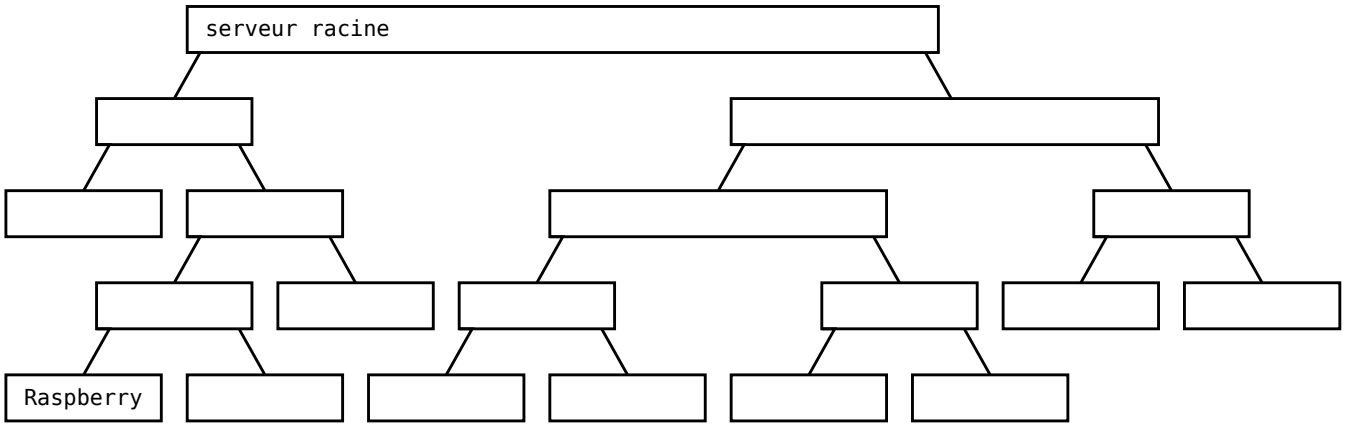
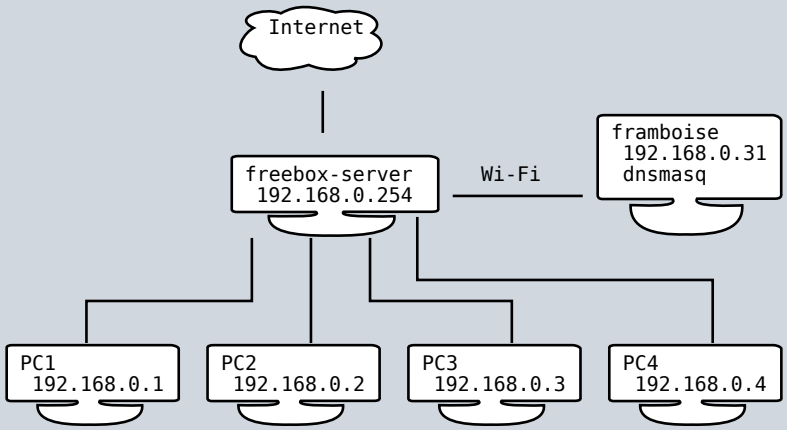
Logiciel

DnsMasq : utiliser votre Raspberry Pi comme serveur DNS (et accélérer Internet)

Voici un schéma des serveurs DNS ; le Raspberry Pi est au bas de l'arbre, le nœud du haut est le serveur racine qui connaît tous les noms de domaine existants :



Voici un exemple de réseau dans lequel un Raspberry Pi est le serveur DNS :



Ce réseau local comporte autour d'une freebox :

- quatre PCs : PC1, PC2, PC3, PC4
- un Raspberry pi sur lequel est installé **dnsmasq** dont on utilise les deux fonctionnalités :
  - DHCP ⇒ attribue aux clients du réseau local :
    - adresses IP
    - noms d'hôtes
    - DNS
    - domaines

- DNS :
  - résout les adresses du domaine local
  - sert de de cache DNS
  - transfère aux DNS externes s'il ne sait pas résoudre en local

Pré-requis

Installation

1. Connectez-vous à votre Raspberry Pi via SSH
2. Mettez à jour votre système :

```
pi@framboise4:~ $ sudo apt update
pi@framboise4:~ $ sudo apt upgrade
```

3. Installez le package **DNSMasq** :

```
pi@framboise4:~ $ sudo apt install dnsmasq
```

DNSMasq est maintenant installé.

Configuration

Voir :

- Comment configurer Dnsmasq, serveur dns et dhcp léger
- Dnsmasq.conf : options

On ne touche pas à :

- **/etc/resolv.conf** qui renvoie vers **127.0.0.1** (dnsmasq installé sur cette machine)
  - ⇒ remplacé par **/etc/resolv.dnsmasq**
- **/etc/dnsmasq.conf** (fichier de configuration principal, entièrement en commentaire)
  - ⇒ remplacé par **/etc/dnsmasq.d/local.conf**

Les fichiers de configuration à utiliser sont :

Fichier à utiliser		au lieu de
/etc/hosts	résolution des noms d'hôte du réseau	
/etc/resolv.dnsmasq	serveurs DNS à utiliser	**/etc/resolv.conf**
/etc/dnsmasq.d/local.conf	fichier de configuration dans lequel une ligne demande d'utiliser **/etc/resolv.dnsmasq**	**/etc/dnsmasq.conf**

Configuration du Raspberry

On ne touche pas au fichier de configuration /etc/dnsmasq.conf.

La configuration sera écrite dans /etc/dnsmasq.d/*.conf

1. Pour une configuration basique, éditez avec les droits d'administration le fichier **/etc/dnsmasq.d/basique.conf** :

   /etc/dnsmasq.d/basique.conf

   ```
   domain-needed
   bogus-priv
   expand-hosts
   domain=me.local
   ```

   - **domain-needed** : n'envoyer des requêtes DNS au serveur DNS principal que pour le nom de domaine
   - **bogus-priv** : pas de requête DNS au serveur DNS principal s'il s'agit d'une adresse IP locale
   - **expand-hosts** : servira à ajouter un faux nom de domaine pour nos appareils locaux
   - **domain=me.local** : un nom de domaine local : Si un hôte se nomme pc, on peut y accéder avec pc.me.local
2. Redémarrez DNSMasq pour appliquer les modifications :

   ```
   pi@framboise4:~ $ sudo systemctl restart dnsmasq.service
   ```

3. Pour créer un hôte local (comme pc.me.local), Éditez avec les droits d'administration le fichier **/etc/hosts** pour ajouter à la fin une ligne pour cet hôte :

   /etc/hosts

   ```
   192.168.1.17    pc
   ```

Cette ligne dit au Raspberry Pi que le nom pc permet d'atteindre l'IP 192.168.1.17 : Le Raspberry Pi peut désormais utiliser pc comme nom d'hôte et tout ordinateur utilisant le Raspberry Pi comme serveur DNS peut utiliser pc.me.local.

Fichier /etc/hosts du serveur DNS

Inutile d'associer freebox-server = 192.168.0.254, cela ne fonctionnera pas, même si une règle dhcp-host est spécifiée dans le fichier de configuration.

La freebox-server est en effet configurée en IP statique sur la freebox (192.168.0.254). Elle ne lance pas de requête DHCP → dnsmasq ne reçoit donc pas de requête DHCP de la part de la freebox.

De même pour le Raspberry Pi qui héberge le serveur dnsmasq et est configuré en IP statique (bail statique dans la box ou fichier /etc/network/interfaces).

Mais en mettant ces adresses dans /etc/hosts, dnsmasq est configuré par défaut pour les lire.

Dans le fichier /etc/hosts, il ne faut pas indiquer **framboise** pour 127.0.0.1 sinon dnsmasq répondra à une requête DNS externe par la réponse framboise → 127.0.0.1, 192.168.0.31. Le PC distant contactera alors 127.0.0.1 (donc lui-même, alors qu'il pensait contacter framboise).

Voici un exemple de contenu du fichier /etc/hosts de framboise :

/etc/hosts

```
127.0.0.1 localhost
192.168.0.31 framboise
192.168.0.31 dns.local
```

Fichier /etc/dnsmasq.d/local.conf (pour ne pas toucher à /etc/dnsmasq.conf)

Voir http://skimpax.web4me.fr/wiki/doku.php/linux/dnsmasq

/etc/dnsmasq.d/local.conf

```
# Ne jamais faire sortir d'adresses locales
#   - Ne jamais transmettre les noms simples (sans point ni
domaine)
```

```
domain-needed
#   - Ne jamais transmettre les adresses dans les espaces
d'adressage non-routés.
bogus-priv

# Fichier contenant la définition des serveurs en amont
#   (au lieu de /etc/resolv.conf)
resolv-file=/etc/resolv.dnsmasq

# Essayer avec chaque serveur dans l'ordre du fichier ci-dessus
strict-order

# Teste les changements du fichier de résolution et le relit
no-poll

# Add other name servers here, with domain specs if they are for
# non-public domains.
#server=/localnet/192.168.0.1

# Example of routing PTR queries to nameservers: this will send
all
# address->name queries for 192.168.3/24 to nameserver 10.1.2.3
#server=/3.168.192.in-addr.arpa/10.1.2.3

# Add local-only domains here, queries in these domains are
answered
# from /etc/hosts or DHCP only.
#local=/localnet/

# Add domains which you want to force to an IP address here.
# The example below send any host in doubleclick.net to a local
# webserver.
#address=/doubleclick.net/127.0.0.1

# nom de l'interface (par exemple eth0)
interface=eth0

# Pour que dnsmasq ne lise pas /etc/hosts
# no-hosts

# pour qu'un nom de domaine soit automatiquement ajouté aux noms
simples du fichier hosts.
expand-hosts
# domaine pour dnsmasq (facultatif ; fait les actions suivantes :
# 1) Permet aux hôtes DHCP d'avoir des noms de domaine complets,
tant que la partie domaine correspond à ce paramètre.
# 3) Fournit le nom de domaine pour "expand-hosts"
domain=maison.lan

# active le serveur DHCP intégré, fournir la plage d'adresses
disponibles et éventuellement une durée de bail
```

```
dhcp-range=192.168.0.100,192.168.0.150,24h

# Baux statiques avec : adresse MAC,nom,adresse IP, durée du bail
#    Baux permanents
dhcp-host=00:24:d4:af:a8:0c,freebox-server,192.168.0.254,infinite
dhcp-host=00:24:d4:7c:59:53,freebox-player,192.168.0.253,infinite
dhcp-host=2C:B0:5D:8B:6C:12,swnetgear,192.168.0.239,infinite
dhcp-host=14:DA:E9:6B:B2:03,omv,192.168.0.250,infinite
dhcp-host=00:50:43:01:51:9A,sheebian,192.168.0.252,infinite
dhcp-host=00:22:F4:42:A3:B1,picuntu,192.168.0.204,infinite
#    Baux à durée limitée
dhcp-host=d0:66:7b:03:fa:66,samsungtv,192.168.0.20,48h
dhcp-host=00:1e:8f:61:a4:0a,imprimante,192.168.0.21,48h
dhcp-host=E0:2A:82:5B:34:F8,domadix,192.168.0.10,48h

# Donner à la machine qui annonce se nommer "SqueezeboxTouch"
l'adresse IP 192.168.0.115 et un bail permanent
dhcp-host=SqueezeboxTouch,192.168.0.115,infinite

# Remplacer la route par défaut fournie par Dnsmasq (qui suppose
que le routeur est sur la même machine que dnsmasq)
#    ici: mettre l'IP de la freebox comme routeur par défaut
dhcp-option=option:router,192.168.0.254

# Set the cachesize here.
cache-size=256

#    ajouter une directive de journalisation, non incluse par
défaut
log-facility=/var/log/dnsmasq.log

#    Pour le débogage, journaliser chaque requête DNS qui passe
par Dnsmasq.
log-queries

#    journaliser beaucoup d'informations supplémentaires sur les
transactions DHCP.
log-dhcp
```

Fichier /etc/resolv.dnsmasq (pour ne pas toucher à /etc/resolv.conf)

Le fichier resolv.conf renvoie vers dnsmasq qui est installé sur cette machine :

Le fichier /etc/resolv.dnsmasq spécifie les serveurs DNS externes à utiliser par dnsmasq :

/etc/resolv.conf

```
nameserver 127.0.0.1
```

Dnsmasq a été configuré pour respecter l'ordre de ce fichier (les préférés en premiers), cf. directive strict-order

Voici un exemple :

/etc/resolv.dnsmasq

```
# Free
nameserver 212.27.40.240
nameserver 212.27.40.241

# OpenDNS
nameserver 208.67.222.222
nameserver 208.67.220.220

# OVH
nameserver 91.121.161.184
nameserver 91.121.164.227
nameserver 188.165.197.144

# Google
nameserver 8.8.8.8
nameserver 4.4.4.4
```

Journalisation

Lorsqu'on active les traces dans dnsmasq, on se rend compte que c'est tout de suite très verbeux, notamment à causes des requêtes DNS. On configure donc les traces avec logrotate, de façon à ne pas saturer de disque.

Configurer dnsmasq

On peut activer indépendamment les traces DNS et DHCP. Un exemple avec les deux activés (commenter log-queries ou log-dhcp pour inhiber) :

```
# Define the log output
log-facility=/var/log/dnsmasq.log

# For debugging purposes, log each DNS query as it passes through
# dnsmasq.
log-queries
```

```
# Log lots of extra information about DHCP transactions.
log-dhcp
```

Les adresses IP statiques

Solution sur framboise :

Dans **/etc/hosts**, indiquer les hôtes ayant une adresse IP statique et configurer dnsmasq pour lire le fichier **/etc/hosts** au démarrage. Pour cela, commenter la règle suivante :

```
#no-hosts
```

Dans le fichier /etc/hosts, il ne faut pas indiquer 'framboise' pour 127.0.0.1 sinon dnsmasq répondra à une requête DNS externe par la réponse framboise → 127.0.0.1, 192.168.0.250. L'hôte distant contactera donc 127.0.0.1 (c'est-à-dire lui-même, alors qu'il pensait contacter framboise)

Contenu de /etc/hosts de framboise

On spécifie les hôtes avec une adresse IP statique, donc non allouée pas dnsmasq.

/etc/hosts

```
127.0.0.1 localhost
192.168.0.250 sheebian sheebian.maison.lan
```

Contenu de /etc/resolv.conf de sheebian

Ce fichier spécifie le serveur DNS à utiliser, donc renvoie vers dnsmasq qui est installé sur cette machine.

/etc/resolv.conf

```
nameserver 127.0.0.1
```

Contenu de /etc/resolv.dnsmasq de framboise

Cf la page officielle de man en français : http://www.linuxcertif.com/man/8/dnsmasq/

recopiée ici : Paramètres de dnsmasq.conf

Les directives de configuration seront écrites dans un fichier **/etc/dnsmasq.d/local.conf** créé pour l'occasion et pris en charge automatiquement [1)]

Ce fichier spécifie en particulier les DNS externes à utiliser par dnsmasq (lignes **nameserver**). Dnsmasq a été configuré pour respecter l'ordre de ce fichier (les préférés en premiers), cf. directive strict-order

/etc/dnsmasq.d/local.conf

```
# Free
nameserver 212.27.40.240
nameserver 212.27.40.241

# OpenDNS
nameserver 208.67.222.222
nameserver 208.67.220.220

# OVH
nameserver 91.121.161.184
nameserver 91.121.164.227
nameserver 188.165.197.144

# Google
nameserver 8.8.8.8
nameserver 4.4.4.4
```

éditez avec les droits d'administration le fichier **/etc/dnsmasq.d/local.conf** pour paramétrer les adresses IP voulues comme ceci :

On commence par empêcher de faire sortir sur internet les requêtes de domaines locaux :

Pour que dnsmasq ajoute automatiquement le nom de domaine quand il sert la demande, ajoutez l'option **expand-hosts** et la définition du nom de domaine (**domain=**).

Avec **expand-hosts**, la recherche DNS pour **hostname.your_domain.com** sauf si **your_domain.com** est spécifié dans l'option **adresse**. Par exemple,

```
domain=your_domain.com
expand-hosts
address=/zirconium.your_domain.com/zr.your_domain.com/192.168.1.31
```

Pour faire des recherches DNS inverses, utiliser **ptr record** :

```
address=/host.example.net/10.1.2.30
ptr-record=30.2.1.10.in-addr.arpa,"host.example.net"
```

Pour créer des baux statiques, utiliser la méthode **dhcp-host** pour les adresses ci-dessus, mais avec des adresses MAC pour ceux qui en ont besoin, par exemple :

```
# This entry is simply a static DNS address, great for mapping print
```

```
servers, etc to names
dhcp-host=zinc,192.168.1.30
# This entry assigns the given IP address to the MAC address for static IP
addresses
# Note that the IP address listed does NOT have to be in the DHCP range
given, just on the same subnet
dhcp-host=11:22:33:44:55:66,zinc,192.168.1.30,infinite
```

```
domain-needed
bogus-priv
```

/etc/dnsmasq.d/local.conf

```
domain-needed
bogus-priv
cache-size=1024

expand-hosts
domain=lan

resolv-file=/etc/resolv.dnsmasq

# Free
nameserver 212.27.40.240
nameserver 212.27.40.241

# OpenDNS
nameserver 208.67.222.222
nameserver 208.67.220.220

# OVH
nameserver 91.121.161.184
nameserver 91.121.164.227
nameserver 188.165.197.144

# Google
nameserver 8.8.8.8
nameserver 4.4.4.4

#    Adresses à forcer
address=/localhost/127.0.0.1
address=/framboise/192.168.0.100
address=/chateau/192.168.0.1
address=/trianon/192.168.0.2
address=/framboise.lan/192.168.0.100
address=/chateau.lan/192.168.0.1
address=/trianon.lan/192.168.0.2
```

domain-needed

Interdit à Dnsmasq de transmettre en amont les requêtes de noms simples (sans point ni nom de domaine).

- Si le nom n'est pas dans **/etc/hosts** ou dans la liste des baux DHCP, dnsmasq répond “non trouvé”.
- Avec **bogus-priv**, cela évite de faire sortir les requêtes de domaines locaux

; bogus-priv

dnsmasq ne transmet pas aux serveurs DNS amont les requêtes DNS inverses pour des adresses IP privées (ie 192.168.x.x, etc…) qui ne sont ni dans **/etc/hosts** ni dans les baux DHCP.

- Il retourne dans ce cas “no such domain”.

; cache-size=<taille>

taille du cache de Dnsmasq

- valeur par défaut : 150 noms.
- une valeur de zéro désactive le cache.

; domain=<domaine>[,<gamme d'adresses>]

domaine du serveur DHCP.

- Ce domaine local sera ajouté aux noms des machines assignées par le DHCP
- Le domaine peut être donné :
  - sans spécifier de gamme d'adresses IP
  - ou pour des gammes d'adresses IP limitées.
- Cela a deux effets :
  1. le serveur DHCP retourne le domaine à tous les hôtes qui le demandent
  2. cela spécifie le domaine valide pour les hôtes DHCP configurés.
- cela empêche un hôte sur le LAN de fournir via DHCP un nom tel que par exemple “microsoft.com” et capturer illégitimement du trafic.
- Si aucun nom de domaine n'est spécifié, les noms d'hôtes avec un nom de domaine (avec un point) seront interdits et enregistrés dans le journal (logs).

- Si un suffixe est fourni,
  - les noms d'hôtes possédant un domaine sont autorisés si le nom de domaine coïncide avec <domaine>
  - les noms d'hôtes ne possédant pas de nom de domaine se voient rajouter le suffixe <domaine>
  - Par exemple, sur mon réseau, je peux configurer **domain=thekelleys.org.uk** avec une machine dont le nom DHCP serait **laptop**.
  - L'adresse IP de cette machine sera disponible à la fois
    - pour **laptop**
    - et **laptop.thekelleys.org.uk**
- Si la valeur fournie pour <domaine> est ”#”, le nom de domaine est positionné à la première valeur de la directive “search” du fichier **/etc/resolv.conf** (ou équivalent).
- La gamme d'adresses est de la forme **<adresse ip>,<adresse ip>** ou **<adresse ip>/<masque de réseau>** voire une simple <adresse ip>.

; expand-hosts

Ajoute le nom de domaine <domaine> défini par **domain=<domaine>** aux noms simples (dont le nom ne contient pas de point) :

- contenus dans le fichier **/etc/hosts**
- et pour le service DHCP

; resolv-file=<fichier>

Lit les adresses des serveurs de nom amont dans le fichier de nom <fichier>, au lieu du fichier **/etc/resolv.conf**.

- Pour le format de ce fichier, voir dans le manuel pour resolv.conf(5) les entrées correspondant aux serveurs de noms (nameserver).
- Dnsmasq peut lire plusieurs fichiers de type resolv.conf, le premier remplace le fichier par défaut, le contenu des suivants est rajouté dans la liste des fichiers à consulter.
- Seul le dernier fichier modifié sera chargé en mémoire.

; server=[/[<domaine>]/[domaine/]][<Adresse IP>[#<port>][@<Adresse IP

source>|<interface>[#<port>]]]

adresse IP d'un serveur de nom amont.

- Cette option n'empêche pas la lecture du fichier **/etc/resolv.conf**.
- Si un ou plusieurs noms de domaine sont fournis,
  - ce serveur ne concernera que ce ou ces domaines : toute requête concernant les domaines <domaine> ne sera adressée qu'à ce serveur.
  - Cette option est destinée aux serveurs de nom privés : si, sur votre réseau, un serveur de nom a pour adresse IP **192.168.1.1** et résout les noms de la forme xxx.internal.thekelleys.org.uk,
    - **server/internal.thekelleys.org.uk/192.168.1.1** enverra les requêtes pour les machines internes vers ce serveur de nom,
    - toutes les autres requêtes seront adressées aux serveurs indiqués dans le fichier /etc/resolv.conf.
  - Si le domaine spécifié est vide (//), ce serveur ne concerne que les noms “non qualifiés”, c'est-à-dire les noms ne possédant pas de point.
- On peut préciser un port non standard à la suite des adresses IP en utilisant le caractère #.
- On peut mettre plus d'une option server en répétant les domaines et adresses IP comme requis.
- Le domaine le plus spécifique l'emporte sur le domaine le moins spécifique, ainsi :

```
server=/google.com/1.2.3.4
server=/www.google.com/2.3.4.5
```

  - enverra les requêtes pour ***.google.com** → **1.2.3.4**,
  - mais ***www.google.com** → **2.3.4.5**.
- L'adresse spéciale **#** signifie “utiliser les serveurs standards”, ainsi

  - ```
    server=/google.com/1.2.3.4
    server=/www.google.com/#
    ```

    enverra les requêtes pour ***.google.com** → **1.2.3.4**

  - mais ***www.google.com** ira comme d'habitude
    ```

aux serveurs définis par défaut.

- On peut aussi donner un nom de domaine mais sans adresse IP. C'est alors un domaine local : dnsmasq doit répondre aux requêtes le concernant à partir des entrées du fichier **/etc/hosts** ou des baux DHCP, et ne jamais transmettre les requêtes aux serveurs amont
- **local** est synonyme de **server** pour clarifier l'utilisation de cette option pour cet usage particulier.
- La chaîne de caractères optionnelle suivant le caractère **@** définit la source que Dnsmasq doit utiliser pour les réponses à ce serveur de nom.
  - Ce doit être une adresses IP appartenant à la machine sur laquelle tourne Dnsmasq ; sinon la ligne sera ignorée et une erreur sera consignée dans le journal des événements.
  - Si un nom d'interface est donné, alors les requêtes vers le serveur de nom seront envoyées depuis cette interface ;
  - si une adresse IP est donnée, alors l'adresse source de la requête sera l'adresse en question.
- L'option **query-port** est ignorée pour tous les serveurs dont l'adresse source est spécifiée, mais il est possible de la donner directement dans la spécification de l'adresse source.

; address=/nom_machine/adresse_ip

dhcp-host=nom_machine,adresse_ip

définit une adresse IP pour la machine nom_machine ; on peut mettre plusieurs lignes.

exemples :

```
address=/zinc/192.168.1.30
address=/zirconium/zr/192.168.1.31
dhcp-host=zinc,192.168.1.30
```

autres exemples :

```
address=/localhost/127.0.0.1 # le localhost de la machine depuis laquelle on consulte le serveur
address=/framboise/192.168.0.100 # domaine framboise et ses sous-domaines *.framboise
address=/chateau/192.168.0.1 # domaine chateau et ses sous-domaines *.chateau
```

```
address=/trianon/192.168.0.2 #: domaine trianon et ses sous-domaines *.trianon
address=/framboise.lan/192.168.0.100 # domaine framboise.lan et ses sous-domaines *.framboise.lan
etc. :
```

</WRAP>

Pour renseigner ce fichier, voir Paramètres de dnsmasq.conf

C'est tout. Redémarrer le service en tapant la commande :

```
$ sudo service dnsmasq restart
```

Désormais, les domaines framboise.lan, etc ainsi que leurs sous-domaines (*.framboise.lan, etc.) existent et permettent l'utilisation des sous-domaines automatiques.

Il ne reste qu'à déclarer les serveurs DNS. Éditez avec les droits d'administration le fichier **/etc/resolv.dnsmasq** pour y écrire l'adresse IP des serveurs DNS comme ceci :

/etc/resolv.dnsmasq

```
nameserver 192.168.0.100
nameserver 192.168.0.254
```

Ne pas oublier de laisser l'adresse de la box (ici, 192.168.0.254)

Un exemple

Créer un petit intranet «maison» :

- avec un nom de domaine qui ne sera fonctionnel que sur le LAN : **mondomaine.lan**
- en IP privées de classe C
- serveur DNS :
  - nom : **ns.mondomaine.lan**
  - adresse IP : **192.168.0.100**
- Adresses du DNS du fournisseur d'accès : 212.27.32.5 212.27.32.5
- machines :

- machine1 :
  - nom : **machine1.mondomaine.lan**
  - adresse IP : **192.168.0.1**
- machine2 :
  - nom : **machine2.mondomaine.lan**
  - adresse IP : **192.168.0.2**
- machineX :
  - nom : **machineX.mondomaine.lan**
  - adresse IP : **192.168.0.X**

Carte du réseau :

Nom d'hôte	**Nom de domaine**	**nom complet de l'hôte**	**Adresse IP**
serveur DNS	mondomaine.lan	ns.mondomaine.lan	192.168.0.100
machine1	mondomaine.lan	machine1.mondomaine.lan	192.168.0.1
machine2	mondomaine.lan	machine2.mondomaine.lan	192.168.0.2
machineX	mondomaine.lan	machineX.mondomaine.lan	192.168.0.X

Configuration des adresses

Déclarer le serveur DNS d'adresse 192.168.0.1 en premier dans le fichier /etc/resolv.conf en mettant sa ligne au début.

Les adresses de DNS du fournisseur d'accès sont spécifiées dans le fichier /etc/named.conf par l'instruction forwarders.

Fixer l'adresse IP du serveur

éditez avec les droits d'administration le fichier **/etc/network/interfaces** pour le modifier comme ceci :

/etc/network/interfaces

```
# This file describes the network
interfaces available on your system
# and how to activate them. For
more information, see
interfaces(5).
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth0
iface eth0 inet static
address 192.168.1.254
netmask 255.255.255.0
network 192.168.1.0
```

```
broadcast 192.168.1.255
gateway 192.168.1.1
# dns-* options are implemented by
the resolvconf package, if
installed
dns-nameservers 192.168.1.1
```

Configuration des noms d'hôte

Fichier /etc/hostname

éditez avec les droits d'administration le fichier **/etc/hostname** pour renseigner le nom du serveur DNS comme ceci :

/etc/hostname

```
dns.mondomaine.lan
```

Démarrer le service :

- `/etc/init.d/hostname.sh start`

Fichier /etc/hosts

Éditez avec les droits d'administration le fichier **/etc/hosts** pour y inscrire le nom du serveur et son adresse IP :

/etc/hosts

```
127.0.0.1     localhost.localdomain
localhost
192.168.0.200 dns.mondomaine.lan
dns
```

Fichier /etc/host.conf (vérification)

Éditez avec les droits d'administration le fichier **/etc/host.conf** pour y insérer les paramètres suivants (normalement c'est déjà fait par l'installation de bind9) :

/etc/host.conf

```
order hosts, bind
multi on
```

Fichier /etc/resolv.conf

Éditez avec les droits d'administration le fichier **/etc/resolv.conf** pour y entrer le domaine, la zone de recherche et le nom du serveur DNS :

/etc/resolv.conf

```
domain mondomaine.lan
search mondomaine.lan
nameserver 192.168.0.200
```

Configuration de l'ordinateur

La dernière étape consiste à configurer votre ordinateur pour utiliser le Raspberry Pi comme serveur DNS. Pour ce faire, vous avez deux options :

1. **Configuration manuelle** : selon votre système d'exploitation, allez dans vos paramètres réseau et définissez le serveur DNS principal avec l'adresse IP du Raspberry Pi.
    - **Sous Windows**, c'est dans le **Panneau de configuration > Réseau et Internet > Centre Réseau et partage > Modifier les paramètres de l'adaptateur**.
      Faites un clic droit sur l'interface réseau que vous utilisez et allez dans Propriétés.
      Double-cliquez sur IPV4 et définissez le serveur DNS préféré avec l'adresse IP du Raspberry Pi
    - **Sous Ubuntu / Debian**, vous pouvez le modifier directement dans le fichier /etc/resolv.conf ou dans le gestionnaire de réseau si vous avez un bureau graphique
2. **Configuration DHCP** (sans doute le meilleur choix) :
   L'autre option est de changer le serveur DNS de votre serveur DHCP.
   Dans la configuration DHCP, vous pouvez choisir la plage IP et le serveur DNS envoyés aux clients.
   Cela fonctionnera directement avec tous vos appareils, y compris les smartphones.
   Si vous choisissez cette option, n'oubliez pas de définir manuellement un serveur DNS externe sur votre Raspberry Pi.

Utilisation

Tests

1. Vérifiez que le serveur DNS fonctionne correctement :

```
...@...:~$ nslookup
> server 192.168.0.32
Default server: 192.168.0.32
Address: 192.168.0.32#53
> google.fr
Server:         192.168.0.32
Address:   192.168.0.32#53

Non-authoritative answer:
Name:  google.fr
Address: 216.58.209.227
Name:  google.fr
Address: 2a00:1450:4007:807::2003
> exit
```

   - server 192.168.0.32 : adresse du Raspberry
   - google.fr : Il montre une adresse IP correspondant au nom de domaine → Votre serveur DNS fonctionne.
   - Vous pouvez également essayer de résoudre l'hôte local (ex: kodi.local.me) pour vérifier que cela fonctionne aussi

Vérification

Pour vérifier, lancer depuis un PC du réseau :

```
$ dig framboise.lan
```

Il faut installer le paquet **dnsutils** ou en console :

```
$ sudo apt install dnsutils
```

Vérifier si la syntaxe des fichiers de configuration est correcte avec les commandes suivantes :

Pour tester la résolution de noms et la recherche inverse,

```
$ host nom_du_serveur
$ host @IP_du_serveur
```

Par exemple

```
$ host -a mondomaine.lan
$ host test1.mondomaine.lan
$ host test2.mondomaine.lan
$ host test3.mondomaine.lan
```

La commande « nslookup » (Name System Look Up) permet d'interroger un serveur de noms pour obtenir les informations concernant un domaine ou un hôte et diagnostiquer les problèmes de configuration du DNS :

```
$ nslookup nom_du_serveur
$ nslookup @IP_du_serveur
```

La commande « dig » permet sensiblement la même chose que « nslookup » :

```
$ dig nom_du_serveur
$ dig nom_de_domaine
$ dig -x @IP_du_serveur
```

Quelques autres commandes utiles

- classique :

  ```
  $ ping nom_du_serveur
  ```

  ou

  ```
  $ ping adresse_IP_du_serveur
  ```

- ```
  $ dhclient eth0
  ```

  pour redonner une @ IP au client

Pour aller plus loin

Voici quelques définitions utiles à la compréhension et au paramétrage du DNS :

serveur maître

> s'occupe des enregistrements DNS d'un domaine enregistré ou non (par exemple pour le cas d'un réseau local fermé comme celui de notre tuto maison.lan)
  ```

; une zone

: ensemble des enregistrements DNS pour un nom de domaine. Il existe une zone de recherche directe (correspondance entre le nom d'hôte et l'adresse IP) et une zone de recherche inverse (correspondance entre l'adresse IP et le nom d'hôte)

; serveur secondaire

: assure la redondance du serveur maître et prend le relais de celui-ci en cas de panne

; serveur esclave

: une réplication du serveur maître utilisé comme sauvegarde

; serveur cache

: stocke les requêtes déjà résolues ce qui permet d'économiser la bande passante et de réduire le temps de latence

Maintenant vous disposez d'un serveur DNS digne de ce nom (sans faire de jeu de mots). Le DNS est important surtout si vous souhaitez installer d'autres services comme la messagerie par exemple.

Cohabitation avec systemd

Voir :

1. https://www.osradar.com/how-to-install-configure-dnsmasq-on-ubuntu-18-04-lts/
2. https://unix.stackexchange.com/questions/304050/how-to-avoid-conflicts-between-dnsmasq-and-systemd-resolved
3. https://bbs.archlinux.org/viewtopic.php?id=246649
4. https://askubuntu.com/questions/1012641/dns-set-to-systemds-127-0-0-53-how-to-change-permanently
5. https://askubuntu.com/questions/898605/how-to-disable-systemd-resolved-and-resolve-dns-with-dnsmasq

Problèmes rencontrés

- Comment éviter les conflits entre dnsmasq et systemd-

resolved ?
- DNSMasq : Utilisation du plug-in DNSMasq de NetworkManager

Questions et Réponses

- Cohabitation avec systemd : depuis Ubuntu 18.04, dnsmasq ne fait plus de résolution DNS

=?= depuis Ubuntu 18.04 ===

Cohabitation avec systemd : depuis Ubuntu 18.04, dnsmasq ne fait plus de résolution DNS

Solution de contournement : désactiver le resolver natif **systemd-resolved.service** et utiliser **dnsmasq**.

Création d'un fichier pour **dnsmasq** pour y renseigner ses serveurs DNS et les autres commandes de **dnsmasq**.

Créez avec les droits d'administration le fichier **/etc/NetworkManager/dnsmasq.d/monresolv.conf** pour y écrire :

/etc/NetworkManager/dnsmasq.d/monresolv.conf

```
cache-size=1000
except-interface=lo
server=/localhost/127.0.0.1
server=/nom.domaine.tld/adresse_ip_serveur_dns_1
server=/nom.domaine.tld/adresse_ip_serveur_dns_2

# adresses :
address=/domaine1.tld/adresse_ip_1 # domaine domaine1.tld et ses sous-domaines
*.domaine1.tld
```

Stoppez et désactivez le service systemd-resolved :

```
$ sudo systemctl disable systemd-resolved.service
...
```

```
Removed /etc/systemd/system/multi-
user.target.wants/systemd-
resolved.service.
Removed /etc/systemd/system/dbus-
org.freedesktop.resolve1.service.
$ sudo systemctl stop systemd-
resolved.service
```

Supprimez le fichier resolv.conf :

```
$ sudo rm /etc/resolv.conf
```

Éditez avec les droits d'administration le fichier **/etc/NetworkManager/NetworkManager.conf** et ajoutez la méthode dns=dnsmasq :

/etc/NetworkManager/NetworkManager.conf

```
[main]
plugins=ifupdown,keyfile
dns=dnsmasq
```

Redémarrez le service NetworkManager :

```
$ sudo systemctl restart
NetworkManager
```

Vérifiez que le service dnsmasq est bien lancé :

```
$ sudo ps -ef | grep dnsmasq
...
xxxxxxx  13654  4831  0 09:57 pts/0
00:00:00 grep --color=auto dnsmasq
```

Vérifiez le fichier resolv.conf :

```
$ cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 127.0.1.1
```

source : https://it.izero.fr/linux-remplacer-resolver-dns-systemd-resolved-par-dnsmasq/

Désinstallation

Pour supprimer cette application, il suffit de supprimer son paquet.

Selon la méthode choisie, la configuration globale de l'application est conservée ou supprimée.

Les journaux du système, et les fichiers de préférence des utilisateurs dans leurs dossiers personnels sont toujours conservés.

Voir aussi

- **(fr)** Comment utiliser votre Raspberry Pi comme serveur DNS local ?
- **(en)** https://unix.stackexchange.com/questions/304050/how-to-avoid-conflicts-between-dnsmasq-and-systemd-resolved
- **(fr)** http://irp.nain-t.net/doku.php/160dns:30_construire_un_dns et le reste du site
- **(fr)** http://www.ced-info.com/administration-reseaux/bind9-installation-et-parametrage
- **(fr)** http://linux.crdp.ac-caen.fr/Lcs3/x870.html
- **(fr)** http://webadonf.net/2011/03/configurer-un-serveur-dns-avec-bind9-sur-debian-squeeze/

Basé sur « Comment utiliser votre Raspberry Pi comme serveur DNS local ? » par raspberrytips.fr.

1)

Par défaut, le fichier de configuration de dnsmasq est **/etc/dnsmasq.conf**, dont toutes les lignes sont commentées. En n'y touchant pas, on facilite les mises à jour.