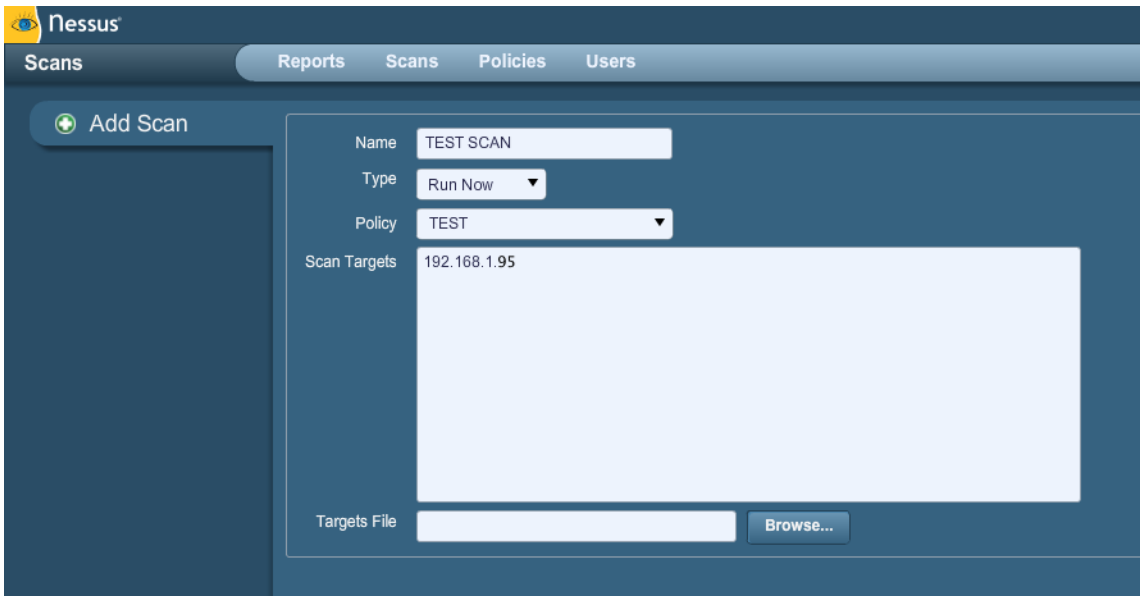


FIGURE 43 – Ajout d'un scan

6.6 Résultat du scan

Une fois le scan terminé, nous pouvons aller dans la section report afin de voir le résultat. Nous sélectionnons ainsi le scan puis la machine cible que nous avons analysé. Les résultats sont ensuite organisé par protocole puis par port. Pour chaque protocole, il est indiqué le nombre de failles importantes, moyennes et faibles.

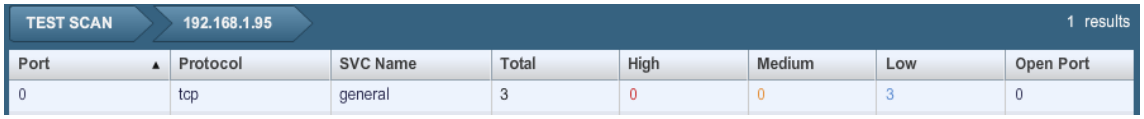


FIGURE 44 – Liste des protocoles possédant des failles

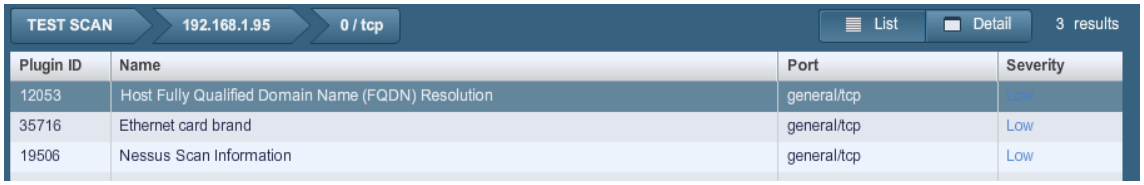


FIGURE 45 – Liste des failles du protocole TCP

Nous pouvons ensuite sélectionner une faille spécifique afin de voir à quoi celle-ci correspond. Pour la faille ci-dessous, Nessus indique qu'il a réussi à obtenir le nom complet de la cible. Dans ce cas, il n'y a aucun risque, il n'est donc pas nécessaire de prévoir quelque chose pour corriger cette faille. Il n'y a donc aucune solution d'indiquée.

FIGURE 46 – Précision d'une faille

Dans l'exemple ci-dessous (scan effectué sur une autre machine), on peut voir apparaître une faille important lié à l'application VMware. Il est indiqué qu'un pirate peut utiliser différentes méthodes pour s'introduire dans le système et gagner un accès privilégier (root). On peut, cette fois, voir apparaître une solution pour résoudre ce problème. Ici, il s'agit de simplement mettre à jour le logiciel.

FIGURE 47 – Faille VMware

6.7 snort

Il s'agit d'un logiciel permettant de détecter les tentatives d'intrusions au sein de notre système. Pour cela, il faut définir un certain nombre de paramètres au sein d'un fichier de configuration puis, d'indiquer les règles que nous souhaitons mettre en route (analyser HTTP, SMTP, ...). Il a l'avantage de fournir plusieurs possibilité de log des infos (fichier texte, base de données, évènements récupérable à partir d'autres applications, ...).

7 Analyse de fichier log

Pour observer le contenu du fichier, nous avons utiliser Wireshark.

Suite à l'analyse du fichier log, nous avons pu déterminer que l'adresse de l'attaquant est 192.168.0.9 et celle de la cible 192.168.0.99. En effet, tous les paquets de demande de connexion sont envoyé de l'attaquant vers la cible.

On peut voir qu'il s'agit un scan *SYN*. En effet, l'attaque effectue des demandes de connexion *SYN* sur chaque port qu'il souhaite analyser. Si la cible répond *[SYN, ACK]* c'est que le port est ouvert, s'il répond *[RST, ACK]* c'est qu'il ne l'ait pas. Afin de connaître les ports ouverts de la cible, il ne nous fallait plus qu'à effectuer un filtre sur les paquets affichés. Une fois ceci effectué, nous avons pu discerner que les ports SSH, SunRPC, filenet-tms, HTTP, HTTPS et DOMAIN étaient ouverts.

8 Annexe

8.1 Configuration firewall

```
: Saved
:
ASA Version 7.2(3)
!
hostname ciscoasa
domain-name ensicaen.fr
enable password 8Ry2YjIyt7RRXU24 encrypted
names
!
interface Vlan1
 nameif inside
 security-level 100
 ip address 192.168.252.25 255.255.255.248
!
interface Vlan2
 nameif outside
 security-level 0
 ip address 192.168.1.64 255.255.255.0
!
interface Vlan3
 no forward interface Vlan1
 nameif dmz
 security-level 50
 ip address 192.168.250.104 255.255.255.0
!
interface Ethernet0/0
 switchport access vlan 2
!
interface Ethernet0/1
!
interface Ethernet0/2
!
interface Ethernet0/3
!
interface Ethernet0/4
!
interface Ethernet0/5
!
interface Ethernet0/6
!
interface Ethernet0/7
 switchport access vlan 3
!
passwd 2KFQnbNIdI.2KYOU encrypted
ftp mode passive
dns server-group DefaultDNS
```

```
47  domain-name ensicaen.fr
48 same-security-traffic permit inter-interface
49 same-security-traffic permit intra-interface
50 access-list outside_access_in extended permit icmp host
      192.168.1.1 interface outside
51 access-list outside_access_in extended permit tcp any
      host 192.168.250.204 eq ssh
52 access-list outside_access_in extended permit tcp any any
       eq www
53 access-list inside_access_in extended permit udp
      192.168.252.24 255.255.255.248 host 193.49.200.16 eq
      domain
54 access-list inside_access_in extended permit tcp
      192.168.252.24 255.255.255.248 host 193.49.200.16 eq
      domain
55 access-list inside_access_in extended permit tcp
      192.168.252.24 255.255.255.248 any eq ssh
56 access-list inside_access_in extended permit tcp
      192.168.252.24 255.255.255.248 any eq www
57 access-list inside_access_in extended permit tcp
      192.168.252.24 255.255.255.248 host 193.49.200.22 eq
      3128
58 access-list inside_access_in extended permit icmp any any
59 access-list inside_nat0_outbound extended permit ip
      192.168.252.24 255.255.255.248 host 192.168.250.204
60 access-list inside_nat0_outbound extended permit ip
      192.168.252.24 255.255.255.248 192.168.64.0
      255.255.255.240
61 access-list dmz_access_in extended permit tcp host
      192.168.250.204 host 193.49.200.16 eq domain
62 access-list dmz_access_in extended permit tcp host
      192.168.250.204 any eq www
63 access-list dmz_access_in extended permit tcp host
      192.168.250.204 host 193.49.200.22 eq 3128
64 access-list dmz_access_in extended permit tcp host
      192.168.250.204 any eq ssh
65 access-list dmz_access_in extended permit udp host
      192.168.250.204 host 193.49.200.16 eq domain
66 access-list dmz_access_in extended permit icmp any any
67 access-list tpsecualt_splitTunnelAcl standard permit
      192.168.252.24 255.255.255.248
68 access-list tpsecualt_splitTunnelAcl standard permit
      192.168.250.0 255.255.255.0
69 access-list dmz_nat0_outbound extended permit ip
      192.168.250.0 255.255.255.0 192.168.64.0
      255.255.255.240
70 pager lines 24
71 logging enable
72 logging asdm informational
73 mtu inside 1500
```

```
74 mtu outside 1500
75 mtu dmz 1500
76 ip local pool vpn 192.168.64.1-192.168.64.10 mask
      255.255.255.0
77 ip local pool poolvpntpsecualt 192.168.84.2-192.168.84.9
      mask 255.255.255.0
78 icmp unreachable rate-limit 1 burst-size 1
79 asdm image disk0:/asdm-523.bin
80 no asdm history enable
81 arp timeout 14400
82 global (outside) 1 interface
83 nat (inside) 0 access-list inside_nat0_outbound
84 nat (inside) 1 192.168.252.24 255.255.255.248
85 nat (dmz) 0 access-list dmz_nat0_outbound
86 nat (dmz) 1 192.168.250.204 255.255.255.255
87 static (dmz,outside) tcp interface www 192.168.250.204
      www netmask 255.255.255.255
88 static (dmz,dmz) tcp interface ssh 192.168.250.204 ssh
      netmask 255.255.255.255
89 access-group inside_access_in in interface inside
90 access-group outside_access_in in interface outside
91 access-group dmz_access_in in interface dmz
92 route outside 0.0.0.0 0.0.0.0 192.168.1.1 1
93 timeout xlate 3:00:00
94 timeout conn 1:00:00 half-closed 0:10:00 udp 0:02:00 icmp
       0:00:02
95 timeout sunrpc 0:10:00 h323 0:05:00 h225 1:00:00 mgcp
      0:05:00 mgcp-pat 0:05:00
96 timeout sip 0:30:00 sip_media 0:02:00 sip-invite 0:03:00
      sip-disconnect 0:02:00
97 timeout uauth 0:05:00 absolute
98 http server enable
99 http 192.168.252.24 255.255.255.248 inside
100 no snmp-server location
101 no snmp-server contact
102 snmp-server enable traps snmp authentication linkup
      linkdown coldstart
103 crypto ipsec transform-set ESP-3DES-SHA esp-3des esp-sha-
      hmac
104 crypto dynamic-map outside_dyn_map 20 set pfs
105 crypto dynamic-map outside_dyn_map 20 set transform-set
      ESP-3DES-SHA
106 crypto dynamic-map outside_dyn_map 40 set pfs
107 crypto dynamic-map outside_dyn_map 40 set transform-set
      ESP-3DES-SHA
108 crypto map outside_map 65535 ipsec-isakmp dynamic
      outside_dyn_map
109 crypto map outside_map interface outside
110 crypto isakmp enable outside
111 crypto isakmp policy 10
```

```
 authentication pre-share
 encryption 3des
 hash sha
 group 2
 lifetime 86400
telnet timeout 5
ssh timeout 5
console timeout 0
dhcpd auto_config outside
!
dhcpd address 192.168.252.26-192.168.252.30 inside
!

!
class-map inspection_default
 match default-inspection-traffic
!
!
policy-map type inspect dns preset_dns_map
 parameters
  message-length maximum 512
policy-map global_policy
 class inspection_default
  inspect dns preset_dns_map
  inspect ftp
  inspect h323 h225
  inspect h323 ras
  inspect rsh
  inspect rtsp
  inspect esmtp
  inspect sqlnet
  inspect skinny
  inspect sunrpc
  inspect xdmcp
  inspect sip
  inspect netbios
  inspect tftp
!
service-policy global_policy global
group-policy tpsecualt internal
group-policy tpsecualt attributes
 dns-server value 193.49.200.16
 vpn-tunnel-protocol IPSec
 split-tunnel-policy tunnelspecified
 split-tunnel-network-list value tpsecualt_splitTunnelAcl
 default-domain value ensicaen.fr
group-policy ensicaen internal
group-policy ensicaen attributes
 dns-server value 193.49.200.16
 vpn-tunnel-protocol IPSec
```

```
username ensicaen password IFOoXQbk3nUfUVkR encrypted
   privilege 0
username ensicaen attributes
 vpn-group-policy ensicaen
tunnel-group ensicaen type ipsec-ra
tunnel-group ensicaen general-attributes
 address-pool vpn
 default-group-policy ensicaen
tunnel-group ensicaen ipsec-attributes
 pre-shared-key *
tunnel-group tpsecualt type ipsec-ra
tunnel-group tpsecualt general-attributes
 address-pool vpn
 default-group-policy tpsecualt
tunnel-group tpsecualt ipsec-attributes
 pre-shared-key *
prompt hostname context
Cryptochecksum:f01ca91930c6d893183e634f806b37b2
: end
asdm image disk0:/asdm-523.bin
no asdm history enable
```