Héberger un site Web chez soi

Par François ALLAIS

Sommaire

[1. Introduction 3](#_Toc295469252)

[1.1. Architecture globale 3](#_Toc295469253)

[2. Le routeur 4](#_Toc295469254)

[3. DMZ 4](#_Toc295469255)

[3.1. Cas particulier 5](#_Toc295469256)

[3.2. Remarques 5](#_Toc295469257)

[4. Serveur 6](#_Toc295469258)

[4.1. Configuration IP 6](#_Toc295469259)

[4.2. Apache 6](#_Toc295469260)

[4.3. Journalisation des évènements 8](#_Toc295469261)

[4.4. Pare-feu 8](#_Toc295469262)

[4.5. Vérification 9](#_Toc295469263)

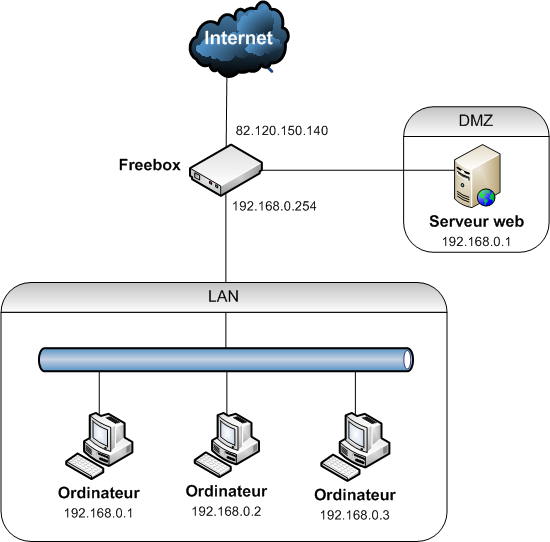
# Introduction

Aujourd’hui, beaucoup de personnes développent des sites Web chez eux en freelance ou simplement pour le plaisir mais se heurtent aux problèmes de l’hébergement. Le projet doit être accessible à tous les développeurs, sécurisé, extensible et tout cela à moindre cout. C’est la raison pour laquelle beaucoup se tournent vers l’hébergement à domicile.

Cette solution est tout à fait possible puisqu’elle est l’objet de ce tutoriel qui proposera un serveur sécurisé, accessible et journalisé.

## Architecture globale

Le schéma suivant présente l’architecture qui va être réalisée dans ce tutoriel.

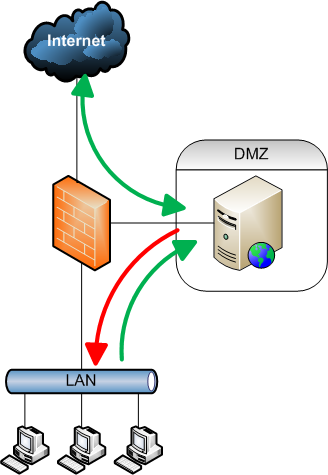


# Le routeur

Pour réaliser ce tutoriel, un routeur est nécessaire. Il peut être matériel, ou inclus dans les modem vendus sur le marché. Si tel est le cas, il doit être activé. Consultez la FAQ de votre fournisseur d’accès pour activer cette fonction.

# DMZ

Une DMZ, *Demilitarized Zone* en anglais, va permettre d’isoler le serveur Web de votre LAN tout en le rendant accessible depuis Internet. Ce concept est largement répandu dans la sécurité informatique car concrètement si le serveur web est compromis, le réseau local reste protégé.



**Fonctionnement d’une DMZ**

La plupart des FAI proposent des modems qui disposent de la fonction DMZ, mais également les équipements réseaux disponibles sur le marché. Pour savoir si votre équipement dispose de cette fonctionnalité, il suffit de se rendre sur la page d’administration et aller dans les fonctions relatives au routage. Un onglet ou un champ nommé DMZ doit s’y trouver.

Si tel est le cas, vous devez renseigner une adresse IP qui sera utilisée par le serveur dans la suite du tutoriel. Utilisez par exemple l’adresse **192.168.0.1**.

## Cas particulier

Si votre modem ou votre équipement ne propose pas la fonction DMZ, vous pouvez procéder autrement grâce à une astuce. En effet, il vous suffit de rediriger l’intégralité des ports (1 à 65535) vers le serveur, donc vers l’adresse 192.168.0.1.

## Remarques

Il faut absolument veiller à ce que l’adresse de la DMZ ne soit pas comprise entre les intervalles du DHCP. En effet, si le serveur Web vient à être arrêté, l’adresse de la DMZ pourrait être attribuée à un ordinateur qui ne serait pas le serveur Web et cela serait dramatique.

Vous pouvez par exemple fixer le DHCP comme ceci :

* Début : 192.168.0.10
* Fin : 192.168.0.20

# Serveur

Dans ce tutoriel, le serveur fonctionnera avec Linux, plus particulièrement la distribution Ubuntu. Notez que le serveur peut tout à fait être virtuel à l’aide de solutions telles que VMWare (payante) ou VirtualBox (gratuite). Dans ce cas, d’un point de vue conceptuel, l’architecture reste la même mais vous économisez une machine.

## Configuration IP

Le serveur doit avoir une IP statique de valeur identique à celle renseignée dans le routeur, dans notre exemple **192.168.0.1** (Masque 255.255.255.0 et Passerelle 192.168.0.254). Ainsi toutes les nouvelles connexions entrantes seront redirigées vers le serveur Web.

## Apache

Si vous lisez ce tutoriel, il est fort probable que PHP, MySQL et apache soit installés sur votre serveur. Si ce n’est pas le cas, la documentation d’Ubuntu à ce sujet est très complète.

### Le fichier httpd.conf

Pour bien comprendre le mécanisme mis en place, nous allons configurer deux sites Web. Le premier site web est un projet sur les pommes, il est en phase de test et doit donc être accessible par tous les membres du projet et les bêta-testeurs. Le deuxième site est un projet sur les bananes, il commence à peine et doit uniquement être accessible par les utilisateurs du LAN.

Les besoins en terme de sécurité sont résumés dans ce tableau :

|  |  |  |  |
| --- | --- | --- | --- |
|  | LAN | Locale | Extérieur |
| Site n°1 (apple) | check, confirm, ok, yes icon | check, confirm, ok, yes icon | check, confirm, ok, yes icon |
| Site n°2 (banana) | check, confirm, ok, yes icon | check, confirm, ok, yes icon | close, closing, delete, dialog, disabled, exit, no, red x, x icon |

|  |
| --- |
| **# Site n°1 (apple)**  **# Déclaration des serveurs virtuels**  NameVirtualHost 127.0.0.1:8080  NameVirtualHost 192.168.0.1:8080  **# Déclaration des cibles d’écoute**  Listen 127.0.0.1:8080  Listen 192.168.0.1:8080  # Locale  <VirtualHost 127.0.0.1:8080>  DocumentRoot "/home/websites/apple/www"  DirectoryIndex index.html  **CustomLog /var/log/websites/apple.log combined**  <Directory "/home/websites/apple/www">  AllowOverride All  Allow from All  </Directory>  </VirtualHost>  # DMZ  <VirtualHost 192.168.0.1:8080>  DocumentRoot "/home/websites/apple/www"  DirectoryIndex index.html  **CustomLog /var/log/websites/apple.log combined**  <Directory "/home/websites/apple/www">  AllowOverride All  Allow from All  </Directory>  </VirtualHost>  **# Site n°2**  NameVirtualHost 127.0.0.1:8081  NameVirtualHost 192.168.0.1:8081  Listen 127.0.0.1:8081  Listen 192.168.0.1:8081  <VirtualHost 127.0.0.1:8081>  DocumentRoot "/home/websites/banana/www"  DirectoryIndex index.html  **CustomLog /var/log/websites/banana.log combined**  <Directory "/home/websites/banana/www">  AllowOverride All  Allow from All  </Directory>  </VirtualHost>  <VirtualHost 192.168.0.1:8081>  DocumentRoot "/home/websites/banana/www"  DirectoryIndex index.html  **CustomLog /var/log/websites/banana.log combined**  <Directory "/home/websites/banana/www">  AllowOverride All  Allow from All  </Directory>  </VirtualHost> |

Pensez à créer les deux répertoires **apple** et **banana** dans **/home**. Chacun de ces répertoires doit contenir un fichier index.html contenant un petit message d’accueil personnalisé.

Ainsi, chaque projet pourra être consulté en interne ou en externe de la manière suivante :

* Site n°1 (apple)
  + [IP DMZ] : http://192.168.0.1:8080
  + [IP Locale] : http://127.0.0.1 :8080
  + [IP Publique] : http://82.120.150.140:8080
* Site n°2 (banana)
  + [IP DMZ] : http://192.168.0.1:8081
  + [IP Locale] : http://127.0.0.1 :8081

## Journalisation des évènements

Nous avons vu dans le **http.conf** que les logs sont externalisés dans un répertoire dédié. Il est donc nécessaire de créer ce répertoire et les fichiers associés.

|  |
| --- |
| cd /var/log/  mkdir websites  cd websites/  touch apple.log banana.log |

Les logs peuvent être étudiés via une interface graphique, je vous conseille d’utiliser **AWStats**.

## Pare-feu

Pour le pare-feu, nous allons utiliser la commande **iptables** de Linux. Cette commande permet en fait de configurer Netfilter en ligne de commande. Très simplement, Netfilter fonctionne avec des chaînes INPUT, OUTPUT et FORWARD qui vont contenir différentes règles pour réguler le trafic entrant, sortant et traversier.

Le configuration du pare-feu consiste donc à créer un script shell qui sera situé dans **/etc/init.d** et que l’on appellera **firewall.sh**. Pour créer le fichier, tapez la commande suivante :

|  |
| --- |
| gedit /etc/init.d/firewall.sh |

Puis copiez toutes les lignes suivantes :

|  |
| --- |
| #!/bin/sh    **# Vider les tables actuelles**  iptables -t filter -F    **# Vider les règles personnelles**  iptables -t filter -X    **# Interdire toute connexion entrante et sortante**  iptables -t filter -P INPUT DROP  iptables -t filter -P FORWARD DROP  iptables -t filter -P OUTPUT DROP  **# Autoriser le rebouclage** iptables -t filter -A INPUT -i lo -j ACCEPT  iptables -t filter -A OUTPUT -o lo -j ACCEPT    **# Autoriser ICMP (Ping)**  **# Uniquement dans le sens LAN -> DMZ** iptables -t filter -A INPUT -p icmp --icmp-type echo-request –d 192.168.0.0/24 -j ACCEPT  iptables -t filter -A INPUT -p icmp --icmp-type echo-reply –d 192.168.0.0/24 -j ACCEPT  **# Autoriser SSH**  iptables –t filter -A INPUT -p tcp --dport 22 -j ACCEPT    **# Autoriser HTTP** iptables -t filter -A OUTPUT -p tcp --dport 8080,8081,8082 -j ACCEPT   iptables -t filter -A INPUT -p tcp --dport 8080,8081,8082 -j ACCEPT   **# Autoriser HTTPS**  iptables -t filter -A INPUT -p tcp --dport 443 -j ACCEPT  iptables -t filter -A INPUT -p tcp --dport 8443 -j ACCEPT  iptables -t filter -A OUTPUT -p tcp --dport 443 -j ACCEPT |

Une fois le script crée, il faut lui donner les droits d’exécution.

|  |
| --- |
| chmod +x /etc/init.d/firewall.sh |

Enfin, il faut l’ajouter à la liste de démarrage, mais pensez à tester le script avant d’effectuer cette action. Essayez de faire un ping, d’accéder à votre site depuis l’extérieur, etc.. Une fois que c’est fait vous pouvez taper cette commande :

|  |
| --- |
| update-rc.d firewall.sh defaults |

### Remarques

Le choix fait dans ce tutoriel est d’installer le pare-feu sur la machine en DMZ. Il faut donc garder à l’esprit qui si la machine subit une attaque avec élévation de privilèges, le pare-feu sera compris et la machine ne sera plus isolée du LAN.

Une autre solution consiste à utiliser un pare-feu matériel de type IpCop. Vous trouverez des tutoriel à ce sujet sur Internet. Néanmoins cela nécessitera une machine supplémentaire, la sécurité a un coût !

# Conclusion

A ce stade, vous pouvez vérifier que votre site est bien accessible depuis l’extérieur à l’adresse **http://82.120.150.140** (demandez à un ami de se rendre à cette adresse ou utilisez un proxy pour tester) mais également depuis votre LAN en rentrant l’adresse **http://192.168.0.1** dans votre navigateur. Notez que le ping du serveur vers le LAN n’est pas autorisé.