# Mise en place d'un serveur de sauvegarde à moindre coût

Yann MORÈRE

19 février 2006

**Résumé**

Ce document propose la mise en place d'un serveur de sauvegarde simple, à l'aide d'outils libres et de quelques disques durs. Ce n'est pas effectivement une sauvegarde au sens pur du terme, mais cela permet une synchronisation automatique des fichiers à partir de postes Windows et Linux sur une machine distante avec une connexion sécurisée, sans l'intervention de l'utilisateur d'une part, et d'autre part un «*mirroring*» sur deux autres disques sur cette même machine distante.

## Table des matières

**1 Présentation du projet** **2**
1.1 Cahier des charges . . . 2
1.2 Solution matérielle . . . 2

**2 Quelques notions de cryptage** **2**

**3 Le RAID** **4**
3.1 Présentation de la technologie RAID . . . 4
3.2 Niveau 0 . . . 4
3.3 Niveau 1 . . . 5
3.4 Niveau 2 . . . 5
3.5 Niveau 3 . . . 5
3.6 Niveau 4 . . . 6
3.7 Niveau 5 . . . 6
3.8 Niveau 6 . . . 6
3.9 Comparaison . . . 6
3.10 Mise en place d'une solution RAID . . . 7

**4 OpenSSH** **7**
4.1 SSH, clé publique, clé privée, ... . . . 7
4.2 SSH : connexion automatique . . . 7

**5 Présentation de rsync** **8**

**6 Installation et utilisation de `cwrsync` sous windows** **8**
6.1 Génération de la clé DSA pour authentification . . . 8
6.2 Création de l'utilisateur de sauvegarde sur le serveur (sous linux) . . . 9
6.3 Connexion automatique sur le serveur avec `ssh` . . . 9
6.4 Sauvegarde par `rsync` . . . 9
6.5 Important : Limitation . . . 17

**7 Restauration des données** **17**
7.1 Configuration de Samba . . . 18
7.2 Connexion Du coté client . . . 19

**8 Installation et utilisation de `rsync` sous Linux/Unix** **20**

**9 Sauvegarde des données du serveur *Mirroring*** **21**

# 1 Présentation du projet

## 1.1 Cahier des charges

Le but était de réaliser la sauvegarde des fichiers et travaux des personnels du laboratoire. Cette sauvegarde devait être transparente et automatique pour les utilisateurs (evite les oublis).
De plus l'outil doit permettre de choisir simplement les fichiers à sauvegarder ainsi que la fréquence de sauvegarde.
Ensuite la mise en œuvre doit être indépendante du système d'exploitation et doit fonctionner de la même manière sur les système Unix et Windows du laboratoire.
Il faut aussi que la récupération des données soit aisée afin qu'il n'y ait pas d'intervention de la personne qui gère le serveur. Elle doit être aussi très souple afin de permettre la restauration d'un fichier comme d'une arborescence.
Le transfert des données sur le réseau doit être sécurisées.
Bien sur tout ceci avec un coût minimum pour le laboratoire.

## 1.2 Solution matérielle

Elle s'articule autour d'un PC dépassé, un Pentium III 500Mhz 256Mo de Ram, dans lequel nous avons placé un carte UDMA 133 (30 €), le disque dur d'origine de 6Go pour le système et 3 disque UDMA 133 de 120Go (300€ à l'époque 6 mois) pour les sauvegardes, et une carte réseau 100Mbits.
Bien sur les puristes me diront qu'il ne s'agit pas d'une vrai sauvegarde, dans le sens ou toutes les données sont placées au même endroit sur des supports non amovibles.
Mais cette solution est pour l'instant le meilleur compromis, prix, rapidité (sauvegarde et restauration). Le premier disque sert de sauvegarde principale et les deux autres servent pour un mirroring alterné (1 jour sur 2). La probabilité que les 3 disques tombent en panne en même temps est faible (sauf s'il y a un défaut sur la série :-(, car ils sont identiques).

# 2 Quelques notions de cryptage

source `http://www.devparadise.com/technoweb/secure/a413.php`.
Pour pouvoir sécuriser des données transitant sur internet, on utilise la cryptographie. Les données à protéger sont encodées à l'aide d'un algorithme (formule mathématique +/- complexe) basé sur une chaîne de caractères appelée clé de cryptage.
Il existe 3 méthodes de cryptage :

– le cryptage symétrique ou cryptage à clé privée ou cryptage à clé secrète : Cette méthode utilise une seule clé appelée clé privée, commune à l'expéditeur et au destinataire qui est indispensable pour permettre à l'algorithme de crypter et décrypter les données.

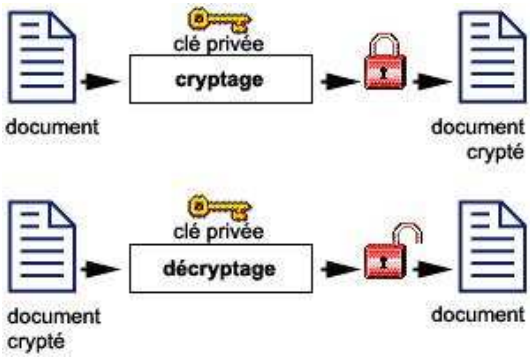



FIG. 1 – crytage symétrique

Dans ce cas, les algorithmes utilisent des codage par chaîne ou par bloc (ex : DES - Data Encryption Standard : le standard de cryptologie depuis un quart de siècle.) Cette méthode est malheureusement la moins fiable du fait de l'utilisation d'une seule clé qu'il faut bien sûr fournir aux deux protagonistes (expéditeur et destinataire). La taille de la clé est aussi déterminante or en France on ne peut utiliser que des clés de 40 bits qui est rapidement ń craquable ż.

– le cryptage asymétrique ou cryptage à clé publique : l'algorithme utilisé dans ce cas utilise deux clés :
  – une clé publique connue par tous qui est utilisée pour le cryptage des données
  – une clé privée détenue par le destinataire qui est donc le seul à pouvoir décrypter le document.

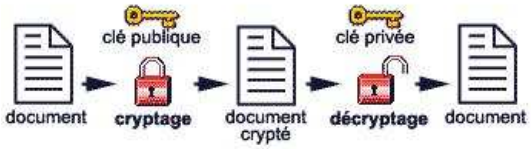



FIG. 2 – crytage asymétrique

– le hashing : On crée un condensât du message qui permet d'obtenir une image condensée spécifique au message. Le destinataire recalculera le condensât pour vérifier l'authenticité du message.
  Le cryptage asymétrique est utilisé pour le cryptage / décryptage de document mais aussi pour authentifier des documents grâce à la signature numérique.
  Cette signature permet :
  – de vérifier l'identité de l'expéditeur du message qui est le seul grâce à sa clé privée à pouvoir créer la signature)
  – d'assurer que le message n'a pas été modifier par un tiers.
  L'expéditeur utilise un algorithme de hachage pour créer un condensât. Ce condensât est crypté à l'aide de la clé privée que seul l'expéditeur connaît. Le résultat obtenu s'appelle la signature. Le message est envoyé avec cette signature. Le destinataire reçoit message et signature.
  Il décrypte la signature à l'aide d'une clé publique fournie par l'expéditeur au préalable. Il obtient ainsi un condensât. A partir du message, il exécute le même algorithme de hachage que l'expéditeur pour obtenir un condensât. Si les deux condensâts ainsi obtenus sont identique, l'expéditeur aura la certitude quie le document est authentique (sous la condition que la clé publique utilisée soit bien celle envoyée par l'expéditeur.

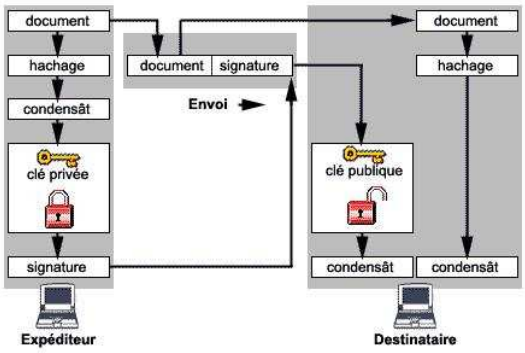



FIG. 3 – document avec signature

Pour vérifier qu'un clé publique appartient bien à un émetteur donné, on utilise un certificat. Celui-ci, est fourni par une partie tierce (l'autorité de certification ex : verisign.com) qui utilise sa propre clé privée (clé confidentielle) pour générer un document électronique contenant :
– des informations concernant le client (expéditeur)
– la clé publique du client
– la signature de l'autorité de certification.

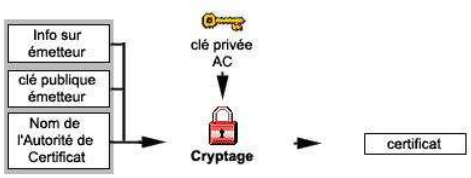



FIG. 4 – création de certificat

# 3 Le RAID

Cette partie est issue de `http://www.commentcamarche.net/`

## 3.1 Présentation de la technologie RAID

La technologie RAID (acronyme de Redundant Array of Inexpensive Disks, parfois Redundant Array of Independent Disks, traduisez Ensemble redondant de disques indépendants) permet de constituer une unité de stockage à partir de plusieurs disques durs. L'unité ainsi créée (appelée grappe) a donc une grande tolérance aux pannes (haute disponibilité), ou bien une plus grande capacité/vitesse d'écriture. La répartition des données sur plusieurs disques durs permet donc d'en augmenter la sécurité et de fiabiliser les services associés.
Cette technologie a été mise au point en 1987 par trois chercheurs (Patterson, Gibson et Katz) à l'Université de Californie (Berkeley). Depuis 1992 c'est le RAID Advisory Board qui gère ces spécifications. Elle consiste à constituer un disque de grosse capacité (donc coûteux) à l'aide de plus petits disques peu onéreux (c'est-à-dire dont le MTBF, Mean Time Between Failure, soit le temps moyen entre deux pannes, est faible).
Les disques assemblés selon la technologie RAID peuvent être utilisés de différentes façons, appelées Niveaux RAID. L'Université de Californie en a défini 5, auxquels ont été ajoutés les niveaux 0 et 6. Chacun d'entre-eux décrit la manière de laquelle les données sont réparties sur les disques :
– Niveau 0 : appelé striping
– Niveau 1 : appelé mirroring, shadowing ou duplexing
– Niveau 2 : appelé striping with parity (obsolète)
– Niveau 3 : appelé disk array with bit-interleaved data
– Niveau 4 : appelé disk array with block-interleaved data
– Niveau 5 : appelé disk array with block-interleaved distributed parity
– Niveau 6 : appelé disk array with block-interleaved distributed parity
Chacun de ces niveaux constitue un mode d'utilisation de la grappe, en fonction :
– des performances
– du coût
– des accès disques

## 3.2 Niveau 0

Le niveau RAID-0, appelé striping (traduisez entrelacement ou agrégat par bande, parfois injustement appelé stripping) consiste à stocker les données en les répartissant sur l'ensemble des disques de la grappe. De cette façon, il n'y a pas de redondance, on ne peut donc pas parler de tolérance aux pannes. En effet en cas de défaillance de l'un des disques, l'intégralité des données réparties sur les disques sera perdue.
Toutefois, étant donné que chaque disque de la grappe a son propre contrôleur, cela constitue une solution offrant une vitesse de transfert élevée.
Le RAID 0 consiste ainsi en la juxtaposition logique (agrégation) de plusieurs disques durs physiques. En mode RAID-0 les données sont écrites par "bandes" (en anglais stripes) :

| Disque 1 | Disque 2 | Disque 3 |
|---|---|---|
| Bande 1 | Bande 2 | Bande 3 |
| Bande 4 | Bande 5 | Bande 6 |
| Bande 7 | Bande 8 | Bande 9 |

Tab. 1 – Raid 0

On parle de facteur d'entrelacement pour caractériser la taille relative des fragments (bandes) stockés sur chaque unité physique. Le débit de transfert moyen dépend de ce facteur (plus petite est chaque bande, meilleur est le débit).
Si un des élements de la grappe est plus grand que les autres, le système de remplissage par bande se trouvera bloqué lorsque le plus petit des disques sera rempli. La taille finale est ainsi égale au double de la capacité du plus petit des deux disques :
– deux disques de 20 Go donneront un disque logique de 40 Go.

– un disque de 10 Go utilisé conjointement avec un disque de 27 Go permettra d'obtenir un disque logique de 20 Go (17 Go du second disque seront alors inutilisés).

**Nota :** Il est recommandé d'utiliser des disques de même taille pour faire du RAID-0 car dans le cas contraire le disque de plus grande capacité ne sera pas pleinement exploité.

## 3.3 Niveau 1

Le niveau 1 a pour but de dupliquer l'information à stocker sur plusieurs disques, on parle donc de mirroring, ou shadowing pour désigner ce procédé.

| Disque 1 | Disque 2 | Disque 3 |
|---|---|---|
| Bande 1 | Bande 1 | Bande 1 |
| Bande 2 | Bande 2 | Bande 2 |
| Bande 3 | Bande 3 | Bande 3 |

Tab. 2 – Raid 1

On obtient ainsi une plus grande sécurité des données, car si l'un des disques tombe en panne, les données sont sauvegardées sur l'autre. D'autre part, la lecture peut être beaucoup plus rapide lorsque les deux disques sont en fonctionnement. Enfin, étant donné que chaque disque possède son propre contrôleur, le serveur peut continuer à fonctionner même lorsque l'un des disques tombe en panne, au même titre qu'un camion pourra continuer à rouler si un de ses pneus crève, car il en a plusieurs sur chaque essieu... En contrepartie la technologie RAID1 est très onéreuse étant donné que seule la moitié de la capacité de stockage n'est effectivement utilisée.

## 3.4 Niveau 2

Le niveau RAID-2 est désormais obsolète, car il propose un contrôle d'erreur par code de Hamming (codes ECC - Error Correction Code), or ce dernier est désormais directement intégré dans les contrôleurs de disques durs.
Cette technologie consiste à stocker les données selon le même principe qu'avec le RAID-0 mais en écrivant sur une unité distincte les bits de contrôle ECC (généralement 3 disques ECC sont utilisés pour 4 disques de données).
La technologie RAID 2 offre de piètres performances mais un niveau de sécurité élevé.

## 3.5 Niveau 3

Le niveau 3 propose de stocker les données sous forme d'octets sur chaque disque et de dédier un des disques au stockage d'un bit de parité.

| Disque 1 | Disque 2 | Disque 3 | Disque 4 |
|---|---|---|---|
| Bande 1 | Bande 2 | Bande 3 | Parité 1+2+3 |
| Bande 4 | Bande 5 | Bande 6 | Parité 4+5+6 |
| Bande 7 | Bande 8 | Bande 9 | Parité 7+8+9 |

Tab. 3 – Raid 3

De cette manière, si l'un des disques venait à défaillir, il serait possible de reconstituer l'information à partir des autres disques. Après "reconstitution" le contenu du disque défaillant est de nouveau intègre. Par contre, si deux disques venaient à tomber en panne simultanément, il serait alors impossible de remédier à la perte de données.

## 3.6 Niveau 4

Le niveau 4 est très proche du niveau 3. La différence se trouve au niveau de la parité, qui est faite sur un secteur (appelé bloc) et non au niveau du bit, et qui est stockée sur un disque dédié. C'est-à-dire plus précisément que la valeur du facteur d'entrelacement est différente par rapport au RAID 3.

| Disque 1 | Disque 2 | Disque 3 | Disque 4 |
|---|---|---|---|
| Block 1 | Block 2 | Block 3 | Parité 1+2+3 |
| Block 4 | Block 5 | Block 6 | Parité 4+5+6 |
| Block 7 | Block 8 | Block 9 | Parité 7+8+9 |

Tab. 4 – Raid 4

Ainsi, pour lire un nombre de blocs réduits, le système n'a pas à accéder à de multiples lecteurs physiques, mais uniquement à ceux sur lesquels les données sont effectivement stockées. En contrepartie le disque hébergeant les données de contrôle doit avoir un temps d'accès égal à la somme des temps d'accès des autres disques pour ne pas limiter les performances de l'ensemble.

## 3.7 Niveau 5

Le niveau 5 est similaire au niveau 4, c'est-à-dire que la parité est calculée au niveau d'un secteur, mais répartie sur l'ensemble des disques de la grappe.

| Disque 1 | Disque 2 | Disque 3 | Disque 4 |
|---|---|---|---|
| Block 1 | Block 2 | Block 3 | Parité 1+2+3 |
| Block 4 | Parité 4+5+6 | Block 5 | Block 6 |
| Parité 7+8+9 | Block 7 | Block 8 | Block 9 |

Tab. 5 – Raid 5

De cette façon, RAID 5 améliore grandement l'accès aux données (aussi bien en lecture qu'en écriture) car l'accès aux bits de parité est réparti sur les différents disques de la grappe.
Le mode RAID-5 permet d'obtenir des performances très proches de celles obtenues en RAID-0, tout en assurant une tolérance aux pannes élevée, c'est la raison pour laquelle c'est un des modes RAID les plus intéressants en terme de performance et de fiabilité. Nota L'espace disque utile sur une grappe de n disques étant égal à n-1 disques, il est intéressant d'avoir un grand nombre de disques pour "rentabiliser" le RAID-5.

## 3.8 Niveau 6

Le niveau 6 a été ajouté aux niveaux définis par Berkeley. Il définit l'utilisation de 2 fonctions de parité, et donc leur stockage sur deux disques dédiés. Ce niveau permet ainsi d'assurer la redondance en cas d'avarie simultanée de deux disques. Cela signifie qu'il faut au moins 4 disques pour mettre en oeuvre un système RAID-6.

## 3.9 Comparaison

Les solutions RAID généralement retenues sont le RAID de niveau 1 et le RAID de niveau 5.
Le choix d'une solution RAID est lié à trois critères :
– la sécurité : RAID 1 et 5 offrent tous les deux un niveau de sécurité élevé, toutefois la méthode de reconstruction des disques varie entre les deux solutions. En cas de panne du système, RAID 5 reconstruit le disque manquant à partir des informations stockées sur les autres disques, tandis que RAID 1 opère une copie disque à disque.

– Les performances : RAID 1 offre de meilleures performances que RAID 5 en lecture, mais souffre lors d'importantes opérations d'écriture.
– Le coût : le coût est directement lié à la capacité de stockage devant être mise en oeuvre pour avoir une certaine capacité effective. La solution RAID 5 offre un volume utile représentant 80 à 90% du volume alloué (le reste servant évidemment au contrôle d'erreur). La solution RAID 1 n'offre par contre qu'un volume disponible représentant 50% du volume total (étant donné que les informations sont dupliquées).

### 3.10 Mise en place d'une solution RAID

Il existe plusieurs façons différentes de mettre en place une solution RAID sur un serveur :
– de façon logicielle : il s'agit généralement d'un driver au niveau du système d'exploitation capable de créer un seul volume logique avec plusieurs disques (SCSI ou IDE).
– de façon matérielle
  – avec des matériels DASD (Direct Access Stockage Device) : il s'agit d'unités de stockage externes pourvues d'une alimentation propre. De plus ces matériels sont dotés de connecteurs permettant l'échange de disques à chaud (on dit généralement que ce type de disque est hot swappable). Ce matériel gère lui-même ses disques, si bien qu'il est reconnu comme un disque SCSI standard.
  – avec des contrôleurs de disques RAID : il s'agit de cartes s'enfichant dans des slots PCI ou ISA et permettant de contrôler plusieurs disques durs.

## 4 OpenSSH

OpenSSH est une version LIBRE de la suite d'outils du protocole SSH de connexion réseau utilisée par un nombre croissant de personnes sur l'Internet. De nombreux utilisateurs de telnet, rlogin, ftp et autres programmes identiques ne réalisent pas que leur mot de passe est transmis non chiffré à travers l'Internet. OpenSSH chiffre tout le trafic (mots de passe inclus) de façon à déjouer les écoutes réseau, les prises de contrôle de connexion, et autres attaques. De plus, OpenSSH fournit toute une palette de possibilités de tunnellisation et de méthodes d'authentification.

### 4.1 SSH, clé publique, clé privée, ...

SSH (Secure Shell) est un protocole encrypté d'accès distant aux machines. Il permet de se connecter facilement et sécuritairement à une machine distante. Voici quelques trucs par rapport à son utilisation avec des clés publiques, clés privées
L'utilité du système clé publique/clé privée est d'éviter l'entrée du mot de passe pour l'utilisateur. C'est un système plus sécuritaire que celui du mot de passe car il nécessite la possession de la clé privée qui correspond à la clé publique. C'est ce qu'on appelle de la cryptographie asymétrique.
Le cryptage asymétrique consiste à utiliser des clés différentes pour le cryptage et le décryptage.
Chaque correspondant :
– diffuse une clé publique
– conserve sa clé privée
Un message peut être crypté avec la clé publique du destinataire afin que seul celui-ci puisse le lire (avec sa clé privée).
Dans cette optique, une communication cryptée bidirectionnelle implique que les 2 correspondants disposent de clés certifiées (ce qui n'est pas souvent le cas pour le commun des mortels sur le Web) et surtout un temps de traitement relativement lourd lié au modèle de cryptage asymétrique (implémenté au travers de systèmes tels que DSA, DSS...).
Des exemples de cryptage asymétrique sont :
– DSA
– DSS
– RSA (disponible dans la JCE)
– Diffie-Hellman (disponible dans la JCE).

### 4.2 SSH : connexion automatique

Pour communiquer avec un serveur et s'y identifier avec une clé publique/clé privée, voici quelques éléments et trucs simples... Pour créer sa paire clé publique/clé privée :

– Commande : `ssh-keygen -t dsa`
– Explications : On vous demandera quel nom donner à votre clé (juste à taper retour, la paire de clé va s'enregistrer dans votre répertoire `.ssh` situé dans votre home). Ensuite, on vous demandera de taper une passphrase. Cette passphrase servira de porte d'accès à la clé publique. On peut ne pas en taper, cependant, cela revient à faire entièrement confiance au système de privilèges de linux. Ce passphrase sera demandé lorsque l'on veut s'authentifier avec le système clé publique-clé privée (on peut aussi avec la commande ssh-add ajouter la clé et son passphrase pour éviter cela plus tard).

Vous retrouverez donc la clé publique `id_dsa.pub` ainsi que votre clé privée `id_dsa` dans votre dossier `.ssh`. SSH fait des vérifications des privilèges de ces clés, `id_dsa` (la clé privée) ne doit être accessible qu'en lecture et seulement à vous (sinon, ça ne fonctionnera pas)
On aurait aussi pu faire la commande `ssh-keygen -t rsa` pour générer des clés de type RSA , mais l'algorithme RSA est toujours sous instance de brevet, DSA est totalement libre.
Pour se connecter via ssh à une autre machine :
– Commande : `ssh -C utilisateur@lenomdelamachine`

Pour se connecter à une autre machine avec sa clé publique
– Commandes : Placer votre clé publique (le contenu du fichier `id_dsa.pub` sur le serveur distant, dans le répertoire `.ssh` du compte sur lequel vous désirez vous connecter, dans un fichier nommé `authorized_keys`.
– Explications : Cela va fonctionner dans la plupart des cas, ça nécessite cependant que le serveur ssh sur la machine distante permette l'utilisation de l'identification par clé. Il faut que le fichier `sshd_config` (situé dans le dossier `/etc`) contienne les ligne suivantes (qui sont commentées parfois)

```
PubkeyAuthentication yes
AuthorizedKeysFile      .ssh/authorized_keys
```

# 5 Présentation de rsync

rsync est un programme de transfert de fichier pour les systèmes Unix. Il utilise son propore algorithme pour le transfert qui lui permet de fournir une méthode très rapide pour rapattrier des fichiers pour la synchronisation.
Ceci est possible parce qu'il envoie juste les données différentes d'un fichier à travers le réseau.
Voici les principales caractéristiques de rsync :
– peut mettre à jour des système de fichiers entiers et arborescences
– peut préserver (optionnellement) les liens et liens symboliques, les droits d'accès, l'appartennance, les fichiers périphériques et les dates.
– ne demande pas de privilège spécifique pour l'installation
– capable d'utiliser rsh, ssh ou les sockets directes pour le transport de fichier

# 6 Installation et utilisation de `cwrsync` sous windows

L'utilitaire `rsync` va permettre de réaliser les sauvegardes sur le serveur. Cette sauvegarde sera intelligente car la commande ne fait que synchroniser les fichiers entre la machine et le serveur de sauvegarde. Seule la première sauvegarde transfèrera l'intégralité des données. Par la suite, seuls les fichiers modifiés entre deux sauvegardes seront transférés.
Cet utilitaire est disponible en standard sur les systèmes Unix, mais n'existe pas sur les systèmes Windows. Il faut pour cela, soit utiliser les outils Cygwin en installant l'environnement, soit en récupérant seulement l'utilitaire `rsync`.
Ce dernier est disponible sur le site `http://www.itefix.no/phpws/index.php`, il s'agit de cwrsync qui se presente sous la forme d'un installable.
Vous pouvez aussi télécharger le paquet suivant qui contient tout ce qui est nécessaire pour une installation sous windows (script d'installation, `scp`, exemple de fichier de configuration de sauvegarde) `lcwrsync.zip`
En tant qu'administrateur, installer le logiciel, en gardant les options par défaut.
Dans une fenêtre de commande (démarrer->executer-> taper cmd puis ok) on va générer les fichiers necessaires à la connexion.

## 6.1 Génération de la clé DSA pour authentification

```
c:\cwrsync\ssh-keygen -t dsa -f c:\cwrsync\id_dsa
```

Un jeu de clé publique et privée est généré et est copié dans le répertoire `cwrsync`.
Si aucun fichier n'est spécifié, les clés sont copiées dans le répertoire `/cygdrive/c/Documents and Settings/user/.ssh/i`
et `/cygdrive/c/Documents and Settings/user/.ssh/id_dsa.pub` par défaut.
Afin de mieux sécuriser le système il est possible de donnée une «*passphrase*» lors de la création des clés.

## 6.2 Création de l'utilisateur de sauvegarde sur le serveur (sous linux)

sur le serveur Unix :

```
adduser morere #ajoute l'utilisateur morere
smbpasswd -a morere # ajoute l'utilisateur samba morere
su morere
ssh-keygen -t dsa
```

La dernière commande génère les clés du coté serveur et crée le répertoire `.ssh`.

## 6.3 Connexion automatique sur le serveur avec `ssh`

On va copier la clé publique du client sur le serveur dans le répertoire de `user/.ssh` afin de réaliser une connexion automatique.
La commande `scp`, n'existe pas non plus disponible en standard sous windows. `Putty` fournit la commande necessaire par `pscp` disponible sur `http://www.chiark.greenend.org.uk/~sgtatham/putty/`.
*Attention, la version 0.52 pose des problèmes avec XP, j'ai utilisé la version 0.55.*

```
pscp "C:\Documents and Settings\user\.ssh\id_dsa.pub"
morere@1974.57.140.212:.ssh/authorized_keys
```

Un test de connexion est réalisé par la commande

```
ssh morere@194.57.140.212
```

ou dans le cas ou on spécifie le nom de fichier de clé :

```
pscp "C:\cwrsync\id_dsa.pub" morere@1974.57.140.212:.ssh/authorized_keys
```

Un test de connexion est réalisé par la commande

```
ssh -i c:\cwrsync\id_dsa morere@194.57.140.212
```

et la connexion devrait être automatique. L'option `-i` permet de spécifier ou `ssh` doit aller chercher la clé privée pour valider la connexion.
Dans le cas ou plusieurs machines se connectent sur le même compte pour faire des sauvegardes, il faut concaténer à la main les clés publiques dans le fichier `$HOME/.ssh/authorized_keys` sur le serveur.

## 6.4 Sauvegarde par `rsync`

Maintenant il faut envoyer par rsync les fichiers sur le serveur.
Il existe un problème avec les noms de fichiers longs et les `C:\Documents and Settings\user` par exemples, que `rsync` refuse.
Une première astuce est de se placer dans le répertoire qui convient, et d'appeller la commande `rsync` avec son chemin. Voici un exemple de fichier «`sauv.bat`» que l'on place dans le répertoire `.ssh` de l'utilisateur.

```
@ECHO OFF
cd C:\Documents and Settings\user
c:\cwrsync\rsync -av --rsh="c:\cwrsync\ssh -i c:\cwrsync\id_dsa -l morere"
"Bureau" 194.57.140.212:backup
c:\cwrsync\rsync -av --rsh="c:\cwrsync\ssh -i c:\cwrsync\id_dsa -l morere"
"Favoris" 194.57.140.212:backup
c:\cwrsync\rsync -av --rsh="c:\cwrsync\ssh -i c:\cwrsync\id_dsa -l morere" "Mes
documents" 194.57.140.212:backup
```

Une autre astuce, beaucoup plus propre est de passer par l'utilisation de «/cygdrive/c/» qui remplace «c:\» et permet d'utiliser des chemins absolus.

```
@ECHO OFF
echo utilisation de /cygdrive/c/Documents and Settings/user/
c:\cwrsync\rsync -av --rsh="c:\cwrsync\ssh -i c:\cwrsync\id_dsa -l morere"
"/cygdrive/c/Documents and Settings/user/Bureau" 194.57.140.212:backup
c:\cwrsync\rsync -av --rsh="c:\cwrsync\ssh -l morere"
"/cygdrive/c/Documents and Settings/user/Favoris" 194.57.140.212:backup
c:\cwrsync\rsync -av --rsh="c:\cwrsync\ssh -l morere"
"/cygdrive/c/Documents and Settings/user/Mes documents" 194.57.140.212:backup
```

Ces actions permettent de sauver les répertoires `Bureau`, `Favoris` et `Mes Documents` sur le serveur de sauvegarde dans le répertoire `backup`.
Rsync possède de nombreuses options listées ci-dessous

```
 -v, --verbose               increase verbosity
 -q, --quiet                 decrease verbosity
 -c, --checksum              always checksum
 -a, --archive               archive mode, equivalent to -rlptgoD
 -r, --recursive             recurse into directories
 -R, --relative              use relative path names
     --no-relative           turn off --relative
     --no-implied-dirs       don't send implied dirs with -R
 -b, --backup                make backups (see --suffix & --backup-dir)
     --backup-dir            make backups into this directory
     --suffix=SUFFIX         backup suffix (default ~ w/o --backup-dir)
 -u, --update                update only (don't overwrite newer files)
 -l, --links                 copy symlinks as symlinks
 -L, --copy-links            copy the referent of all symlinks
     --copy-unsafe-links     copy the referent of "unsafe" symlinks
     --safe-links            ignore "unsafe" symlinks
 -H, --hard-links            preserve hard links
 -p, --perms                 preserve permissions
 -o, --owner                 preserve owner (root only)
 -g, --group                 preserve group
 -D, --devices               preserve devices (root only)
 -t, --times                 preserve times
 -S, --sparse                handle sparse files efficiently
 -n, --dry-run               show what would have been transferred
 -W, --whole-file            copy whole files, no incremental checks
     --no-whole-file         turn off --whole-file
 -x, --one-file-system       don't cross filesystem boundaries
 -B, --block-size=SIZE       checksum blocking size (default 700)
 -e, --rsh=COMMAND           specify the remote shell
     --rsync-path=PATH       specify path to rsync on the remote machine
     --existing              only update files that already exist
     --ignore-existing       ignore files that already exist on receiver
     --delete                delete files that don't exist on sender
     --delete-excluded       also delete excluded files on receiver
     --delete-after          receiver deletes after transfer, not before
     --ignore-errors         delete even if there are I/O errors
     --max-delete=NUM        don't delete more than NUM files
     --partial               keep partially transferred files
     --force                 force deletion of dirs even if not empty
     --numeric-ids           don't map uid/gid values by user/group name
     --timeout=TIME          set I/O timeout in seconds
 -I, --ignore-times          turn off mod time & file size quick check
     --size-only             ignore mod time for quick check (use size)
```

```
     --modify-window=NUM     compare mod times with reduced accuracy
 -T  --temp-dir=DIR          create temporary files in directory DIR
     --compare-dest=DIR      also compare received files relative to DIR
     --link-dest=DIR         create hardlinks to DIR for unchanged files
 -P                          equivalent to --partial --progress
 -z, --compress              compress file data
 -C, --cvs-exclude           auto ignore files in the same way CVS does
     --exclude=PATTERN       exclude files matching PATTERN
     --exclude-from=FILE     exclude patterns listed in FILE
     --include=PATTERN       don't exclude files matching PATTERN
     --include-from=FILE     don't exclude patterns listed in FILE
     --files-from=FILE       read FILE for list of source-file names
 -0  --from0                 all file lists are delimited by nulls
     --version               print version number
     --daemon                run as an rsync daemon
     --no-detach             do not detach from the parent
     --address=ADDRESS       bind to the specified address
     --config=FILE           specify alternate rsyncd.conf file
     --port=PORT             specify alternate rsyncd port number
     --blocking-io           use blocking I/O for the remote shell
     --no-blocking-io        turn off --blocking-io
     --stats                 give some file transfer stats
     --progress              show progress during transfer
     --log-format=FORMAT     log file transfers using specified format
     --password-file=FILE    get password from FILE
     --bwlimit=KBPS          limit I/O bandwidth, KBytes per second
     --write-batch=PREFIX    write batch fileset starting with PREFIX
     --read-batch=PREFIX     read batch fileset starting with PREFIX
 -h, --help                  show this help screen
```

Ce qu'il faut retenir c'est que `-a` remplace `-rlptgoD` qui est à peut près tout ce dont on a besoin. On peut ajouter `-v` pour avoir le mode verbeux.
**Attention :** Dans le cas où les utilisateurs de windows ont configuré leur compte de telle manière que les autres utilisateurs de la machine ne puissent pas accéder à leurs fichiers (options à la création des comptes windows), l'option `-a` et plus particulèrement l'option `-p` (preserve permission) pose problème. En effet un problème de gestion de permission entre rsync et windows, fait que les fichiers et répertoire sur la machine de sauvegarde sont créés avec des droits erronés, ce qui empêche la sauvegarde des sous répertoires. Pour les clients windows, pour éviter tout problème, je vous conseille d'utiliser la ligne de commande suivante :

```
c:\cwrsync\rsync -rltgoDv --rsh="c:\cwrsync\ssh -i c:\cwrsync\id_dsa -l morere"
"/cygdrive/c/Documents and Settings/user/Bureau" 194.57.140.212:backup
```

```
-a, --archive
    Ceci est équivalent à -rlptgoD. C'est un moyen rapide de dire que vous
 voulez les sous répertoire en préservant tout.
```

Pour aller plus loin, il faudrait stocker dans un fichier les répertoires à sauver et les répertoires à exclure. Ceci peut être fait avec les options

```
--exclude=PATTERN
    This option allows you to selectively exclude certain files from the list of
 files to be transferred. This is most useful in combination with a recursive
transfer.
    You may use as many --exclude options on the command line as you like to
build up the list of files to exclude.
    See the EXCLUDE PATTERNS section for detailed information on this option.
--exclude-from=FILE
    This option is similar to the --exclude option, but instead it adds all
exclude patterns listed in the file FILE to the exclude list. Blank lines in
```

```
FILE and lines starting with ';' or '#' are ignored. If FILE is - the list will
be read from standard input.
--include=PATTERN
    This option tells rsync to not exclude the specified pattern of filenames.
This is useful as it allows you to build up quite complex exclude/include rules.
    See the EXCLUDE PATTERNS section for detailed information on this option.
--include-from=FILE
    This specifies a list of include patterns from a file. If FILE is - the list
 will be read from standard input.
--files-from=FILE
    Using this option allows you to specify the exact list of files to transfer
(as read from the specified FILE or "-" for stdin). It also tweaks the default
behavior of rsync to make transferring just the specified files and directories
easier. For instance, the --relative option is enabled by default when this
option is used (use --no-relative if you want to turn that off), all directories
specified in the list are created on the destination (rather than being noisily
skipped without -r), and the -a (--archive) option's behavior does not imply -r
(--recursive) -- specify it explicitly, if you want it.
    The file names that are read from the FILE are all relative to the source
dir -- any leading slashes are removed and no ".." references are allowed to go
higher than the source dir. For example, take this command:

    rsync -a --files-from=/tmp/foo /usr remote:/backup

    If /tmp/foo contains the string "bin" (or even "/bin"), the /usr/bin
directory will be created as /backup/bin on the remote host (but the contents of
 the /usr/bin dir would not be sent unless you specified -r or the names were
explicitly listed in /tmp/foo). Also keep in mind that the effect of the
(enabled by default) --relative option is to duplicate only the path info that
is read from the file -- it does not force the duplication of the source-spec
path (/usr in this case).
    In addition, the --files-from file can be read from the remote host instead
of the local host if you specify a "host:" in front of the file (the host must
match one end of the transfer). As a short-cut, you can specify just a prefix of
":" to mean "use the remote end of the transfer". For example:

    rsync -a --files-from=:/path/file-list src:/ /tmp/copy

    This would copy all the files specified in the /path/file-list file that was
 located on the remote "src" host.
```

Il est possible de faire plus générique, en plaçant dans des fichiers la liste des choses que l'on veut sauver et ce que l'on veut éviter.
Dans notre cas, on va sauver les répertoires `/cygdrive/c/Documents and Settings/user/`, c'est à dire les données de l'utilisateur `user` en evitant de sauver les fichiers inutiles comme `Application Data`.
fichier `sauv.bat`

```
@ECHO OFF
c:\cwrsync\rsync -av --include-from="c:\cwrsync\include.txt" --exclude-from="c:\
cwrsync\exclude.txt" --rsh="c:\cwrsync\ssh -l yannsauv" "/cygdrive/c/Documents a
nd Settings/Administrateur/" 194.57.140.212:backup_admin

yann@tuxpowered:~/temp/cwrsync$ more include.txt
Favoris
Mes documents
Bureau
yann@tuxpowered:~/temp/cwrsync$ more exclude.txt
.ssh
Application Data
```

```
Local Settings
Modèles
ntuser.ini
Voisinage réseau
Recent
SendTo
Voisinage d'impression
ntuser.dat.LOG
NTUSER.DAT
yann@tuxpowered:~/temp/cwrsync$
```

voici un autre exemple plus récent : fichier `sauv.bat`

```
@ECHO OFF
rem Sauvegarde du rep Documents and settings
c:\cwrsync\rsync -av --include-from="c:\cwrsync\includec.txt" --exclude-from="c:\cwrsync\excludec.tx
rem Sauvegarde du rep utilisateur
c:\cwrsync\rsync -av --include-from="c:\cwrsync\included.txt" --exclude-from="c:\cwrsync\excluded.tx
```

fichier `excludec.txt`

```
.ssh
Local Settings
Modèles
ntuser.ini
NTUSER.DAT
NTUSER.DAT.LOG
Voisinage réseau
Voisinage d'impression
SendTo
Menu Démarrer
Recent
```

fichier `excluded.txt`

```
HTML
devis_commandes
doc_materiel
fond d'ecran
icones
image_skel
sites_web
temp
```

Afin da faire une installation *quasi* automatique sur les postes clients, il est possible d'utiliser le petit script suivant (fichier `install_user.bat`) :

```
@ECHO OFF
echo Installation pour utilisateur %1
echo Génération des clés
c:\cwrsync\ssh-keygen -t dsa -f c:\cwrsync\id_dsa
echo Copie des clés sur le serveur
C:\cwrsync\scp  "c:\cwrsync\id_dsa.pub" %1@194.57.140.212:.ssh/authorized_keys
echo Test de connexion automatique
c:\cwrsync\ssh  -i c:\cwrsync\id_dsa %1@194.57.140.212
```

L'utilisation est assez simple, il suffit d'appeler le script avec le login de l'utilisateur à créer :

```
install_user morere
```

on peut aussi passer en paramètres le serveur de sauvegarde :

```
@ECHO OFF
echo Installation pour utilisateur %1 sur la machine %2
echo Génération des clés
c:\cwrsync\ssh-keygen -t dsa -f c:\cwrsync\id_dsa
echo Copie des clés sur le serveur
C:\cwrsync\scp  "c:\cwrsync\id_dsa.pub" %1@%2:.ssh/authorized_keys
echo Test de connexion automatique
c:\cwrsync\ssh  -i c:\cwrsync\id_dsa %1@%2
```

L'utilisation est assez simple, il suffit d'appeler le script avec le login de l'utilisateur à créer et la machine serveur :

```
install_user morere 194.57.140.212
```

Il suffit alors de répondre aux questions posées par les différents programmes.
De même, si vous avez deux disques durs locaux, il est possible de réaliser une sauvegarde synchronisée, toujours en utilisant `cwrsync`.
Le fichier `sauv_local.bat` permet de faire cela :

```
@ECHO OFF
echo Sauvegarde du rep Documents and settings
c:\cwrsync\rsync -av --include-from="c:\cwrsync\includec.txt"
--exclude-from="c:\cwrsync\excludec.txt" "/cygdrive/c/Documents and
Settings/yann/" "/cygdrive/h/utilisateurs/yann_windows/documents_and_settings"
echo Sauvegarde du rep utilisateur
c:\cwrsync\rsync -av --include-from="c:\cwrsync\included.txt"
--exclude-from="c:\cwrsync\excluded.txt" "/cygdrive/d/utilisateurs/yann/"
"/cygdrive/h/utilisateurs/yann_windows/"
```

Ensuite il est possible de réaliser cela de manière automatique, grâce au «Tâches Planifiées sous windows».
Menu Démarrer->Paramètres->Panneau de configuration...

Ensuite double click sur Tâche planifiée, puis parcourir

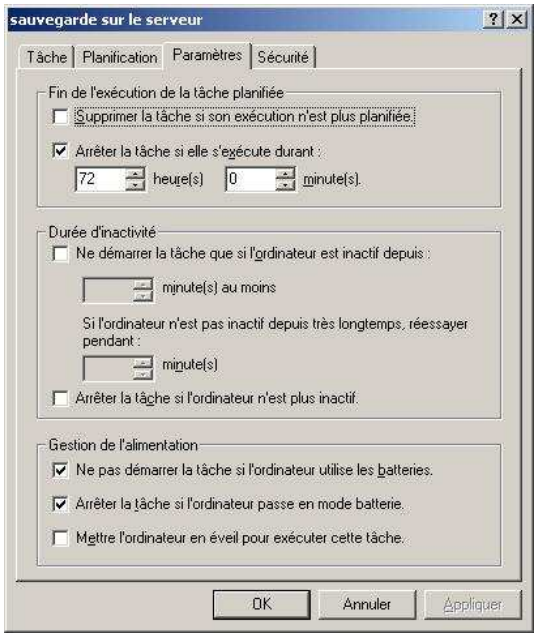
Choisir le nom de la tâche et la fréquence d'exécution puis suivant

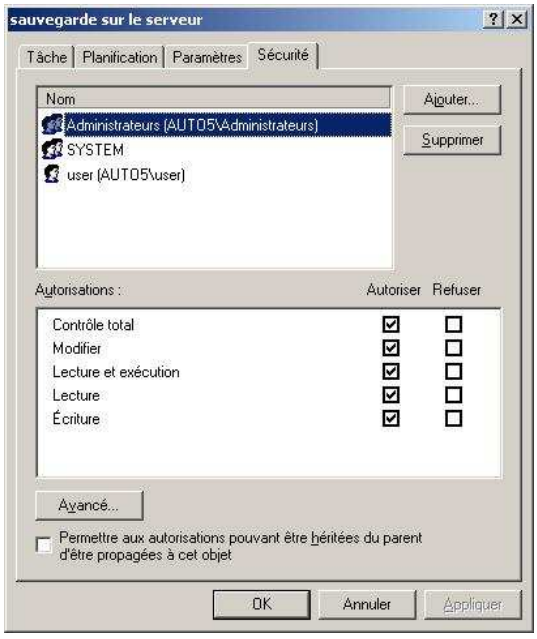
choisir l'heure de démarrage

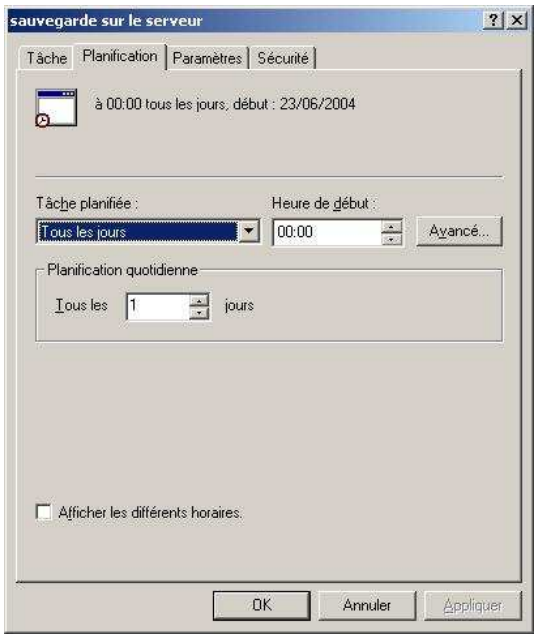
Choisir le programme qui doit être lancer par le planificateur. Dans notre cas il s'agira d'un fichier `bat` qui contiendra les commandes de sauvegarde.

Choisir l'utilisateur sous lequel va s'exécuter la sauvegarde

Visualiser les propriétés avancées





La tâche est maintenant planifiée.

### 6.5 Important : Limitation

Afin que cette tâche puisse être lancée par le planificateur, il est nécessaire que l'utilisateur soit loggé sur la machine (sinon les connexions réseau ne sont pas activées et le `rsync` via `ssh` ne peut pas fonctionner). Ainsi, si l'utilisateur doit quitter son bureau et que la sauvegarde est programmée à 00h00 tous les jours, il doit verrouiller sa station (touche Windows-L ou démarrer->verrouiller) et non se déconnecter.
Une piste pour résoudre ce problème, est de créer un service Windows qui fera le travail.

## 7 Restauration des données

Afin de simplifier le problème de la restauration des données, c'est un serveur samba qui a été choisi. Ce choix donne aussi de la souplesse aux utilisateurs, qui pourront choisir les fichiers à restaurer sur leur machine.

Bien sur il est tout à fait possible de restaurer les données de l'utilisateurs d'un seul coup en utilisant le script `restaure.bat` suivant :

```
@ECHO OFF
rem Sauvegarde du rep Documents and settings
c:\cwrsync\rsync -av --rsh="ssh -i c:\cwrsync\id_dsa -l morere" 194.57.140.212:windows/Documents_and
rem Sauvegarde du rep utilisateur
c:\cwrsync\rsync -av --rsh="ssh -i c:\cwrsync\id_dsa -l morere"  194.57.140.212:windows/utilisateurs
```

Il suffit d'intervertir l'ordre de la source et de la destination par rapport à la sauvegarde.

## 7.1 Configuration de Samba

Pour cela je vous renvoie à la page très bien faite de Yves Agostini `http://www.crium.univ-metz.fr/docs/system/samba/`.
mon fichier de configuration est le suivant `/etc/samba/smb.conf`

```
[global]
   workgroup = LASC
   netbios name = SERVEUR_KSTORE
   server string = Serveur Samba du LASC
   #optionnel  netbios aliases = SERVEUR_KSTORE

   os level = 99
   local master = yes

   printcap name = /etc/printcap
   load printers = yes

   socket options = TCP_NODELAY

   #security =  share
   #encrypt passwords = no
   security = user
   encrypt passwords = yes
   smb passwd file = /etc/samba/smbpasswd
   guest account = nobody
   hosts allow = 194.57.140. 127.

[public]
   comment = Public Stuff
   path = /home/samba
   public = yes
   writable = no
   printable = no
   browseable = yes

[homes]
   comment = Home Directories
   browseable = no
   writable = yes

[printers]
   comment = All Printers
   path = /var/spool/samba
   browseable = no
# Set public = yes to allow user 'guest account' to print
   public = yes
   writable = no
   printable = yes
```

l'ajout d'un utilisateur samba sur le serveur se fait par la commande

```
smbpasswd -a login
```

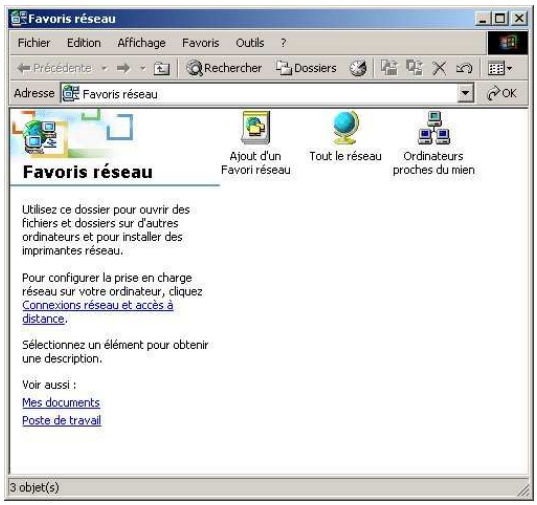
le système demande alors un mot de passe et un confirmation de mot de passe.

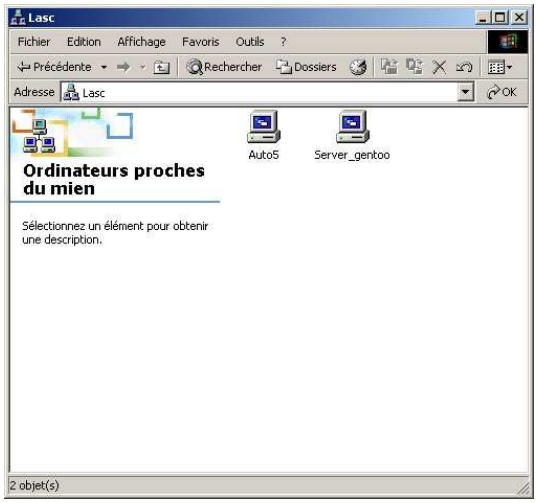
Il est possible d'utiliser pour samba les mêmes mots de passe que pour accéder à la machines windows. Dans ce cas, l'accès au serveur Samba sera quasiment transparent pour l'utilisateur (pas de login et pas de mot de passe).

Si les mots de passe sont différents, ce qui est préférable pour la sécurité, l'utilisateur, lors de la connexion sera invité à saisir un login et un mot de passe.

## 7.2 Connexion Du coté client

Par l'intermédaire du voisinage réseau, l'utilisateur se connecte sur le serveur Samba de son groupe de travail, par l'intermédiaire de son login et mot de passe.



Un fois connecté sur le serveur, il ne reste plus à l'utilisateur de copier du serveur et coller en local les répertoire ou fichier qu'il veut restaurer.
Il est vraiment difficile de faire moins compliquer.

# 8 Installation et utilisation de `rsync` sous Linux/Unix

Sous Unix/Linux, la commande `rsync` est disponible. Si elle n'est pas installée par défaut, installez le paquetage correspondant. Pour notre debian un simple `apt-get install rsync` suffit en tant que `root`.
Ensuite comme sous windows, il faut générer le jeu de clé publique/privée pour l'utilisateur. La syntaxe de commande est la même :

```
ssh-keygen -t dsa
```

Cette commande génère les clés de l'utilisateur et les place dans le répertoire `$HOME/.ssh`.
Il faut ensuite copier la clé publique sur le serveur.

```
scp ~\.ssh\id_dsa.pub morere@1974.57.140.212:.ssh/authorized_keys
```

Un test de connexion est réalisé par la commande

```
ssh morere@194.57.140.212
```

Il suffit ensuite d'utiliser la commande `rsync` dans un script shell de la manière suivante (fichier `sauvegarde.sh`) :

```
#!/bin/bash
rsync -av --exclude-from=$HOME/sauvegarde/exclude.txt --rsh="ssh -l morere"
$HOME 194.57.140.212:linux
```

avec son fichier `exclude`

```
temp
GNUstep
```

Ensuite on insert une ligne dans la `crontab` par `crontab -u user -e`

```
0 2 * * * /home/user/sauvegarde/sauvegarde.sh | mail -s Sauvegarde morere@sciences.univ-metz.fr
```

et voila le tour est joué, on sauvegarde le répertoire `home` sur le serveur dans le répertoire `linux` tous les jours à 2h00.

# 9 Sauvegarde des données du serveur *Mirroring*

Afin de limiter au maximum les coûts, la lourde tâche de sauvegarde sera assurée par deux disques durs. Le but est alors de sauver alternativement le contenu du répertoire `/home` du troisième disque sur l'un des deux disques de sauvegarde.
Pour cela on utilise 3 disques de 120Go pour le stockage et un disque de 3.2Go pour le système (Une debian Sarge (testing)).

```
morere@kstore:~$ df
Filesystem           1K-blocks      Used Available Use% Mounted on
/dev/hda2              2845540   1529672   1171324  57% /
tmpfs                   128332         0    128332   0% /dev/shm
/dev/hde1            118176996   1796260 110377696   2% /home
/dev/hdg1            118176996   1794960 110378996   2% /sauvegarde1
/dev/hdh1            118176996     32828 112141128   1% /sauvegarde2
morere@kstore:~$
```

Un des disques de 120Go contient les répertoires `home` des utilisateurs et les deux autres servent à faire les sauvegardes synchronisées un jour sur deux (ou moins suivant le choix que l'on fait).
Dans mon cas, tous les jours impairs, les répertoires `home` sont synchronisés sur le disque `/sauvegarde1`, et tous les jours pairs sur le disque `/sauvegarde2`.
Ceci est réalisé par une simple commande insérée dans le `cron` par `crontab -e` en tant que `root`.

```
0 0 * * 0,2,4,6 rsync -av --delete --exclude-from=/root/exclude /home/ /sauvegarde1/ | mail -s "Sauv
0 0 * * 1,3,5 rsync -av --delete --exclude-from=/root/exclude /home/ /sauvegarde2/ | mail -s "Sauveg
```

```
kstore:~# more exlude
/home/webCDcreator/
/home/lost+found/
/home/ftp/
kstore:~#
```

Voila, pour l'instant le serveur est opérationnel pour notre laboratoire. La notion de quotas n'a pas été prise en compte. En effet les disques sont pour l'instant surdimensionnés pour nos besoins.
De même la suite du projet concerne l'écriture d'un programme C/GTK+ pour la configuration simplifiée et la création automatique des fichiers `sauv.bat` , `exclude.txt` etc.
Les questions et commentaires sont les bienvenus à l'adresse suivante : `morere@lasc.univ-metz.fr`