Virtual Services Howto

Brian Ackerman, brian@nycrc.net
Adaptation française Etienne BERNARD bernard@isia.cma.fr v1.2, 4 Novembre 1997

Ce document a été écrit pour répondre au nombre grandissant de questions sur la manière de rendre un service virtuel.

Table des matières

1 Introduction **2**

1.1 Pré-requis . . . 2
1.2 But . . . 3
1.3 Commentaires . . . 3
1.4 Histoire des changements . . . 3
1.5 Copyright/Distribution . . . 3

2 IP aliasing **4**

3 Virtuald **4**

3.1 Comment ça marche . . . 4
3.2 inetd . . . 4
3.3 virtual.conf . . . 5
3.4 Le code source de virtuald . . . 5

4 Scripts utiles **8**

4.1 virtfs . . . 8
4.2 virtexec . . . 12
4.3 Notes à propos de virtfs et virtexec . . . 15

5 DNS **15**

6 Syslog **15**

6.1 Problème . . . 15
6.2 Solution . . . 16

7 FTP virtuel **17**

8 Web virtuel **18**

9 Mail virtuel avec Pop **18**

9.1 Avis aux utilisateurs de QMail 18

9.2 Problème 18

9.3 Une mauvaise solution 19

9.4 Une bonne solution 19

10 Le reste **21**

11 Conclusion **22**

12 FAQ **22**

1 Introduction

1.1 Pré-requis

Créer une machine pour des services virtuels n'est pas du tout difficile. Cependant, des connaissances basiques ne sont pas suffisantes. De plus, ce document n'est pas destiné à vous expliquer comment configurer une machine sous Linux.

Afin que vous puissiez comprendre cet HOWTO, nous supposerons que les documents suivants vous sont tout à fait familiers:

- Compiler un noyau Linux et ajouter le support de l'IP aliasing IP alias mini-HOWTO
- Installer et configurer des périphériques réseaux NET-3 HOWTO
- Installer inetd NET-3 HOWTO
- Compiler et installer divers paquetages de logiciels en réseau comme Le site de Sendmail Le site d'Apache La FAQ de Wu-Ftpd
- Mettre en place le DNS DNS HOWTO

Si vous n'êtes pas certain de la marche à suivre pour effectuer une des actions précédents, il est FORTEMENT recommandé que vous suiviez les liens proposés pour vous familiariser avec tous ces paquetages. Je ne répondrai à AUCUN mail concernant les points précédents. Veuillez s'il vous plaît rediriger toute question à l'auteur du HOWTO approprié.

1.2 But

Le but des services virtuels est de permettre à une seule machine de reconnaître de multiples adresses IP sans avoir de multiples cartes réseau. L'IP aliasing est une option du noyau qui vous permet d'assigner plus d'une adresse IP à chaque périphérique réseau. Le noyau multiplexe (les échange très rapidement) en tâche de fond et l'utilisateur a l'impression d'avoir plusieurs cartes réseau dans sa machine.

Ce multiplexage permet à de multiples domaines (www.domaine1.com, www.domaine2.com, etc.) d'être logés sur la même machine pour le même coût que pour un seul domaine. Malheureusement, la plupart des services (ftp, web, courrier électronique) n'ont pas été conçus pour gérer de multiples domaines. Afin de les faire fonctionner correctement, vous devrez modifier à la fois les fichiers de configuration et le code source. Ce document décrit comment réaliser ces modifications pour la mise en place d'une machine virtuelle.

Un démon est également nécessaire afin de faire fonctionner les services virtuels. Le code source de ce démon (virtuald) est fourni plus loin dans ce document.

1.3 Commentaires

Ce document va grossir au fur et à mesure que les paquetages seront mis à jour et que le code source ou que les modifications proposées changeront. Si quelque partie de ce document n'est pas claire, envoyez moi vous questions ou suggestions. Afin que je n'ai pas à chercher dans le HOWTO en entier, assurez vous s'il vous plaît que les commentaires sont aussi spécifiques que possible et incluent la section où se trouve le point discutable. Il est important que tout mail ait un champ Subject: contenant VIRTSERVICES HOWTO. Tout autre mail sera considéré comme du mail personnel, et tous mes amis savent que je ne lis pas tout le temps mon mail personnel et il risque donc d'être effacé avec le leur.

Veuillez également noter que mes exemples ne sont pas autre chose que des exemples, et ne doivent pas être recopiés tels quels. Il se peut que vous ayez à insérer vos propres valeurs. Si vous rencontrez des problèmes, envoyez moi un mail, contenant tous les fichiers de configuration pertinents et les messages d'erreur que vous avez obtenu lors de l'installation. Je vous renverrai mes suggestion.

1.4 Histoire des changements

V1.0 Document initial.

V1.1 Correction d'une erreur dans la section sur le web virtuel.

V1.2 Correction de la date.

1.5 Copyright/Distribution

Ce document est Copyright (c) 1997 par The Computer Resource Center Inc.

Une copie de ce document peut être reproduite ou distribuée sur n'importe quel support physique ou électronique sans la permission de l'auteur. De la même façon, les traductions sont autorisées sans permission

expresse si elle incluent un mot disant qui l'a traduit (NdT: voir le début du document pour mes coordonnées). Une redistribution commerciale est autorisé et encouragée. Dans ce cas cependant, faites en part à Computer Resource Center.

Vous pouvez utiliser des extraits de ce document sans accord de l'auteur, à condition que l'oeuvre dérivée contienne une copie de ce document ou un pointeur vers une copie de ce document.

Vous avez la permission d'effectuer des copies ainsi que de les distribuer à condition que le paragraphe sur le copyright ainsi que cette note soient préservés sur toutes les copies.

En bref, nous désirons promouvoir la dissémination de cette information par quelque moyen que ce soit. Cependant, je désire conserver le copyright sur ce document, et aimerait être tenu au courant de tous les plans de redistributions de ce HOWTO.

2 IP aliasing

L'IP aliasing est une option du noyau qui doit être mise en place afin de pouvoir faire tourner des services virtuels sur une machine. Il existe déjà un mini-HOWTO expliquant l' IP aliasing. Référez vous à ce document pour toute question concernant la mise en place de cette option.

3 Virtuald

3.1 Comment ça marche

Toute connexion réseau est composée de deux paires adresse IP/port. L'API (Applications Program Interface, ou Interface de Programmation d'Applications) pour la programmation réseau est nommée l'API Sockets. La socket agit comme un fichier ouvert, et vous pouvez envoyer ou recevoir des données à travers une connexion réseau en lisant ou en écrivant dans la socket. Il existe une fonction, `getsockname`, qui retourne l'adresse IP de la socket locale. Virtuald utilise `getsockname` pour déterminer sur quel adresse IP de la machine locale la connexion a été faite. Virtuald lit un fichier de configuration pour récupérer le répertoire associé à cette adresse IP. Il va utiliser `chroot` sur ce répertoire et prendre en compte la connexion au service. `Chroot` change le répertoire / (le répertoire root) vers un nouveau point, de sorte que tout ce qui est au dessus de ce répertoire devient inaccessible pour le programme. Ainsi, chaque adresse IP se voit assigné un système vi!!! rtuel de fichiers. Pour le programme réseau, ceci est transparent, et le programme va se comporter comme si de rien n'était. Virtuald, en conjonction avec un programme comme inetd, peut être utilisé pour virtualiser n'importe quel service.

3.2 inetd

Inetd est un super serveur réseau qui écoute sur de multiples ports et, lorsqu'il reçoit une demande de connexion (par exemple, une requête POP), inetd réalise la connexion et la passe au programme spécifié. Cela évite de faire tourner des serveurs pour rien lorsqu'il n'y a aucune demande pour eux.

Un fichier `/etc/inetd.conf` standard ressemble à ceci :

```
ftp stream tcp nowait root /usr/sbin/tcpd wu.ftpd -l -a
pop-3 stream tcp nowait root /usr/sbin/tcpd in.qpop -s
```

Un `/etc/inetd.conf` virtuel ressemble à cela :

```
ftp stream tcp nowait root /usr/bin/virtuald virtuald /virtual/conf.ftp wu.ftpd -l -a
pop-3 stream tcp nowait root /usr/bin/virtuald virtuald /virtual/conf.pop in.qpop -s
```

3.3 virtual.conf

Chaque service se voit attribué un fichier de configuration qui contrôlera quelles IPs et quels répertoires sont autorisés pour ce service. Vous pouvez avoir un fichier de configuration principal ou de nombreux fichiers de configuration si vous désirez que chaque service se voit attribuer une liste de domaines différente. Un fichier `virtual.conf` ressemble à ceci :

```
# C'est un commentaire, comme le sont les lignes blanches

# Format IP <SPACE> dir <NOSPACES>
10.10.10.129 /virtual/foo.bar.com
10.10.10.130 /virtual/bar.foo.com
10.10.10.157 /virtual/boo.la.com
```

3.4 Le code source de virtuald

```
#include <netinet/in.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <stdarg.h>
#include <string.h>
#include <syslog.h>
#include <stdio.h>

#define BUFSIZE 8192

main(int argc,char **argv)
{
        char buffer[BUFSIZE];
        char *ipaddr,*dir;

        logit("Virtuald Starting: $Revision: 1.19 $");
        if (!argv[1])
```

```
        {
                logit("invalid arguments: no conf file");
                quitting_virtuald(0);
        }
        if (!argv[2])
        {
                logit("invalid arguments: no program to run");
                quitting_virtuald(0);
        }
        if (getipaddr(\&ipaddr))
        {
                logit("getipaddr failed");
                quitting_virtuald(0);
        }
        sprintf(buffer,"Incoming ip: %s",ipaddr);
        logit(buffer);
        if (iptodir(\&dir,ipaddr,argv[1]))
        {
                logit("iptodir failed");
                quitting_virtuald(0);
        }
        if (chroot(dir)<0)
        {
                logit("chroot failed: %m");
                quitting_virtuald(0);
        }
        sprintf(buffer,"Chroot dir: %s",dir);
        logit(buffer);
        if (chdir("/")<0)
        {
                logit("chdir failed: %m");
                quitting_virtuald(0);
        }
        if (execvp(argv[2],argv+2)<0)
        {
                logit("execvp failed: %m");
                quitting_virtuald(0);
        }
}

int logit(char *buf)
{
        openlog("virtuald",LOG_PID,LOG_DAEMON);
        syslog(LOG_ERR,buf);
```

```
        closelog();
        return 0;
}

int quitting_virtuald(int retval)
{
        exit(retval);
        return 0;
}

int getipaddr(char **ipaddr)
{
        struct sockaddr_in virtual_addr;
        static char ipaddrbuf[BUFSIZE];
        int virtual_len;
        char *ipptr;

        virtual_len=sizeof(virtual_addr);
        if (getsockname(0,(struct sockaddr *)\&virtual_addr,\&virtual_len)<0)
        {
                logit("getipaddr: getsockname failed: %m");
                return -1;
        }
        if (!(ipptr=inet_ntoa(virtual_addr.sin_addr)))
        {
                logit("getipaddr: inet_ntoa failed: %m");
                return -1;
        }
        strncpy(ipaddrbuf,ipptr,sizeof(ipaddrbuf)-1);
        *ipaddr=ipaddrbuf;
        return 0;
}

int iptodir(char **dir,char *ipaddr,char *filename)
{
        char buffer[BUFSIZE],*bufptr;
        static char dirbuf[BUFSIZE];
        FILE *fp;

        if (!(fp=fopen(filename,"r")))
        {
                logit("iptodir: fopen failed: %m");
                return -1;
        }
```

```
        *dir=NULL;
        while(fgets(buffer,BUFSIZE,fp))
        {
                buffer[strlen(buffer)-1]=0;
                if (*buffer=='#' || *buffer==0)
                        continue;
                if (!(bufptr=strchr(buffer,' ')))
                {
                        logit("iptodir: strchr failed");
                        return -1;
                }
                *bufptr++=0;
                if (!strcmp(buffer,ipaddr))
                {
                        strncpy(dirbuf,bufptr,sizeof(dirbuf)-1);
                        *dir=dirbuf;
                        break;
                }
        }
        if (fclose(fp)==EOF)
        {
                logit("iptodir: fclose failed: %m");
                return -1;
        }
        if (!*dir)
        {
                logit("iptodir: ip not found in conf file");
                return -1;
        }
        return 0;
}
```

4 Scripts utiles

4.1 virtfs

Chaque domaine doit avoir une arborescence de répertoires. Puisque vous utilisez `chroot`, vous aurez besoin de copies multiples des librairies, binaires, fichiers de configuration, etc... partagés. J'utilise le répertoire `/virtual/domaine.com` pour chaque domaine que je crée. Je comprends bien que cela représente du gaspillage d'espace disque, mais l'espace disque est meilleur marché qu'une nouvelle machine ou que des cartes réseau. Si vous désirez réellement sauver de l'espace disque, vous pouvez lier les fichiers entre eux, afin qu'une seule copie de chaque binaire soit présente.

Voici un fichier virtfs d'exemple :

```
#!/bin/bash

echo '$Revision: 1.19 $'

echo -n "Saisissez le nom de domaine : "
read domain

if [ "$domain" = "" ]
then
        echo Vous n'avez rien saisi : on arrete la
        exit 0
fi

leadingdir=/virtual

echo -n "Saisissez le nom du repertoire contenant les domaines (defaut: $default): "
read ans

if [ "$ans" != "" ]
then
        leadingdir=$ans
fi

newdir=$leadingdir/$domain

if [ -d "$newdir" ]
then
        echo Le repertoire $newdir existe deja
        exit 0
else
        echo Nouveau repertoire : $newdir
fi

echo Creation de $newdir
mkdir -p $newdir

echo Creation de bin
cp -pdR /bin $newdir

echo Creation de dev
cp -pdR /dev $newdir
```

```
echo Creation de dev/log
ln -f /virtual/log $newdir/dev/log

echo Creation d'etc
mkdir -p $newdir/etc
for i in /etc/*
do
        if [ -d "$i" ]
        then
                continue
        fi
        cp -pd $i $newdir/etc
done

echo Creation de etc/skel
mkdir -p $newdir/etc/skel

echo Creation de home
for i in a b c d e f g h i j k l m n o p q r s t u v w x y z
do
        mkdir -p $newdir/home/$i
done

echo Creation de home/c/crc
mkdir -p $newdir/home/c/crc
chown crc.users $newdir/home/c/crc

echo Creation de lib
mkdir -p $newdir/lib
for i in /lib/*
do
        if [ -d "$i" ]
        then
                continue
        fi
        cp -pd $i $newdir/lib
done

echo Creation de proc
mkdir -p $newdir/proc

echo Creation de sbin
cp -pdR /sbin $newdir
```

```
echo Creation de tmp
mkdir -p -m 0777 $newdir/tmp
chmod +t $newdir/tmp

echo Creation de usr
mkdir -p $newdir/usr

echo Creation de usr/bin
cp -pdR /usr/bin $newdir/usr

echo Creation de usr/lib
mkdir -p $newdir/usr/lib

echo Creation de usr/lib/locale
cp -pdR /usr/lib/locale $newdir/usr/lib

echo Creation de usr/lib/terminfo
cp -pdR /usr/lib/terminfo $newdir/usr/lib

echo Creation de usr/lib/zoneinfo
cp -pdR /usr/lib/zoneinfo $newdir/usr/lib

echo Creation de usr/lib/\*.so\*
cp -pdR /usr/lib/*.so* $newdir/usr/lib

echo Creation de usr/sbin
cp -pdR /usr/sbin $newdir/usr

echo Lien de usr/tmp vers /tmp
ln -s /tmp $newdir/usr/tmp

echo Creation de var
mkdir -p $newdir/var

echo Creation de var/lock
cp -pdR /var/lock $newdir/var

echo Creation de var/log
mkdir -p $newdir/var/log

echo Creation de var/log/wtmp
cp /dev/null $newdir/var/log/wtmp

echo Creation de var/run
```

```
cp -pdR /var/run $newdir/var

echo Creation de var/run/utmp
cp /dev/null $newdir/var/run/utmp

echo Creation de var/spool
cp -pdR /var/spool $newdir/var

echo Lien de var/tmp vers /tmp
ln -s /tmp $newdir/var/tmp

echo Creation de var/www/html
mkdir -p $newdir/var/www/html
chown webmast.www $newdir/var/www/html
chmod g+s $newdir/var/www/html

echo Creation de var/www/master
mkdir -p $newdir/var/www/master
chown webmast.www $newdir/var/www/master

echo Creation de var/www/server
mkdir -p $newdir/var/www/server
chown webmast.www $newdir/var/www/server

exit 0
```

4.2 virtexec

Afin d'exécuter des commandes dans un environnement virtuel, vous devez utilise `chroot` sur ce répertoire puis lancer la commande. J'ai écris un script shell nommé virtexec se chargeant de ces opérations, pour n'importe quelle commande :

```
#!/bin/sh

echo '$Revision: 1.19 $'

BNAME=‘basename $0‘
FIRST4CHAR=‘echo $BNAME | cut -c1-4‘
REALBNAME=‘echo $BNAME | cut -c5-‘

if [ "$BNAME" = "virtexec" ]
then
        echo Vous ne pouvez pas lancer virtexec directement. Il FAUT un lien symbolique
        exit 0
```

```
fi

if [ "$FIRST4CHAR" != "virt" ]
then
        echo Le lien ne pointe pas sur une fonction virtuelle
        exit 0
fi

list=""
num=1
for i in /virtual/*
do
        if [ ! -d "$i" ]
        then
                continue
        fi
        if [ "$i" = "/virtual/lost+found" ]
        then
                continue
        fi
        list="$list $i $num"
        num=‘expr $num + 1‘
done

if [ "$list" = "" ]
then
        echo Je ne trouve pas d’environnement virtuel
        exit 0
fi

dialog --clear --title ’Virtexec’ --menu Pick 20 70 12 $list 2> /tmp/menu.$$
if [ "$?" = "0" ]
then
        newdir=‘cat /tmp/menu.$$‘
else
        newdir=""
fi
tput clear
rm -f /tmp/menu.$$

echo ’$Revision: 1.19 $’

if [ ! -d "$newdir" ]
then
```

```
        echo Le nouveau repertoire $newdir N'EXISTE PAS
        exit 0
else
        echo Nouveau repertoire : $newdir
fi

echo bname: $BNAME

echo realbname: $REALBNAME

if [ "$*" = "" ]
then
        echo argumentss: aucun
else
        echo args: $*
fi

echo Changement de repertoire vers $newdir
cd $newdir

echo Lancement de $REALBNAME

chroot $newdir $REALBNAME $*

exit 0
```

Veuillez noter que vous devez disposer du programme `dialog` sur votre système pour que ce script fonctionne. Pour utiliser virtexec, créez un lien symbolique d'un programme vers celui-ci. Par exemple :

```
ln -s /usr/bin/virtexec /usr/bin/virtpasswd
ln -s /usr/bin/virtexec /usr/bin/virtvi
ln -s /usr/bin/virtexec /usr/bin/virtpico
ln -s /usr/bin/virtexec /usr/bin/virtemacs
ln -s /usr/bin/virtexec /usr/bin/virtmailq
```

A présent, si vous tapez `virtvi` ou `virtpasswd` ou encore `virtmailq`, cela vous permettra d'éditer un fichier, changer le mot de passe d'un utilisateur, ou vérifier la queue de mail sur votre système virtuel. Vous pouvez créer autant de liens vers virtexec que vous désirez. Cependant, notez bien que si votre programme nécessite une librairie partagée, celle-ci doit se trouver sur le système de fichiers virtuel. Le binaire doit exister sur le système de fichiers virtuel également.

4.3 Notes à propos de virtfs et virtexec

J'installe tous les scripts dans `/usr/bin`. Tout ce que je ne désire pas mettre sur un système de fichier virtuel, je le place dans `/usr/local`. Le script ne touche à rien dans ce répertoire lors de la copie. Les fichiers ne devant pas chevaucher plusieurs systèmes de fichiers virtuels doivent être supprimés. Par exemple, `ssh` est installé sur mon système, et je n'ai pas voulu que les clefs privées soient disponibles sur tous les systèmes de fichier. J'ai donc supprimé le fichier des systèmes de fichiers virtuels après avoir lancé `virtfs`. Je change également le `resolv.conf` et supprime tout ce qui contient le nom d'un autre domaine, pour des raisons légales. Par exemple, les fichiers `/etc/hosts` et `/etc/HOSTNAME`.

Les programmes pour lesquels je fais un lien symbolique vers virtexec sont :

- virtpasswd – changer le mot de passe d'un utilisateur
- virtadduser – ajouter un utilisateur
- virtdeluser – supprimer un utilisateur
- virtsmbstatus – consulter l'état de samba
- virtvi – éditer un fichier
- virtmailq – vérifier la mailq
- virtnewaliases – reconstruire la table des alias mail

5 DNS

Vous pouvez configurer le DNS normalement. Ce qui est bien avec ce système, c'est que tous les services se comportement normalement, comme s'ils se trouvaient sur des machines physiques différentes. Vous pouvez consulter le HOWTO sur le DNS.

6 Syslog

6.1 Problème

Syslog est l'outil de *logging* couramment utilisé sur les système UNIX. Syslog est un démon qui ouvre un fichier spécial appelé FIFO. Une FIFO est un fichier spécial, se comportant comme une file d'attente. Tout ce qui y est écrit "ressortira" en lecture. Le démon syslog attend les données en lecture. Il existe des fonctions C pour écrire dans les FIFO. Si vous utilisez ces fonctions C dans vos programmes. Souvenez vous que vous avez utilisé `chroot` et que la FIFO `/dev/log` ne se trouve pas dans l'environnement virtuel. Cela implique que aucun des environnements virtuels ne pourra utiliser `syslog`. Nous ne pouvons pas tout simplement copier le fichier, puisque les programmes utilisent `/dev/log` au lieu du nouveau que nous aurions créé.

Attention, certaines version de `syslog` utilisent une socket udp au lieu d'une FIFO. Cependant, ce n'est généralement pas le cas.

6.2 Solution

Syslog peut scruter d'autres FIFO si vous lui dites en ligne de commande. Lancer donc syslog avec l'argument :

```
syslog -p /virtual/log
```

Faites alors un lien de `/dev/log` vers `/virtual/log` (un lien symbolique) :

```
ln -sf /virtual/log /dev/log
```

Puis liez toutes les copies de `/dev/log` vers ce fichier avec la commande (attention, c'est un lien NON symbolique) :

```
ln /virtual/log /virtual/domain.com/dev/log
```

Le script virtfs ci-dessus le fait pour vous. Puisque `/virtual` est un disque entier, et que les `/dev/log` sont liés, ils ont le même numéro d'inode et pointent vers les mêmes données. Le `chroot` ne peut pas empêcher cela, et donc tous vos `/dev/log` virtuels vont à présent fonctionner. Notez également que tous les messages de toutes les machines virtuelles seront écrits dans un même fichier. Cependant, vous pouvez écrire des programmes pour filtrer les données. Si vous ne désirez pas écrire un programme et que vous avez néanmoins besoin de fichiers de log séparé, vous pouvez lancer des programmes syslog différents pour chaque système de fichiers virtuel avec la commande :

```
syslog -p /virtual/domain1.com/dev/log
syslog -p /virtual/domain2.com/dev/log
```

Cependant, cela gaspille des processus et je ne le recommande pas. Cette version du fichier `syslog.init` refait les liens vers les `/dev/log` à chaque fois que vous le lancez, au cas ou la configuration ait été endommagée. Voici un `syslog.init` modifié :

```
#!/bin/sh

# Source function library.
. /etc/rc.d/init.d/functions

case "$1" in
  start)
        echo -n "Starting dev log: "
        ln -sf /virtual/log /dev/log
        echo done
        echo -n "Starting system loggers: "
        daemon syslogd -p /virtual/log
        daemon klogd
        echo
```

```
        echo -n "Starting virtual dev log: "
        for i in /virtual/*
        do
                if [ ! -d "$i" ]
                then
                        continue
                fi
                if [ "$i" = "/virtual/lost+found" ]
                then
                        continue
                fi
                ln -f /virtual/log $i/dev/log
                echo -n "."
        done
        echo " done"
        touch /var/lock/subsys/syslog
        ;;
  stop)
        echo -n "Shutting down system loggers: "
        killproc syslogd
        killproc klogd
        echo
        rm -f /var/lock/subsys/syslog
        ;;
  *)
        echo "Usage: syslog {start|stop}"
        exit 1
esac

exit 0
```

Notez que vous n'avez pas à mettre tous les systèmes virtuels de fichiers sur un seul disque. Cependant, vous devrez lancer un syslog différent pour chaque partition qui contient un système virtuel de fichiers..

7 FTP virtuel

Wu-ftpd intégre en standard le support des domaines virtuels. Cependant, vous ne pouvez pas utiliser des fichiers de mot de passe différents pour chaque domaine. Par exemple, si `bob@domaine1.com` et `bob@domaine2.com` désirent tous les deux un compte, vous devrez donner à l'un des deux le nom `bob2` ou demander à un des utilisateurs de choisir un autre nom de login. Puisque vous disposez à présent de systèmes de fichiers différents pour chaque domaine, vous disposez de fichiers de mot de passe différents et ce problème disparaît. Vous n'avez qu'à créer un script `virtnewuser` et `virtpasswd` de la façon qui est expliquée ci-dessus, et tout fonctionnera. Vous pouvez également disposer d'un ftp *anonymous* dans chaque environnement virtuel,

puisque ce n'est pas affecté par le système de fichiers.

Les entrées pour wu-ftp dans `inetd.conf` sont :

```
ftp stream tcp nowait root /usr/bin/virtuald virtuald /virtual/conf.ftp wu.ftpd -l -a
```

8 Web virtuel

Apache supporte en standard la gestion des domaines virtuels. C'est d'ailleurs le seul programme pour lequel je recommande d'utiliser le système de gestion des domaines virtuels fourni avec. Lorsque vous lancez un programme par l'intermédiaire d'inetd, il y a un coût supplémentaire, puisque le programme doit démarrer à chaque fois qu'il y a une demande de connexion, et vous obtenez des temps de réponse beaucoup plus longs, inacceptables pour le web. Apache intégre également un mécanisme pour stopper les connexions lorsqu'elles sont trop nombreuses.

Si vous désirez quand même utiliser inetd pour lancer apache, ajoutez la ligne suivante dans votre fichier `inetd.conf` :

```
www stream tcp nowait www /usr/sbin/virtuald virtuald /virtual/conf.www httpd -f /var/www/conf/httpd.co
```

Dans le fichier `/var/www/conf/httpd.conf` vous devrez spécifier :

```
ServerType inetd
```

Puis configurez chaque serveur Apache comme pour un seul domaine.

A l'heure où j'écris ces lignes, il n'existe pas de HOWTO sur le web virtuel. Cependant, je crois qu'il ne va pas tarder à être écrit. Dans ce cas, je mettrai une référence à ce HOWTO. Si ce document tarde à paraître, et que j'ai assez de demandes, j'écrirai une petite partie sur la configuration d'Apache pour les domaines virtuels.

9 Mail virtuel avec Pop

9.1 Avis aux utilisateurs de QMail

Cette section d'applique uniquement a sendmail. Une section pour qmail sera ajoutée dans la prochaine version de ce document.

9.2 Problème

Le support du mail virtuel est une demande toujours grandissante. Sendmail affirme qu'il supporte le mail virtuel. En fait, il se contente d'être à l'écoute de mail pour différents domaines. Vous pouvez alors demander à faire suivre le mail quelque part. Cependant, si vous le faites suivre sur la machine locale et que vous avez du mail pour bob@domaine1.com et bob@domaine2.com, ils vont atteindre la même boîte. C'est un problème puisque les bob sont deux personnes différentes, avec du courrier électronique différent.

9.3 Une mauvaise solution

Vous pouvez vous assurer que chaque nom d'utilisateur est unique en utilisant une numérotation des noms d'utilisateurs :bob1, bob2, etc... Vous pourriez également hacker le mail et le pop pour que ces conversions soient invisibles, mais cela peut devenir désordonné. Le mail sortant a pour domaine domaineprincipal.com et vous désirez que chaque mail envoyé dans chaque sous-domaine ait une adresse From: différente.

9.4 Une bonne solution

Chaque système de fichiers virtuel fourni à chaque domaine un fichier `/etc/passwd`. Cela signifie que bob@domaine1.com et bob@domaine2.com sont des utilisateurs différents dans des fichiers `/etc/passwd` différents, donc le mail ne constituera aucun problème. Ils possèdent également chacun un spool de mail, donc les boîtes aux lettres seront des fichiers différents sur des système de fichiers virtuels différents.

Cependant, sendmail nécessite un changement mineur de son code source. Sendmail a un fichier nommé `/etc/sendmail.cw` qui contient tous les noms de machine pour lequel il délivrera le mail localement au lieu de le faire suivre à une autre machine. Sendmail fait une vérification interne de toutes les interfaces réseau de la machine pour initialiser cette liste avec les adresses IP locales. Cela présente un problème si vous envoyez de mails entre deux domaines virtuels de la même machine. Sendmail pensera que l'autre domaine virtuel est une adresse locale et il delivrera le mail localement. Par exemple, bob@domaine1.com envoie un mail à fred@domaine2.com. Puisque le sendmail de domaine1.com pense que domaine2.com est une adresse locale, il va envoyer ce mail à domaine1.com et ne l'enverra jamais à domaine2.com. Vous avez à modifier sendmail (ce que j'ai fait sans problème sur la version 8.8.5) :

```
vi v8.8.5/src/main.c # Environ a la ligne 494
Vous devriez remplacer la ligne :

load_if_names();

Par :

/* load_if_names(); Commente puisque cela casse les domaines virtuels */
```

Notez que cette modification n'est nécessaire que si vous désirez envoyer du mail entre des domaines virtuels, ce qui est probable, je pense.

Cela corrigera le problème. Cependant, l'adaptateur réseau principal eth0 n'est pas supprimé. Ainsi, si vous envoyez un mail depuis une adresse IP virtuelle vers une adresse sur eth0 de la même machine, il sera délivré localement. Pour cela, j'utilise une adresse IP bidon virtuel1.domaine.com (10.10.10.157). Je n'envoie jamais de mail à cet hôte, les domaines virtuels non plus. C'est aussi l'adresse IP que j'utiliserai pour me connecter sur la machine via ssh, pour vérifier si le système fonctionne.

Éditez `/etc/sendmail.cw` et mettez-y les domaines locaux.

```
vi /etc/sendmail.cw
mail.domaine1.com
```

```
domaine1.com
domaine1
localhost
```

Créez `/etc/sendmail.cf` comme vous le feriez normalement avec m4. J'utilise :

```
divert(0)dnl
VERSIONID(‘@(#)tcpproto.mc      8.5 (Berkeley) 3/23/96’)
OSTYPE(linux)
FEATURE(redirect)
FEATURE(always_add_domain)
FEATURE(use_cw_file)
FEATURE(local_procmail)
MAILER(local)
MAILER(smtp)
```

Editez `/etc/sendmail.cf` pour répondre sur voter domaine virtuel :

```
vi /etc/sendmail.cf # Vers la ligne 86
Vous devriez avoir :

#Dj$w.Foo.COM

Remplacez cette ligne par :

Djdomaine1.com
```

Sendmail ne peut pas être lancé tel quel, vous allez devoir le lancer à travers inetd. C'est un moyen inefficace qui implique un temps de réponse plus long, mais si vous avez un site si occupé pour que la différence soit importante, alors vous devriez utiliser une machine dédiée à ce site. Notez que vous de devez PAS utiliser l'option `-bd`. Notez également que vous devez lancer `sendmail -q` pour chaque domaine que vous gérez. Le nouveau fichier `sendmail.init` est le suivant :

```
#!/bin/sh

# Source function library.
. /etc/rc.d/init.d/functions

case "$1" in
  start)
        echo -n "Starting sendmail: "
        daemon sendmail -q1h
        echo
        echo -n "Starting virtual sendmail: "
```

```
        for i in /virtual/*
        do
                if [ ! -d "$i" ]
                then
                        continue
                fi
                if [ "$i" = "/virtual/lost+found" ]
                then
                        continue
                fi
                chroot $i sendmail -q1h
                echo -n "."
        done
        echo " done"
        touch /var/lock/subsys/sendmail
        ;;
  stop)
        echo -n "Stopping sendmail: "
        killproc sendmail
        echo
        rm -f /var/lock/subsys/sendmail
        ;;
  *)
        echo "Usage: sendmail {start|stop}"
        exit 1
esac

exit 0
```

Pop devrait s'installer normalement, sans effort supplémentaire. Vous n'avez qu'à modifier l'entrée pour pop dans le fichier `inetd.conf` pour utiliser le démon virtuald. Pour sendmail et pop, cela donne:

```
pop-3 stream tcp nowait root /usr/bin/virtuald virtuald /virtual/conf.pop in.qpop -s
smtp stream tcp nowait root /usr/bin/virtuald virtuald /virtual/conf.mail sendmail -bs
```

10 Le reste

Tout autre service devrait suivre une procédure similaire:

- Ajouter le binaire et les librairies au système virtuel de fichiers.
- L'ajouter dans `/etc/inetd.conf`
- Créer un fichier `/virtual/conf.service`.

– Créer tout script virtuel manquant.

J'ai fait des essais avec samba et j'ai changé un mot de passe pop virtuel à travers Eudora. Chacun fonctionne sans aucun problème. Si j'ai suffisamment de demandes, j'écrirai une section sur l'installation d'un samba virtuel.

11 Conclusion

Voici tout ce dont vous avez besoin. J'espère que cet article répond à vos attentes. Vous pouvez utiliser l'email à Computer Resource Center pour tout commentaire (NdT: en anglais bien sûr). Si vous avez une question, ou si vous me proposez une mise à jour, faîtes le moi savoir et je l'ajouterai.

12 FAQ

Q1. Pourquoi n'y a-t-il aucun question dans cette FAQ?

R1. Parce que personne n'en a encore posé.