Rapport de stage :
Solution de sauvegarde par le réseau (BACULA)

Yann BELEMA
Réseaux et Télécommunications 2e année

A l'attention de :

- Mme Danielle CABALLERO, tutrice pédagogique
- M. James REGIS, maître de stage

2010/2011

Rapport de stage :
Installation et déploiement d'une solution de sauvegarde par le réseau (BACULA)

Yann BELEMA
Réseaux et Télécommunications 2e année

A l'attention de :

- Mme Danielle CABALLERO, tutrice pédagogique
- M. James REGIS, maître de stage

2010/2011

Remerciements

Je tiens à remercier mon maître de stage M. James REGIS, ingénieur en Systèmes Réseaux, qui m'a permis de réaliser mon stage dans de bonnes conditions de travail en m'accordant sa confiance et en me consacrant le temps nécessaire pour répondre à mes questions ou m'aider à pallier à certaines difficultés rencontrées.

Je remercie ma tutrice pédagogique et professeur de culture et comunication, Mme Danielle CABALLERO qui s'est assurée du bon déroulement de mon stage et qui n'a cessé, tout au long de l'année, de nous prodiguer des conseils concernant la rédaction de ce rapport et la préparation à la soutenance de stage.

Enfin je remercie toute l'équipe pédagogique de l'IUT justement du soutien scolaire dont elle nous a fait profiter et je remercie particulièrement Mme Chantal LABAT, professeur de réseaux, pour la dynamicité dans l'organisation des stages.

Table des matières

Résumé

Étudiant en seconde année de DUT Réseaux et Télécommunication, j'effectue un stage en entreprise au sein du laboratoire informatique LIX, à l'école polytechnique de Palaiseau. Je travaille dans l'équipe SYSRES (systèmes et Réseaux) dont le principal rôle est d'assurer la maintenance des moyens informatiques du LIX. Aussi est-il important de sauvegarder en permanence et de stocker les données des ordinateurs qui constituent le réseau du laboratoire ; tout en ayant la possibilité de pouvoir les restaurer à tout moment.

Il existe plusieurs méthodes de faire des sauvegardes comme *AMANDA*, une solution de sauvegarde permettant à un administrateur de configurer un serveur unique pour la sauvegarde de plusieurs hôtes de systèmes hétérogènes sur un réseau de lecteurs de bandes, de disques ou supports optiques ; *LaCie*, un logiciel de sauvegarde pour archiver des fichiers sur un disque dur externe ; la solution de sauvegarde *DUMP* pour la sauvegarde du système de fichiers des systèmes UNIX. Ces solutions de sauvegarde vont de celles qui sont adaptées aux particuliers à celles qui sont adaptées aux entreprises.
Le système de sauvegarde existant du LIX est assuré par la DSI au moyen du logiciel *Tina Time Navigator*, une solution de sauvegarde centralisée. Mais le but est que le laboratoire LIX dispose de son propre espace de stockage, géré par une solution disposant au minimum des mêmes capacités ou plus que la solution actuelle, au moindre coût.

BACULA est une suite logicielle consistant à sauvegarder, de restaurer et de vérifier sur les bandes de stockage, en réseau, les données des serveurs et des utilisateurs du LIX. Tout cela se fait en trois grandes étapes : l'installation du projet Bacula, sa configuration ainsi que son déploiement.

La réalisation du projet nécessite de la recherche et l'analyse de la documentation technique de Bacula et tous les éléments dont dépend son fonctionnement ; elle sollicite aussi des connaissances soutenues en administration systèmes et réseaux. De plus ce projet se différencie des autres de part ses nombreuses fonctionnalités et sa flexibilité, et il correspond parfaitement aux attentes des utilisateurs du LIX concernant une sauvegarde fiable.

Les principaux thèmes abordés dans ce projet seront : la sauvegarde, le réseau, la notion du logiciel libre, la licence GNU/GPL, le principe du client/serveur, les disques durs, les bandes de stockage, LT0, les auto-changeurs ainsi que les systèmes d'exploitation UNIX/Linux et Windows.

Abstract

As student in the two-year course in Networks and Telecommunications I did my internship in the computer science lab LIX based in Palaiseau. I worked in the team of Systems Engineers networks whose main role is to ensure the maintenance of the lab LIX network. It is therefore important to permanently save and store data on computers that form the network of laboratory while having the possibility to restore them at any time.

This is the main objective of the subject of the course. Indeed, the Bacula project permits the system administrator to manage backup, recovery and verification of computer data across a network. All this is done in three phases : the installation of Bacula project, its configuration and deployment.

The project requires research and analysis of technical documentation for Bacula and all the factors affecting its operation, it needs also knowledge in systems and networks. Moreover, this project differs from others by its many features and flexibility, and it corresponds perfectly to the needs of users of LIX on a reliable backup.

This work was a rewarding experience because that one is required to communicate frequently in order to develop relationship and to improve myself professionally, so I did a first step into the world of work.

The main topics covered in this project are : the backing up, networks, open source, the GNU / GPL, the concept of client / server, disk drives, storage tapes, LT0, autochangers, UNIX / Linux and Windows.

Glossaire

Auto-changeur : Un périphérique automate qui inclut un robot, un lecteur de bandes et un ou plusieurs magasins pour les cartouches de bandes. Il permet le chargement ou le déchargement automatique des bandes de stockage.

btape : est un programme du projet Bacula qui permet de déterminer la configuration de lecteur de bande.

Catalogue : Service utilisé pour stocker des informations sommaires concernant les jobs et clients (hôtes), les fichiers qui ont été sauvegardés ainsi que le ou les volume(s) où ils ont été sauvegardés.

Daemon : Programme informatique qui s'exécute en arrière-plan, plutôt que sous le contrôle direct d'un utilisateur. Ce programme est connu sous le nom de service sous *Windows*.

iscsi : Internet Small Computer System Interface en anglais, est un standard définissant un bus informatique permettant de relier un ordinateur à des périphériques ou bien même à un autre ordinateur. Ce bus dispose d'une interface plus rapide et plus universelle.

Job : ou travail, est le processus exécuté dans le but de réaliser une sauvegarde ou restaurer des données.

Package :(Red Hat Package manager ou rpm) est un système de gestion de paquets de logiciels utilisé sur certaines distributions GNU/Linux.

Ressource : Une ressource est une partie d'un fichier de configuration qui définit une unité spécifique d'information disponible pour Bacula. Par exemple, la ressource Job définit toutes les propriétés d'un Job spécifique : nom, schedule (planification), volume pool, type de sauvegarde, niveau de sauvegarde, etc.

Table des figures

Introduction

En fin de deuxième de DUT, il est important, pour un étudiant d'effectuer un stage en entreprise, afin de valider le diplôme, mettre en pratique les connaissances théoriques acquises au cours de la formation de l'IUT et de se familiariser avec le monde de l'entreprise, et donc du travail.

Je réalise mon stage de fin d'étude au laboratoire informatique de l'école polytechnique (LIX) à Palaiseau. Les activités du LIX se regroupent en trois grands domaines : algorithmique, réseaux, et méthodes formelles. J'ai travaillé dans l'équipe d'ingénieurs en Systèmes Réseaux dont les principales fonctions sont d'assurer la maintenance du parc informatique du LIX, d'assister les utilisateurs, de créer les scripts utilisateur (python, perl), d'administrer les services réseaux, d'en assurer leur sécurité et de faire évoluer le réseau pour un meilleur fonctionnement. Mon maître de stage, M. James REGIS, est ingénieur expert des Systèmes et Réseaux.

Durant ce stage, j'ai travaillé sur le projet Bacula, une suite logicielle qui permet de mettre en place un système de sauvegarde, de restauration et de vérification de données en réseau.

Ce projet a été réalisé car en plus du système de sauvegarde dont dispose le laboratoire (Time Navigator), on a besoin d'un autre système qui propose une sauvegarde de manière fiable, organisée et sécurisée et qui dispose bien de plus de fonctionnalités comme une interface graphique qui permettrait de suivre dans la transparence la sauvegarde.

Pour le développement du rapport, je présenterai dans un premier temps de **l'entreprise d'accueil**, ses différentes activités, mon service d'accueil ainsi que le sujet du stage. Ensuite, je poursuivrai par la présentation du **projet BACULA**, depuis son utilisation, jusqu'à l'objet d'un test de sauvagarde. Juste après cette partie, je traiterai de la **mise en œuvre de Bacula** qui consiste à son déploiement dans l'environnement concret du laboratoire, avant de terminer sur quelques **fonctionnalités, avantages et inconvénients** de BACULA.
Avis au lecteur :

* Les termes en rouge sont répertoriés dans le glossaire.
* Les phrases suivies de chiffre entre [ ] font référence à la bibliographie (sites web, magazines, annexes...)

∗ Les chiffres en exposant indiquent les notes de bas de page.

1 L'entreprise

1.1 Présentation

Situé au cœur du Centre de Recherche sur le campus de l' École Polytechnique à Palaiseau, le Laboratoire d' Informatique de l' X (LIX) est une UMR X-CNRS d' une centaine de membres dont une moitié de doctorants et plus d'une quarantaine de permanents équitablement répartis entre le CNRS, l'INRIA et l'X.

Parmi les onze équipes présentes dans les locaux du laboratoire, on compte 7 projets INRIA installés au LIX depuis la création du Pôle Commun de Recherche en Informatique du Plateau de Saclay, et une équipe associée avec le CEA LIST qui préfigure les cohabitations futures entre des équipes d'origine différente issues du RTRA Digiteo.

Enfin, le LIX abrite la Chaire «Systèmes Industriels Complexes» financée par Thalès, et la Chaire « Optimisation pour le Développement Durable» financée par Microsoft Research. À tous ces titres, il est devenu un acteur académique essentiel au sein du pôle de compétitivité System@tic.

Les activités du laboratoire s'organisent sur une dizaine d'équipes de recherche :

Algorithme et Optimisation : Cristophe Dürr
BioInformatique : Mireille Regnier
Comete : Catuscia Palamidessi
Cryptologie : Daniel Augot
Hipercom : Thomas Clausen
Typical : Benjamin Werner
Max : Marc Giusti
Modeles Combinatoires : Gilles Schaeffer
Parsifal : Dale Miller
Sysmo : Léo Liberti

L'organigramme du LIX est le suivant :

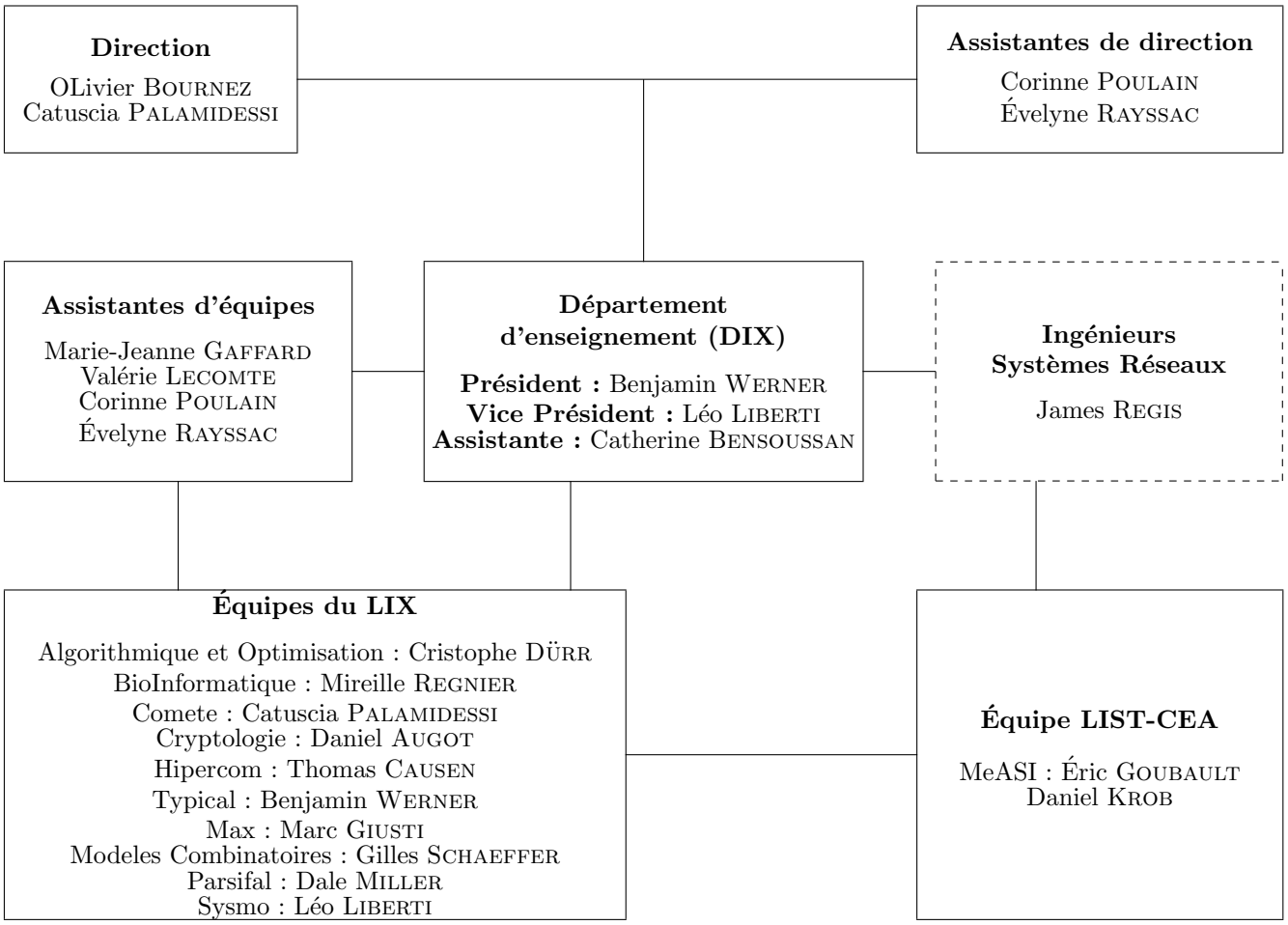


Figure 1.1 – Organigramme du LIX

1.2 Activités du LIX

Les activités du LIX se regroupent en trois grands domaines : algorithmique, réseaux, et méthodes formelles.

Le LIX s'intéresse tout particulièrement au domaine des communications en réseau, afin de se doter dans ce domaine de compétences globales relatives au traitement du signal, au chiffrement et au routage des communications, à la distribution et à la mobilité des calculs ainsi qu'à la sécurité et à l'ingénierie des protocoles.

Au cœur de ces recherches, dont le but est de mettre au point des systèmes de communications fiables et efficaces, l'accent est mis sur les réseaux mobiles, qui deviendront une composante incontournable des systèmes embarqués futurs.

1.3 Partenariats

Le LIX collabore avec le centre de recherche INRIA Saclay île de France, avec lequel il partage 7 équipes-projets, hébergées au LIX : Alien, Amib, Comète, Hipercom, Parsifal, Tanc, et TypiCal.

Le LIX est associé au laboratoire LIST du CEA, à travers l'équipe MeASI.

Le LIX est également membre du réseau Digiteo.

À travers le réseau System@TIC, le LIX collabore avec Thales Research and Technology.

Le laboratoire est fortement impliqué dans les enseignements de l'École Polytechnique, et aussi dans plusieurs MASTER de la région parisienne. Il a des relations contractuelles avec des organismes publics (Ministère de l'Enseignement Supérieur et de la Recherche, Direction Générale de l'Armement, Agence Française pour l'Innovation, **INRIA**...) ou des organisations internationales.

Les principaux organinismes travaillant en partenariat avec les équipes du LIX sont :

- **CNRS**
- **INRIA-FUTURS :** Alien, Comète, Hipercom, Logical, Tanc,Parsifal
- **CEA-LIST :** Equioe de recherche communeMeASI
- **Université de Paris Sud :** LRI, LIMSI, IEF
- **SUPELEC :** LSS, Département Signaux Systèmes Electroniques
- **ENS Cachan :** LSV
- **Thales Research and Technology**

Au cours des dernières années, le LIX s'est attaché à développer de nombreuses collaborations

avec le monde industriel. Outre Thalès, on peut citer parmi nos grands partenaires internationaux Microsoft Research, Hitachi Labs, et la NASA.

1.4 Le service d'accueil

Le laboratoire nécessite une perpétuelle maintenance et optimisation de son réseau informatique de façon à ne pas entraver les projets menés par les différentes équipes. L'équipe d'ingénieurs Systèmes Réseaux est composée pour le moment de M. James REGIS.

L'équipe assure la maintenance et la sécurité du parc informatique du LIX, tout en faisant face aux divers problèmes que les membres du laboratoire peuvent être amenés à rencontrer. Le service intervient rapidement en cas de problèmes et c'est pourquoi la grande majorité des opérations est réalisée par le réseau, c'est-à-dire à distance, permettant ainsi de gagner du temps. En parallèle l'équipe recherche continuellement des solutions en vue d'optimiser et de faire évoluer le réseau du laboratoire.

De par ses responsabilités la service entretient d'étroites relations avec les autres équipes et prévient les membres du laboratoire si besoin est. Le service est chargé de l'implémentation, du remplacement ou de la suppression des équipements (imprimantes, câbles, serveurs, disques durs...) constituant le réseau. En règle générale, les postes de travail et les serveurs utilisent des distributions Linux telles que CentOS ou Fedora Core.

1.5 Le stage

1.5.1 Sujet du stage

L'objectif à atteindre est d'avoir un système de sauvegarde fiable et organisé qui permet de pouvoir gérer d'une manière centralisée et transparente la sauvegarde, la restauration et la vérification des données des ordinateurs du laboratoire LIX.

Le stage consiste à installer, configurer, tester et déployer une solution de sauvegarde, de restauration et de vérification des données des ordinateurs du réseau au travers d'un logiciel adapté : Bacula.

Le point fort de ce logiciel est que son système dispose des services séparés dans le processus de sauvegarde, ce qui peut augmenter la vitesse d'exécution des tâches.

1.5.2 Outils utilisés

Les stations utilisées pour l'installation du logiciel et les test de sauvegarde et présentées dans ce rapport sont des machines virtuelles disposant du système d'exploitation CentOS 5.5 stable et du noyau UNIX/Linux 2.6.18-194.32.1.el5 sur un système x86_64 bits.

L'ensemble des opérations ont été effectuées à distance à l'aide du protocole SSH qui permet d'établir un canal de connexion sécurisée avec les stations grâce à un algorithme de cryptage ; ce canal permet aussi des transfert de données.

Aussi j'ai appris les bases du langage LaTeX, formateur de texte, qui me sert pour rédiger mon rapport de stage.

2 BACULA

2.1 Présentation

« Il arrive la nuit et absorbe l'essence vitale de vos ordinateurs »[1] tel est les slogan du projet Bacula, contraction de backup et Dracula.

Bacula est une solution de sauvegarde par le réseau sur un modèle client/serveur , c'est un logiciel libre sous licence GNU version 2. Il est développé depuis 2000 et dispose d'un grand nombre de fonctions parmi lesquelles le multi-volume et le multi-support.

C'est un ensemble de programmes qui vous permet de gérer vos sauvegardes, restaurations ou vérifications de données d'un ordinateur sur un réseau hétérogène. Il fonctionne sur tout système respectant le standard POSIX et avec un client spécifique sur *Windows* qui utilise le volume *Shadow Snapshot* (VSS) propre à ce système ; il utilise une base de données relationnelle propulsée par MySQL, PostgreSQL, SQLite, crée des signatures SHA-1 (ou MD5) [1] pour chaque fichier sauvegardé, permet le chiffrement des données et des commandes transitant sur le réseau et peut sauvegarder des fichiers de grands volumes. Bacula peut fonctionner complètement sur un seul ordinateur, et il est capable de sauvegarder sur des supports variés, y compris disques et cartouches.

Il met en oeuvre un nombre assez important de démons affectés à des tâches bien précises et des mécanismes de gestion à adopter qui n'en facilitent pas toujours une approche rapide ; tout en offrant de nombreuses fonctions avancées de gestion de stockage qui facilitent la recherche et la restauration de fichiers perdus ou endommagés.

Selon les statistiques de Source Forge [2] (rank et downloads), Bacula est de loin le plus populaire des logiciels Open Source de qualité entreprise.

1. L'algorithme MD5 (tout comme SHA-1) est une fonction dit de hachage cryptographique qui permet d'obtenir l'empreinte numérique d'un fichier (on parle souvent de message). Il a été inventé par Ronald Rivest en 1991.
2. SourceForge.net, la plate-forme d'hébergement de projet

Le tableau suivant donne des informations sur la version utilisée du logiciel :

Nom du logiciel	Bacula®
Type de logiciel	Sauvegarde
Licence	Libre GNU Public Licence
Environnements	Linux, Solaris, FreeBSD, MacOS, Windows
Version utilisée	2.0.3 (06 Mars 2007)
Site web	`www.bacula.org`

FIGURE 2.1 – Tableau d'informations-Bacula

2.2 Architecture du réseau de test

Dans un premier temps je dispose d'un réseau local composé de deux machines virtuelles installées sur un serveur. Ces machines disposent du système d'exploitation CentOS 5.5 stable et du noyau UNIX/Linux 2.6.18-194.32.1.el5 sur un système x86_64, et sont accessibles et administrables à distance via le protocole SSH. [3]

L'une des deux machines virtuelles va servir de serveur sur lequel on installe le logiciel Bacula et tous ses services et l'autre machine nous sert de client afin de vérifier la sauvegarde à distance, par le réseau. Sur les flèches numérotées de la figure suivante :

– 1re : Connexion du directeur au client ; la connexion du directeur au service *SD : Storage Daemon* (Stockage) se fait en local sur la machine qui héberge le service directeur.
– 2e : Connexion entre client et service de stockage, le sens de connexion dépend du *job* lancé : sauvegarde ou restauration.
– 3e : Connexion du service de stockage aux périphériques de stockage.

Ainsi lorsqu'on lance un *job*, le directeur indique au client les fichiers à sauvegarder et se connecte au service de stockage pour lui indiquer les périphériques de stockage définis, le client se connecte ensuite au service de stockage pour fournir les données à écrire sur les périphériques de stockage, enfin le service de stockage écrit les données sur les périphériques de stockage selon les paramètres spécifiés par le directeur.

3. Protocole permettant d'établir un canal de connexion sécurisée avec les stations grâce a un algorithme de cryptage.

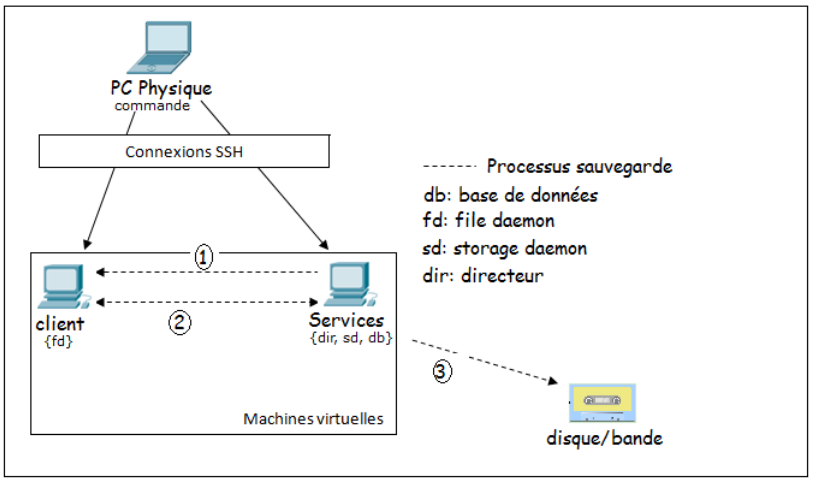


FIGURE 2.2 – Architecture de test pour la sauvegarde.

La connexion sécurisée via SSH entre ses machnines nécessite un minimum de configuration, et sur les machines virtuelles et sur mon compte utilisateur disponible sur une machine physique et accessible par le réseau.

```
## Sur mon compte utilisateur [belema@sixfeet~]$
$ssh-keygen -t rsa                                          #Générer un couple de clés publique
                                                            #/privée (cryptographie asymétrique)
$scp ~/.ssh/id_rsa.pub                                      #copier la publique dans le fichier
                                                            #id_rsa.pub dans le répertoire /home
$ssh-copy-id -i ~/.ssh/id_rsa.pub root@machine.virtuelle    #envoyer la clé publique aux machines
                                                            #virtuelles d'adresses IP:
                                                            #192.168.112.54 (root@192.168.112.54)
                                                            #et 192.168.112.53 (root@192.168.112.
                                                            #53)
$ssh root@machine.virtuelle                                 #On peut maintenant se connecter sur
                                                            #les machines virtuelles
Enter passphrase for key '/users/bofh/belema/.ssh/id_rsa':<mot de passe du fichier qui contient
notre clé privée permettant d'ouvrir la connexion>

## On configure sur les deux machines virtuelles leur adresse IP et nom afin d'éviter de taper
"root@adresse.IP.de.la.machine"

# Ouvrir le fichier /etc/hosts
192.168.112.54 bacula                                       #Adresse IP de la machine serveur
                                                            #et son nom
192.168.112.53 client                                       #Adresse IP de la machine client
                                                            #et son nom
```

```
# Ouvrir le fichier /etc/sysconfig/network
HOSTNAME: <nom de la machine virtuelle>

## Ensuite sur mon compte [sixfeet], créer le fichier ~/.ssh/config pour configurer l'accès aux
machines virtuelles par leur nom.
$vim ~/.ssh/config                                          #On crée le fichier
HOST *
ForwardAgent yes
ForwardX11   yes
HOST=<nom de la machine virtuelle>
Hostname=<Adresse IP de la machine virtuelle>
User=root                                                   #Utilisateur
Compression=yes
##Redémarrer la machine
## Pour les prochaines connexions, pour éviter de retaper à chaque fois le mot de passe:
$ssh-add                                                    #A l'ouverture de session
Enter passphrase for key '/users/bofh/belema/.ssh/id_rsa': <une fois pour toute connection ssh
établie auparavant>


#Ainsi pour établir une connexion SSH:
$ssh bacula (ou client)
```

Ainsi toutes les opérations à venir vont s'exécuter sur nos deux machines virtuelles, avant le déploiement du logiciel Bacula à toute l'étendue du réseau du LIX.

2.3 Les services Bacula

Le projet Bacula se base sur le Principe client/serveur. On dispose d'une machine-'serveur' sur laquelle les services de Bacula sont installés. Les tâches du logiciel sont séparées et sont spécifiques à chaque service donné dans le processus de sauvegarde. A l'étape du test du logiciel, Bacula est installé, tous ses services, ainsi que ses dépendances sur une seule machine qui est la machine virtuelle serveur, mais une tâche correspondant à un démon [4], et toutes les communications s'effectuant sur le réseau en TCP/IP, il est facile de répartir les tâches sur des machines distinctes ; les performances d'exécution des processus de sauvegarde sont de cette manière optimisées.

4. Programme informatique qui s'exécute en arrière-plan, plutôt que sous le contrôle direct d'un utilisateur.

La figure suivante présente l'architecture d'une installation de Bacula simplifiée à un seul client, et montre aussi les interactions des services entre eux :

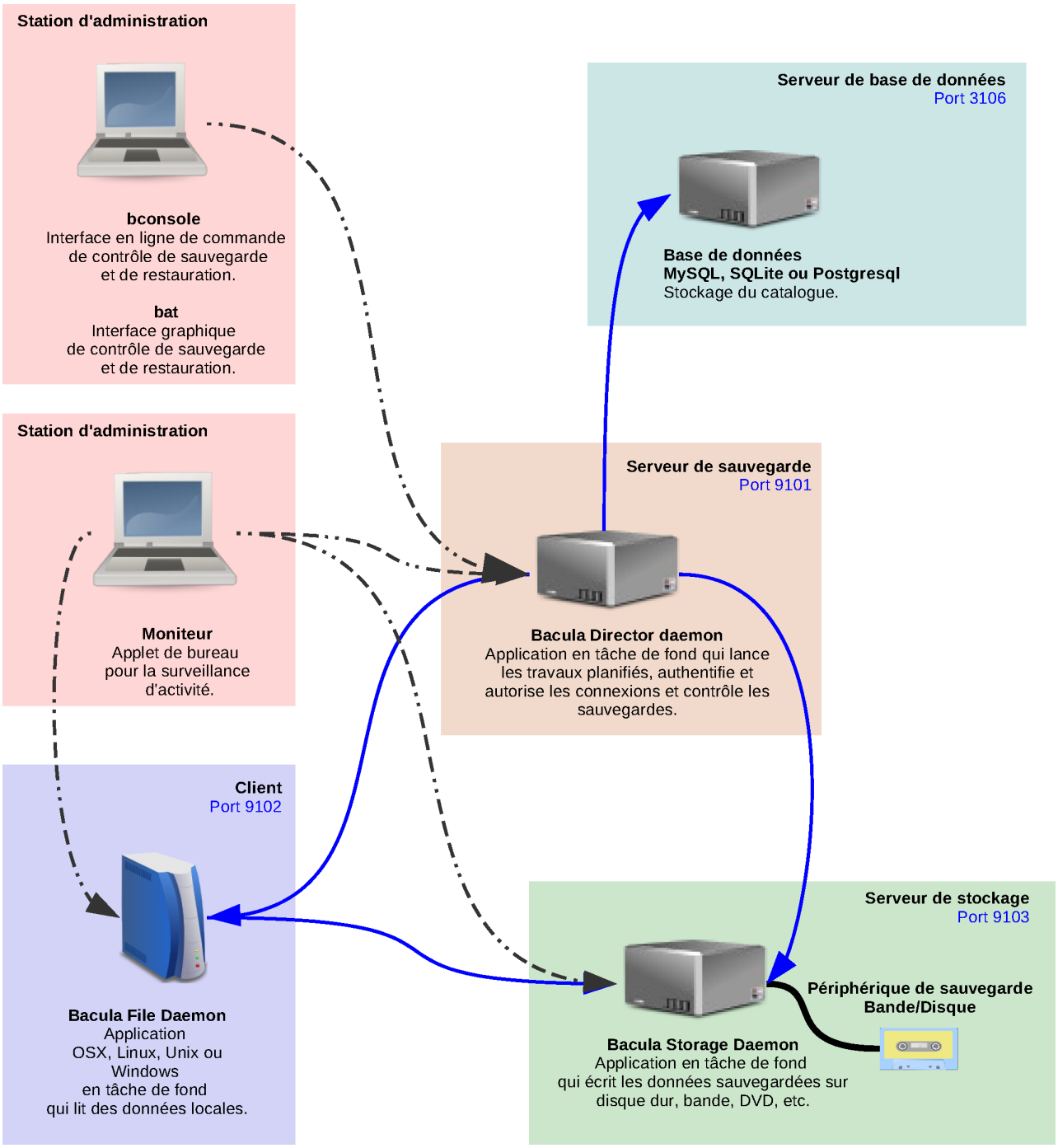


FIGURE 2.3 – Intéractions des services dans Bacula

Comme on peut le constater sur la Figure2.3 ci-dessus, dans une architecture normale de Bacula, il y'a trois services, un serveur de base de données, une interface de commande de contrôle de

sauvegarde et de restauration, et un ou plusieurs moniteurs pour surveiller, depuis un bureau Gnome ou KDE [5], l'activité de services souhaités.

Les services Bacula :

- Serveur de sauvegarde (Bacula Director daemon) ;
- Serveur de stockage (Bacula Storage daemon) ;
- Client (Bacula File daemon).

Le serveur de base de données :

- MySQL ;
- PostgreSQL ;
- ou SQLite.

Interface de commande :

- bconsole (en ligne de commande) ;
- wxconsole ;
- bat (interface graphique) ;
- bweb (dans un navigateur web).

L'élément principal du système est le **directeur** (Bacula-dir). Il est unique ; on l'utilise pour planifier les sauvegardes et restaurer les fichiers. Il est exécuté en tache de fond c'est a dire en tant que daemon ou service. Il contacte les clients, File Daemon (FD), pour qu'ils envoient leurs données sur les volumes du Pool [6] dirigé par le service chargé des supports de sauvegarde, le Storage Daemon(SD). Toutes les informations liées aux différents événements de sauvegarde sont enregistrées, par le directeur, dans le catalogue, stocké dans la base des données.

Le service **Bacula Console** est le programme qui permet à l'administrateur ou à l'utilisateur de communiquer avec le Bacula Director (voir ci-dessus). Actuellement, le service Bacula Console est disponible en trois versions. La première et la plus simple est d'exécuter le programme Console dans une fenêtre shell (ligne de commande). La plupart des administrateurs système trouveront cette méthode parfaitement adéquate ; c'est celle que j'utilise pour interagir avec le directeur. La seconde version est une interface graphique GNOME (graphique) qui est loin d'être complète, mais est tout à fait fonctionnelle puisqu'elle possède la plupart des possibilités de la Console shell ; la version graphique qui est actuellement la plus développée est Bat (Bacula

5. Interface graphique des systèmes d'exploitation UNIX/Linux.
6. Groupe de volumes de stockage

Admin Tool), Figure 2.5 suivante. La troisième version est une interface graphique wxWidgets (wxconsole) qui permet de sélectionner interactivement les fichiers à restaurer. Elle intègre la plupart des fonctionnalités de la console shell, permet la complétion automatique avec la touche tabulation, et fournit une aide instantanée relative à la commande que vous êtes en train de taper. Il existe aussi des projets avec pour objectif une interface de type « web ».

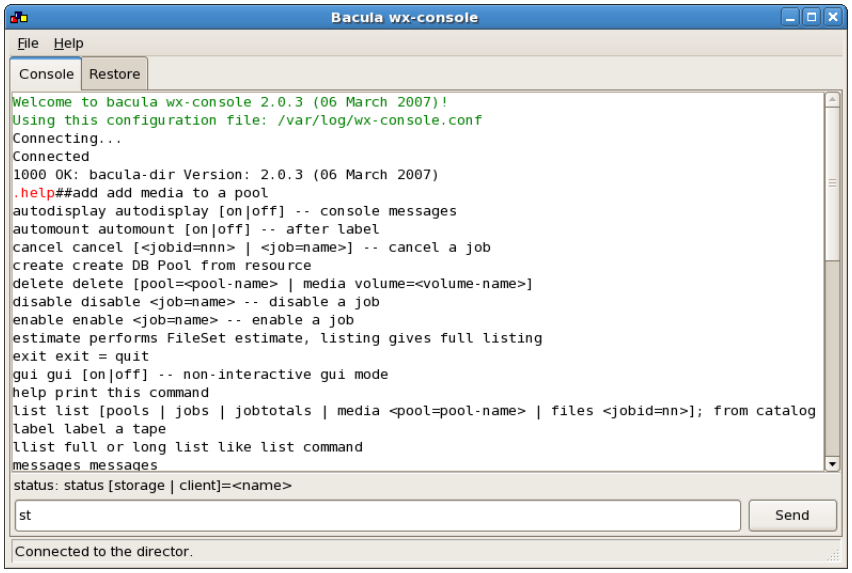


FIGURE 2.4 – Interface wxconsole.

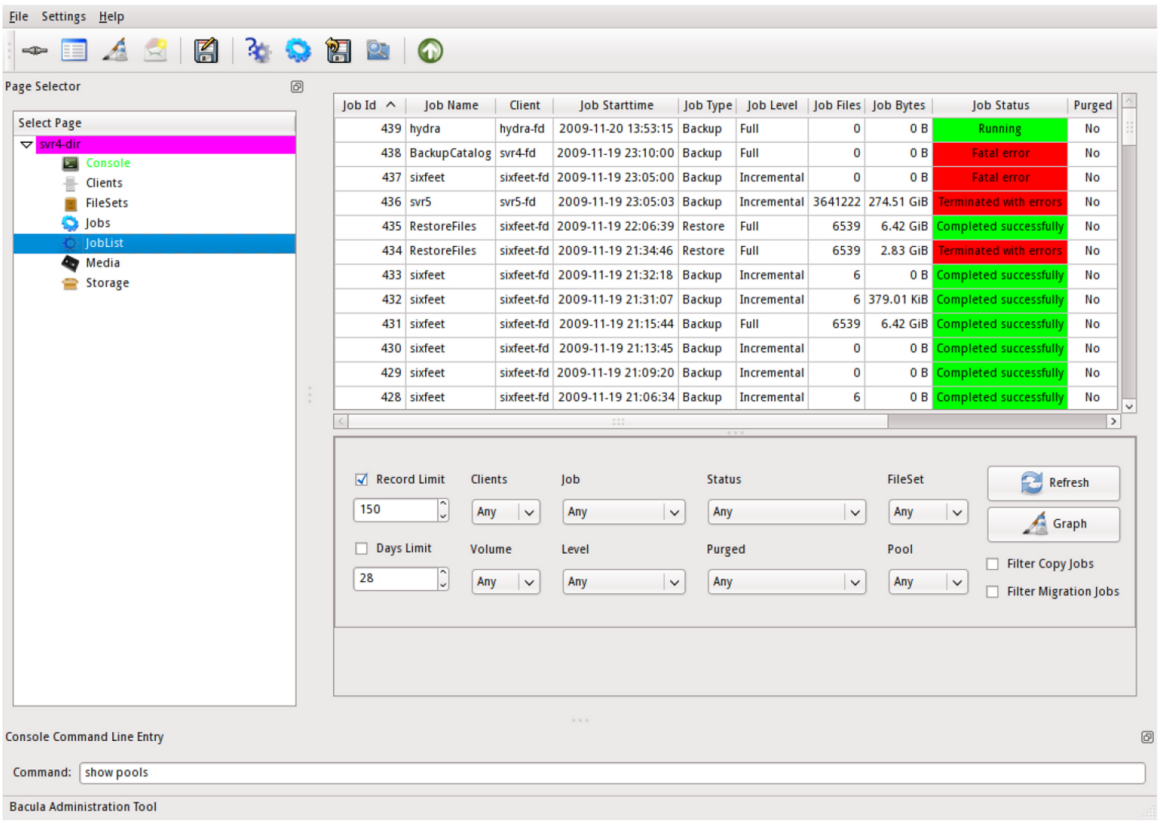


FIGURE 2.5 – Interface Bat.

Le service **Bacula File** (ou programme client, FD) est le programme installé sur la machine à sauvegarder. Il est spécifique au système sur lequel il est exécuté et a la charge de fournir les attributs des fichiers et les données requis par « Bacula Director ». Celui-ci mettra en place, pour l'exécution d'un travail ou (job), une communication entre le client et un Storage Daemon (service de stockage). Les Services File sont aussi chargés de la partie dépendant du système de fichiers lors de la restauration des attributs de fichiers et des données. Ce programme est exécuté en tant que service sur la machine à sauvegarder, et la documentation s'y réfère parfois en tant que Client. En plus du File Daemon pour Unix/Linux, il existe un File Daemon pour Windows (usuellement distribué au format binaire). Le File Daemon Windows fonctionne sur toutes les versions actuelles de Windows (NT, 2000, XP, 2003, Win7, Vista et peut-être aussi 98 et Me).

Le service **Bacula Storage** est le programme qui transfère les données et les attributs de fichiers aux média physiques ou aux volumes et les restitue lors de restaurations. En d'autres termes, le storage Daemon est responsable des opérations de lecture et d'écriture sur le support de sauvegarde (ou autres média de stockage, comme par exemple le système des fichiers). De même, pour l'exécution d'un travail, « Bacula Directeur »établit une communication entre le File Daemon (client) et le Storage Daemon (stockage) afin que le File Daemon fournisse au Storage Daemon les données et les attributs de fichiers requis par Bacula directeur (service directeur).

Les services **Catalogue** ont pour tâche de maintenir à jour la base de données des indexes de fichiers et volumes pour toutes les données sauvegardées. Les services de Catalogue permettent à l'administrateur système ou à l'utilisateur de localiser rapidement et restaurer n'importe quel fichier. Le catalogue Bacula maintient un enregistrement de chaque volume utilisé, chaque job exécuté et chaque fichier sauvegardé ce qui permet des restaurations et une gestion de volumes efficaces. Bacula supporte actuellement trois bases de données différentes, MySQL, PostgreSQL, et SQLite. L'une des trois doit être choisie à la compilation ou l'installation de Bacula.

Enfin, le service **Bacula Monitor** est le programme qui permet à l'administrateur ou à l'utilisateur de voir le statut des daemons Bacula (Bacula Directors, Bacula File Daemons et Bacula Storage Daemons) en mode graphique (voir ci-dessus). Actuellement, la seule version disponible est une version GTK+, qui fonctionne avec Gnome et KDE (ainsi que tout gestionnaire de fenêtre qui respecte le standard system tray FreeDesktop.org). On en parlera pas ou très peu dans le rapport.

Il est donc indispensable, pour réaliser avec succès les opérations de sauvegarde et de restauration, de configurer et de lancer les quatre services vus ci-dessus : le Director Daemon, le File Daemon, le Storage Daemon et MySQL, PostgreSQL ou SQLite.
Dans chacun des cas, l'ensemble des paramètres formant le travail est à la discrétion du direc-

teur. Pour plus de détails, consulter les fichiers de configuration dans les annexes.

2.4 Installation

2.4.1 Sources et Systèmes d'exploitation

Les sources et les fichiers binaires du projet sont hébergés sur la plateforme Sourceforge.

On peut procéder de différentes manières pour installer Bacula.La première, étant l'installation depuis les fichiers sources au format *tar.gz*, est plutôt triviale. Le projet base sa construction sur les outils connus *autoconf* et *automake*. Les options de compilation sont listées avec la commande ***./configure –help*** lancée dans le répertoire racine du projet après sa décompression.

Suivant les options choisies, il faudra plus ou moins de bibliothèques pour satisfaire la compilation du projet ; il faudra juste ne pas oublier de lister toutes les options nécessaires pour bien démarrer avec Bacula.

Pour ces raisons, il est conseillé d'utiliser directement des binaires [7] d'installation disponibles sur le site pour les clients sous Windows.

Concernant la seconde manière d'installation de Bacula, certaines distributions Linux, comme Debian, proposent leurs packages [8] prêts à être installés. Sur Redhat Centos, version sur laquelle nous travaillons, l'utilisation du dépôt EPEL [9] permet d'obtenir une version packagée récente de Bacula ; c'est de cette manière que nous allons procéder pour l'installation du projet, car de cette manière le logiciel va s'installer avec toutes les dépendances nécessaires.

Mais attention, les versions sont souvent vieilles et ne permettent pas de profiter des dernières avancées. La version de Bacula sur Debian Lenny est la 2.4, tandis que sur le dépôt EPEL, la version disponible est la 2.0, qui est encore plus ancienne. Notre version sera la 2.0.3 du 06 mars 2007 ; elle dispose des fonctionnalités nécessaires et suffisantes pour le projet et c'est une version stable.

Ces versions peuvent servir éventuellement à installer les clients pour ces distributions. En effet, le directeur et le démon de stockage en version 3.x acceptent aussi les clients en 2.x.

Des binaires, *dpkg* [10] pour Debian et *RPM* pour Redhat et autres distributions les acceptant, sont disponibles pour installer les programmes dans une version récente.

Voici les systèmes d'exploitation supportés par le projet Bacula :

7. Fichiers compilés pour l'installation du projet
8. Ensemble de programmes et de toutes les fonctionnalités dont il dépend.
9. Ensemble de logiciels packagés de sources fiables et sous licence.
10. Packages.

X : Complètement supporté.
étoiles : Fonctionne, mais non supporté par le projet Bacula.

OS	Version	Director daemon	Client daemon	Storage daemon
GNU/Linux	All	X	X	X
FeeBSD	≥ 5.0	X	X	X
Solaris	≥ 8	X	X	X
OpenSolaris	-	X	X	X
MS Windows 32bit	Win98/Me	-	X	-
	WinNT/2K	*	X	-
	XP	*	X	-
	2008/Vista	*	X	-
MS Windows 64bit	2008/Vista	-	X	-
MacOS X/Darwin	-	-	X	-
OpenBSD	-	*	X	-
NetBSD	-	*	X	-
Irix	-	-	*	-
True64	-	-	*	-
AIX	≥ 4.3	-	*	-
BSDI	-	-	*	-
HPUX	-	-	*	-

FIGURE 2.6 – Systèmes d'exploitation supportés par le projet Bacula

Que vous choisissiez les sources ou les binaires, il vous faudra choisir la base de données relationnelles avec laquelle le directeur stockera et générera son catalogue. SQLite est la solution la plus simple. Même si elle n'est pas conseillée pour la mise en production, elle est bien adaptée à l'évaluation. Dans notre cas, on utilisera le Système de Gestion de Base de Données (SGBD) PostgreSQL qui a de fortes ressemblances avec MySQL.

2.4.2 Installation du serveur Bacula

Comme précisé un peu plus haut, je vais installé la version de Bacula qui se trouve dans le dépôt des fichiers *rpm*. Le logiciel BACULA n'existe pas dans les dépôts de logiciels de *Cent OS 5.5*[11] d'où la nécessité de télécharger sur internet le pack qui contient les fichiers à extensions *.rpm* du logiciel BACULA et ses dépendances sur le site `http://fedoraproject.org/wiki/EPEL`.

EPEL est un groupe spécial des paquets de *Enterprise Linux (EPEL : Extra Packages for*

11. Système d'exploitation

Enterprise Linux)[4]. EPEL se compose d'un package « EPEL-release »qui comprend les clés « *gpg* » pour la signature de paquet et les informations des références. L'installation de ce package pour votre version de *Linux Enterprise* devrait nous permettre d'utiliser des outils tels que « yum »pour installer des paquets et leurs dépendances.

Sur la page Web, dans la liste des packages (*What packages and versions are available in EPEL*), copier le lien du package « *epl-release-5-4.noarch.rpm* » qui est le suivant : `http://download.fedoraproject.org/pub/epel/5/i386/epel-release-5-4.noarch.rpm`[4]

Ensuite aller dans la console terminale de la machine virtuelle qui nous servira de serveur et taper les commandes suivantes :

```
# wget <coller le lien ci-dessus>
# rpm -ivh epel-release-5-4.noarch.rpm
```

Pour vérifier les différents paquets de BACULA :

```
# yum search bacula
```

Voici les paquets importants à installer pour bien démarrer avec BACULA :

```
bacula-client.x86_64 : Bacula backup client
bacula-common.x86_64 : Common Bacula utilities
bacula-console.x86_64 : Bacula management console
bacula-console-gnome.x86_64 : Bacula console for the Gnome desktop environment
bacula-console-wxwidgets.x86_64 : Bacula console using the wx widgets toolkit
bacula-director-common.x86_64 : Common Bacula Director files
bacula-director-postgresql.x86_64 : Bacula Director with PostgresSQL database support
bacula-docs.x86_64 : Bacula documentation
bacula-storage-common.x86_64 : Common Bacula storage daemon files
bacula-storage-mysql.x86_64 : MySQL Bacula storage daemon files
bacula-storage-postgresql.x86_64 : Common Bacula storage daemon files
bacula-traymonitor.x86_64 : Bacula monitor for the Gnome and KDE system tray
```

Installation des paquets :

```
On applique la commande  ' yum install ' à tous les paquets ci-dessus, sauf la
 base de données qu'on devrait choisir au préalable.

# yum install bacula-****.x86_64
```

2.5 Configuration

2.5.1 Mise en place d'une base de données

Avant de démarrer et de configurer BACULA il est important de configurer tout d'abord la base de données qui va stocker les informations du catalogue BACULA. Dans notre cas on va utiliser postgreSQL .

PostgreSQL est un système de gestion de bases de données (SGBD) qui a beaucoup de ressemblance avec MySQL, qu'on utilise pour gérer le catalogue BACULA.

Après avoir installé les packages du SGBD postgreSQL (voir la liste ci-dessus), on va démarrer le service postgresql :

```
[root@localhost bacula]# cd /etc/init.d
[root@localhost init.d]# ./postgresql start
Initializing database:                                  [  OK  ]
Starting postgresql service:                            [  OK  ]
```

On se connecte ensuite à l'environnement PostgreSQL sous l'utilisateur par défaut « postgres »afin de créer les rôles ou les utilisateurs des bases de données et les bases de données elles même.

```
On crée tout d'abord les principaux utilisateurs:
[root@localhost bacula]# su -- postgres
bash-3.2$                                  #on est dans l'environnement bash-3.2$
bash-3.2$ createuser root               #on crée l'utilisateur root
Shall the new role be a superuser? (y/n) y
CREATE ROLE
#L'utilisateur postgres a tous les privilèges pour gérer la base de données.
Mais on a créé un autre utilisateur,  root pour permettre à l'utilisateur root
de notre machine d'avoir les mêmes droits que postgres.

On peut maintenant créer la base de données bacula au répertoire
/usr/libexec/bacula

[root@svr0 bacula]# ./create_bacula_database
Creating PostgreSQL database
CREATE DATABASE
ALTER DATABASE
Creation of bacula database succeeded.
Database encoding OK                    # encodage des fichiers SQL_ASCII
On ne pourra pas lire la base de données si l'encodage n'est pas bon
```

```
Pour vérifier la liste des bases de données et les utilisateurs associés:
[root@bacula bacula]# psql -l

        List of databases

   Name    |  Owner   | Encoding

-----------+----------+-----------

 bacula    | root     | SQL_ASCII

 postgres  | postgres | UTF8

 template0 | postgres | UTF8

 template1 | postgres | UTF8

(4 rows)
```

Maintenant qu'on a créé l'utilisateur (root) et la base de données bacula, on peut créer les tables de la base de données :

```
[root@localhost bacula]# ./make_bacula_tables
```

Pour tester l'accès à la base de données, se connecter sous l'utilisateur « root » :

```
[root@localhost ~]# su -- root
-bash-3.2$ psql bacula
Welcome to psql 8.1.23, the PostgreSQL interactive terminal.
Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
       \g or terminate with semicolon to execute query
       \q to quit
bacula=#

On aura besoin de protéger l'accès à la base de données par mot de passe:
On se connecte à l'environnement postgreSQL sous l'utilisateur 'root' puis:

[root@bacula ~]# psql bacula
Welcome to psql 8.1.23, the PostgreSQL interactive terminal.
Type:  \copyright for distribution terms
       \h for help with SQL commands
       \? for help with psql commands
```

```
        \g or terminate with semicolon to execute query
        \q to quit
bacula=# alter user root with password 'tototiti';   #on ajoute un mot de passe banal
ALTER ROLE

Ensuite, on devra ajouter ce mot de passe à deux endroits dans le fichier de configuration
du Director: une fois dans la ressource 'Catalog' à la ligne:

dbname = bacula; user = root; password = "tototiti" DB Address '' ''

La seconde fois, dans la ressource job de 'BackupCatalog' à la ligne:

RunBeforeJob = "/usr/libexec/bacula/make_catalog_backup bacula postgres tototiti "

Ensuite , il faut créer la tables de la base de données et monter les privilèges
pour la manipulation de ces tables dans le répertoire: /usr/libexec/bacula
#./make_bacula_tables
#./grant_bacula_privileges
Enfin dans le répertoire /var/log/bacula/data on configure les paramètres de
connexion à la base de données:
Dans le fichier ./pg_hba.conf:
ajouter la ligne: local bacula root md5
Dans le fichier ./postgresql.conf, dans la rubrique 'connexion and authentication'
listen-address = 'localhost' devient listen-address = '*' ('*' = tout)
Port = 5432
```

Tous les fichiers de configuration de postgreSQL se trouvent dans le répertoire :*/var/lib/pgsql*.

La connexion entre le directeur et la base de données est donnée dans la ressource *catalog* du directeur, dans l'option *DB Address* qui indique l'adresse de la machine qui héberge la base de données.

Maintenant que la base de données est créée et configurée, on peut configurer les services bacula qui sont dans le répertoire */etc/bacula/*. Pour plus d'information, voir les annexes *fichiers de configuration*.

2.5.2 Configuration du directeur

Les principales ressources du fichier de configuration du directeur sont :

- **Director :** permet de définir les informations du directeur dont son nom et son mot de passe et son port de connexion.
- **JobDefs :** permet de définir un travail de sauvegarde avec tous les paramètres dont les noms d'autres ressources définies ci-dessous, dans le directeur.

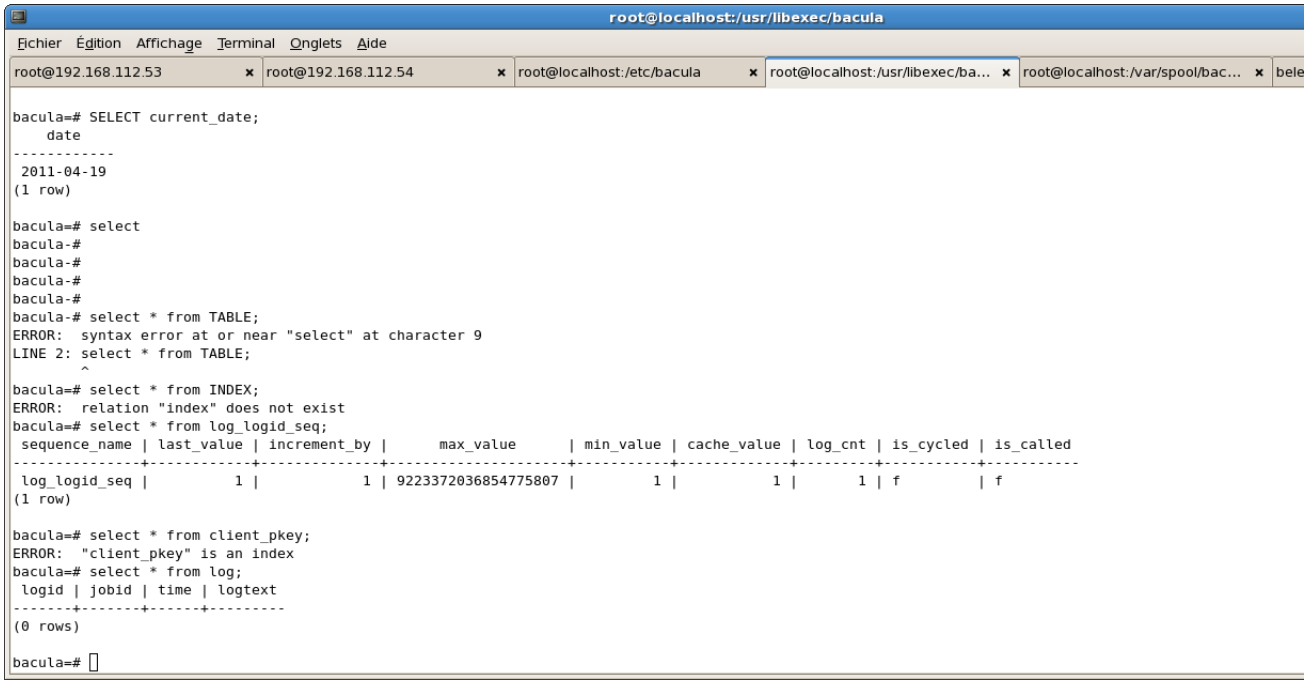


FIGURE 2.7 – Base de données Bacula.

- **Job :**permet de relier un *job* à un client à sauvegarder.
- **FileSet :** Spécifie tous les fichiers à sauver et à exclure lors de la sauvegarde.
- **Client :** permet de définir le client à sauvegarder avec des options telles que son nom, son adresse qui permet au directeur de le contacter, son port de connexion et le catalogue qui va enregistrer les informations de sauvegarde.
- **Storage :** permet de définir les périphériques utilisés pour le stockage ; ils doivent avoir une configuration cohérente du côté du service de stockage. Il faut faire attention à la configuration des machines automates de sauvegarde (voir la partie 3.3 et les annexes *fichiers de configuration*.
- **Schedule :** Spécifie le calendrier et le niveau de sauvegarde.
- **Pool :** Spécifie une série de volumes utilisée par le service de stockage pour stocker les données, et aussi les paramètres de ces volumes dont l'espace maximal de stockage ou encore la période d'utilisation des volumes.
- **Catalog :** Définit dans quelle base de données sera stockée la liste de noms des fichiers et de volumes où ils ont été sauvegardés.
- **Messages :** Bacula dispose d'un serveur mail , le *bsmtp* configuré dans le directeur et qui permet d'envoyer les messages d'erreur et les informations liés aux sauvegardes ; ces messages sont reçus sur compte de messagerie valide (celui de l'administrateur) et dans la console interactive(programme bconsole).
- **Console :** Définit les paramètre de connexion du moniteur aux services.

Comme dans d'autres systèmes de sauvegarde, Bacula dispose de trois niveaux de sauvegarde :

- Full backup :
  Sauvegarde totale, il s'agit de sauver l'ensemble des données des emplacements spécifiés, à sauvegarder.
- Differential backup :
  Sauvegarde différentielle, reprend le principe de la sauvegarde incrémentale sauf que les différences sont obtenues avec la dernière sauvegarde totale effectuée.
- Incremental backup :
  Sauvegarde incrémentale, permet de sauver les données modifiées ou nouvellement créées depuis la dernière sauvegarde, quelque fut son niveau.

2.5.3 Configuration de *Storage daemon*

Le service permet essentiellement de définir les paramètres de connexion au director et de configurer tous les périphériques de stockage que va utiliser Bacula pour stocker les données sauvegarder.
Les principales ressources du fichier de configuration du directeur sont :

- **Storage :** Définit les informations du service de stockage.
- **Director :** Il y a deux ressources *director*, l'une permet de définir les paramètres de connexion du directeur au service de stockage et l'autre ceux de connexion du moniteur.
- **Device :** Spécifie tous les périphériques (dont auto-changeurs) que Bacula utilise pour stocker les données.

Regarder la configuration dans les annexes, *fichiers de configuration.*

2.5.4 Configuration de *File Daemon*

Ce service est très important car, sans lui, le directeur n'arrivera pas à contacter le client à sauvegarder et par conséquent les données ne seront pas sauvées. Il définit essentiellement les paramètres de connexions. Regarder la configuration dans les annexes, *fichiers de configuration.*

Il est important de noter que dans les fichiers de configuration de *bacula-dir, bacula-sd et bacula-fd* il faut absolument faire attention à ne pas laisser les mots de passe respectivement sous la forme suivante :

```
''@@DR-PASSWORD@@''
''@@SD-PASSWORD@@''
```

```
''@@FD-PASSWORD@@''
```

On devra les remplacer par des mots de passe de type "GUGFYTFghjgfugGDRY2738ijkh" pour tous les services.
Dans le cas contraire, on ne pourra pas lancer BACULA automatiquement au démarrage des machines sous le répertoire */etc/init.d/*.

La figure suivante nous montre la configuration des services de *Bacula*.

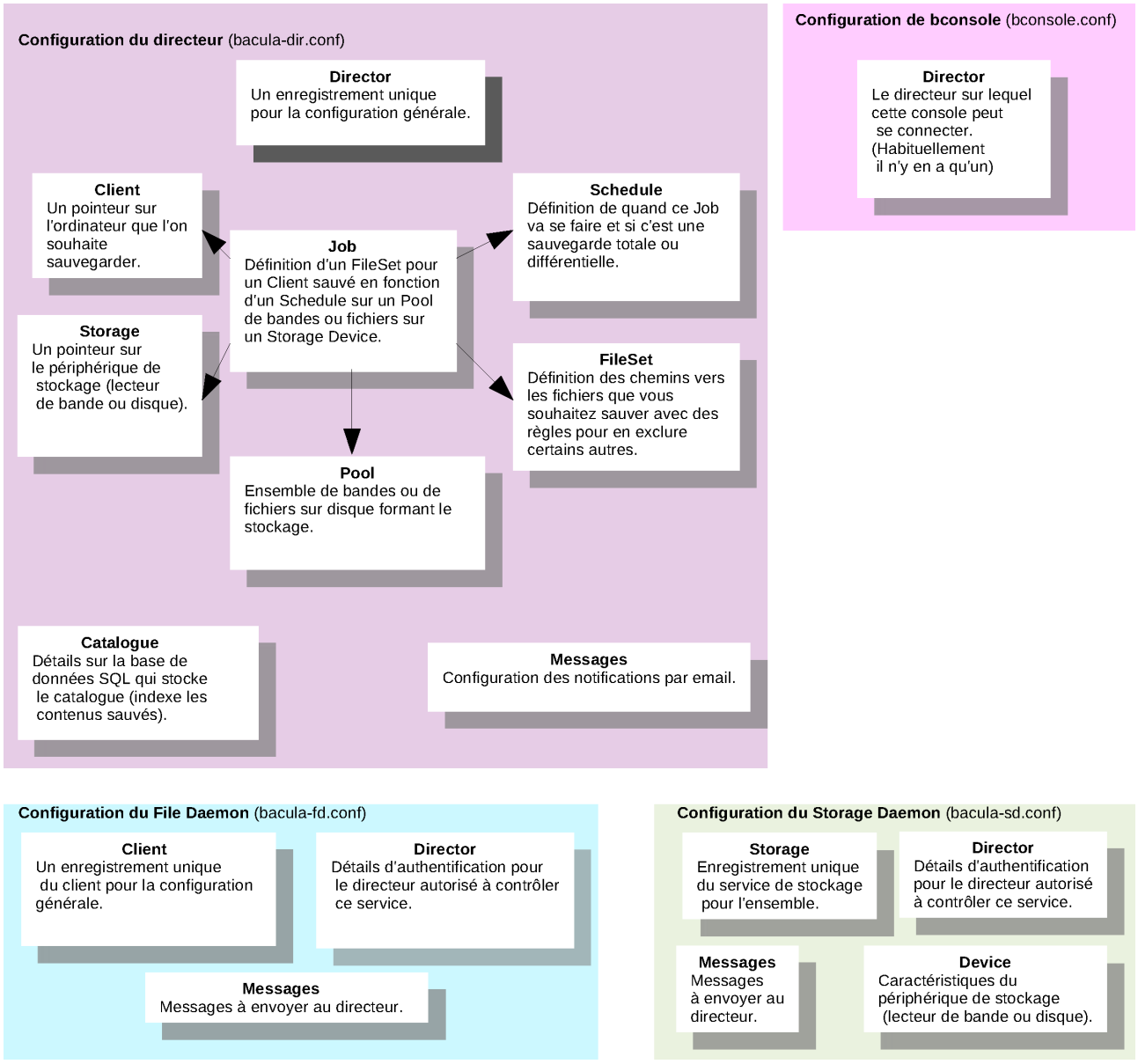


FIGURE 2.8 – Définitions des objets de Bacula

Après avoir vérifier dans les autres fichiers les mots de passe et les adresses on peut démarrer les services :

```
[root@bacula init.d]# ./bacula-dir start  -d100
Starting bacula-dir:                                       [  OK  ]
[root@bacula init.d]# ./bacula-sd start
```

```
Starting bacula-sd:                                        [  OK  ]
[root@bacula init.d]# ./bacula-fd start
Starting bacula-fd:                                        [  OK  ]
```

Il faut régulièrement consulter les fichiers *log* de Bacula afin de voir l'état des démons pour repérer des erreurs dans les fichiers de configuration au répertoire */var/log/bacula*.

2.6 Le programme *bconsole*

Le programme *bconsole* est expliqué dans la partie 2.3 *les services Bacula*. Il permet essentiellement de lancer un *Job*, de nommer des volumes de stockage, de modifier interactivement certaines ressources du directeur liées notamment au *pool, volumes, ou fichiers de restauration* et même d'exécuter des commandes *SQL*. En tapant la commande *help*, on peut avoir la listes de toutes les commandes nécessaires pour commander BACULA, voir la figure ci-dessous.

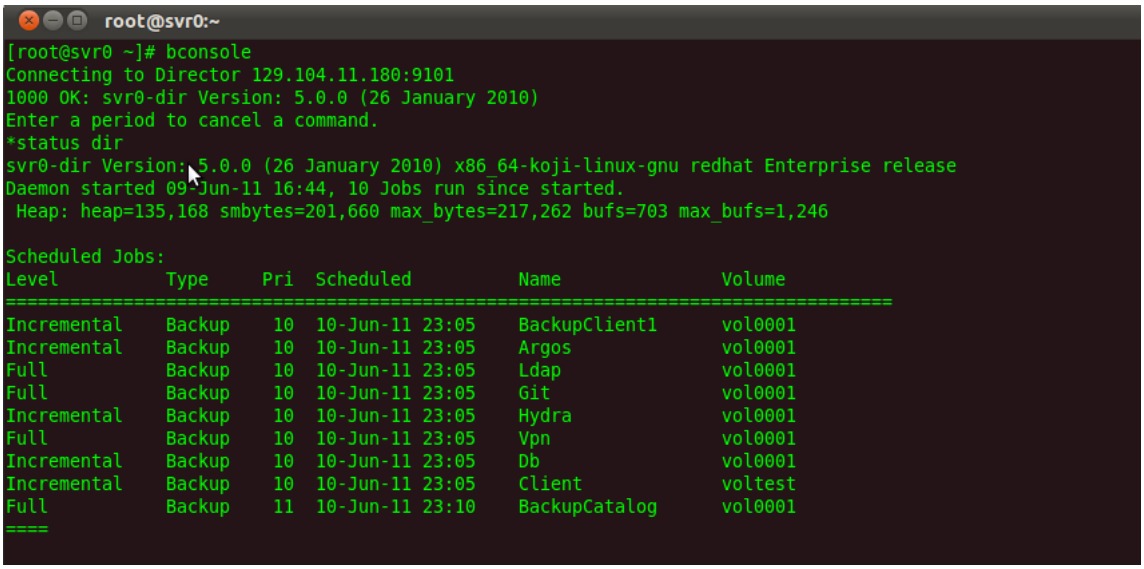


FIGURE 2.9 – connexion via bconsole.

2.7 Premier test de sauvegarde

A chaque sauvegarde partielle, Bacula recherchera la dernière sauvegarde complète qui s'est terminé correctement pour le même *job*. Alors il utilisera la date à laquelle la sauvegarde complète a démarré comme date pour vérifier si les fichiers ont été modifiés.

Bacula refera toujours une sauvegarde complète si aucune sauvegarde complète n'a eu lieu, ou si la dernière sauvegarde s'est terminée avec erreur.

A chaque fois qu'on spécifie dans un *Fileset* de nouveaux fichiers à sauvegarder, Bacula fera une sauvegarde complète. Ce qui est nécessaire pour assurer que tous les fichiers sont proprement sauvés dans le cas où on aurait ajouté beaucoup de fichiers dans le *Fileset*.

Avec le version 1.3.1 de Bacula, les *Fileset* sont datés quand ils sont créés, et cette date est associée au nom du *Fileset* lorsqu'on veut l'afficher ou le sélectionner.

Avant de commencer une première sauvegarde, il faut vérifier l'état de connexion des services (avec la commande $\star status$).

```
[root@bacula sbin]# ./bconsole
Connecting to Director 127.0.0.1:9101
1000 OK: bacula-dir Version: 2.0.3 (06 March 2007)
Enter a period to cancel a command.
*status dir                              #pour vérifier l'état du directeur
et les jobs prévus.

Scheduled Jobs:
Level          Type     Pri  Scheduled          Name               Volume
===================================================================================
Incremental    Backup    10  22-Apr-11 23:05    Client1            *unknown*
Full           Backup    11  22-Apr-11 23:10    BackupCatalog      *unknown*
====

*status storage                     #pour l'état de connexion au bacula-sd  et les
périphériques qui sont montés.

Select daemon type for status (1-4): 2
Automatically selected Storage: File
Connecting to Storage daemon File at bacula:9103

Device status:
Device "FileStorage" (/) is not open.
====

*status Client                        #Pour voir les clients qui sont enregistrés
 dans le directeur et l'état de connexion au bacula-fd
```

```
Select daemon type for status (1-4): 3
Automatically selected Client: bacula-fd
Connecting to Client bacula-fd at bacula:9102

Maintenant on peut démarrer un job avec \star run, vu qu'il n'y a pas erreur de
connexion
*run
A job name must be specified.
The defined Job resources are:
     1: Client1
     2: Client
     3: BackupCatalog
     4: RestoreFiles

You have messages.
*messages
27-Apr 13:38 bacula-dir: No prior Full backup Job record found.
27-Apr 13:38 bacula-dir: No prior or suitable Full backup found in catalog. Doing
FULL backup.
27-Apr 13:38 bacula-dir: Start Backup JobId 5, Job=Client1.2011-04-27_13.38.49
27-Apr 13:38 bacula-sd: Job Client1.2011-04-27_13.38.49 waiting. Cannot find any
appendable volumes.
Please use the "label"  command to create a new Volume for:
    Storage:      "FileStorage" (/)
    Media type:   File
    Pool:         Default
27-Apr 13:38 bacula-dir: Client1.2011-04-27_13.38.49 Error: message.c:713 Operator
mail program terminated in error.
CMD=/usr/sbin/bsmtp -h bacula -f "(Bacula) root@bacula" -s "Bacula: Intervention
needed for Client1.2011-04-27_13.38.49" root@bacula
ERR=Child exited with code 1
```

Le premier message ci-dessus,indique qu'aucun Full Backup n'a eu lieu, alors Bacula va élever le niveau de notre job incrémental à un Full backup (c'est normal) et le second message nous dit que Bacula ne peut trouver de Volumes dans le Pool pour écrire les données en sortie. Ceci est normal parce que nous n'avons pas encore créé (ou nommé) un Volume. Bacula indique tous les détails du volume dont il a besoin. Donc à ce point, le job est BLOQUE en attendant un Volume. On pouvait le remarquer en tapant la commande *status dir*. Pour continuer, nous devons créé un Volume sur lequel Bacula peut écrire les données :

```
*label
Automatically selected Storage: File
Enter new Volume name: backuptest01

*
27-Apr 13:39 bacula-sd: Wrote label to prelabeled Volume "backuptest01" on device
"FileStorage" (/)
27-Apr 13:39 bacula-sd: Job write elapsed time = 00:00:01, Transfer rate = 69
bytes/second
27-Apr 13:39 bacula-dir: Bacula 2.0.3 (06Mar07): 27-Apr-2011 13:39:20
  JobId:                  5
  Job:                Client1.2011-04-27_13.38.49
  Backup Level:       Full (upgraded from Incremental)
  Client:             "bacula-fd" 2.0.3 (06Mar07) x86_64-redhat-linux-gnu,redhat,
  SD Errors:              0
  FD termination status:  OK
  SD termination status:  OK
  Termination:            Backup OK
```

Comme vous pouvez le remarquer la sauvegarde s'est terminée avec succès. Nous allons donc tenter de restaurer les fichiers sauvés.

```
*restore all

Avant vous sélectionner un JobIds ou plus qui contienne des fichiers
à restorer. Il vous sera présenté différentes  méthodes
pour spécifier les JobIds. Ensuite vous devrez sélectioner
lesquels fichiers de ces JobIds sont à restorer.
Select item:  (1-12): 5
Automatically selected Client: bacula-fd
Automatically selected FileSet: Full Set
+-------+-------+----------+----------+---------------------+--------------+
| jobid | level | jobfiles | jobbytes | starttime           | volumename   |
+-------+-------+----------+----------+---------------------+--------------+
|     5 | F     |        1 |        0 | 2011-04-27 13:38:52 | backuptest01 |
+-------+-------+----------+----------+---------------------+--------------+

*
27-Apr 13:45 bacula-dir: Start Restore Job RestoreFiles.2011-04-27_13.45.02
27-Apr 13:45 bacula-sd: Ready to read from volume "backuptest01" on device
 "FileStorage" (/).
  FD Errors:              0
  FD termination status:  OK
  SD termination status:  OK
  Termination:            Restore OK
```

On vient de réaliser un première sauvegarde et restauration de fichiers, en local. On va maintenant tenter de sauvegarder un machine à distance.

2.8 Mise en place d'un client distant

Si on a installé tous les services BACULA sur la machine virtuelle serveur, on ne va uniquement installer le service « bacula-client »sur les machines clientes à sauvegarder.

Avant de configurer un client Bacula sur le réseau, il est important de vérifier les règles *IP-TABLES* [12] du *Pare-feu*. L'image suivante montre les règles *iptables* existant sur les machines :

```
[root@svr4 bacula]# iptables -L
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  anywhere             anywhere

Chain FORWARD (policy ACCEPT)
target     prot opt source               destination
RH-Firewall-1-INPUT  all  --  anywhere             anywhere

Chain OUTPUT (policy ACCEPT)
target     prot opt source               destination

Chain RH-Firewall-1-INPUT (2 references)
target     prot opt source               destination
ACCEPT     all  --  anywhere             anywhere
ACCEPT     icmp --  anywhere             anywhere            icmp any
ACCEPT     esp  --  anywhere             anywhere
ACCEPT     ah   --  anywhere             anywhere
ACCEPT     udp  --  anywhere             224.0.0.251         udp dpt:mdns
ACCEPT     udp  --  anywhere             anywhere            udp dpt:ipp
ACCEPT     tcp  --  anywhere             anywhere            tcp dpt:ipp
ACCEPT     all  --  anywhere             anywhere            state RELATED,ESTABLISHED
ACCEPT     tcp  --  anywhere             anywhere            state NEW tcp dpt:ssh
REJECT     all  --  anywhere             anywhere            reject-with icmp-host-prohibited
```

FIGURE 2.10 – Les règles *iptables* existant

Mais pour simplifier les modes de connexion, nous nous sommes proposé dans un premier temps d'enlever toute règle *iptables* pouvant filtrer les protocoles réseaux avec la commande **$iptables -F**, puis dans un deuxième temps d'éditer les règles *iptables* dans la mise en œuvre de Bacula pour ne laisser passer que les paquets issus de connexions sur les ports SSH (22) et sur les ports de Bacula (9101, 9102 et 9103).

Le but étant de programmer les sauvegardes à distance, il est nécessaire de faire un test de sauvegarde sur un client en réseau, d'où l'utilisation d'une autre machine virtuelle qu'on a appelé *client*. Ce test donne les mêmes résultats que le premier, il n'y a que l'emplacement des fichiers qui change.

On pourra toute fois vérifier la liste des jobs qui ont été exécutés ou les volumes qui ont été créés pour écrire les données, grâce aux commandes qui se trouvent dans l'aide du programme *bconsole*.

12. Ensemble de règles qui permettent de filtrer les protocoles de la couche 4 dans le modèle OSI

2.9 Bilan

Jusque là nous avons réussi à explorer le fonctionnement basique du logiciel Bacula, c'est-à-dire qu'on a installé, configuré et exécuté une première sauvegarde suivie d'une restauration des fichiers sauvegardés. Les étapes les plus délicates dans cette partie sont la configuration des différents services et la mise en place de la base de données. Bacula n'est pas facile à installer et à configurer ce qui suscite la lecture approfondie du manuel technique, qui est en anglais, afin d'éviter des erreurs de configurations. Aussi il est important de lire, en cas de dysfonctionnement, les fichiers *logs*. Toutes les tâches de cette partie ont été réalisées en 3 semaines environ. Après avoir testé les principes de base du projet Bacula, nous allons maintenant procéder à son déploiement dans un environnement concret tel que celui du laboratoire et l'exploiter plus profondément à travers une utilisation approfondie.

3 Mise en œuvre de Bacula

Cette partie permet de comprendre les étapes importantes à suivre lorsqu'on met en production le projet Bacula dans un environnement d'entreprise tel que le LIX.

3.1 Sauvegarde des données du LIX

3.1.1 Politique de sauvegarde

Bacula est un logiciel puissant qui permet de sauvegarder des données mais aussi de les restaurer à tout moment sur n'importe quelle machine ; il permet aussi de faire des sauvegardes régulièrement suivant un calendrier défini, une bonne gestion des historiques de sauvegardes, de l'espace de stockage et dispose d'un algorithme de chiffrement des données pour assurer la sécurité des données sauvegardés.[2]

Le LIX étant un laboratoire et aussi un environnement de type entreprise alors les principales données à sauvegarder peuvent être les projets sur lesquels travaillent les équipes du laboratoire, les mails, les projets en ligne etc...

3.1.2 Machines à sauvegarder

Les machines à sauvegarder sont potentiellement des serveurs qui hébergent les utilisateurs comme *augias*, serveur LDAP ou leurs données comme *hydra*, serveur de fichiers NFS. Le tableau suivant nous donne le liste de ces ordinateurs et les emplacements à sauvegarder.

Serveur	Fonctions	Emplacement à sauvegarder	Type de sauvegarde	Calcul de l'espace (Go)
hydra	Serveur de fichiers NFS	/export	Sur 3 mois : – 1 Full/month – 1 Diff/week – 1 Increm/day	Sur un mois : – Full :410 – Diff :20*4 – Increm :15*26 – Total :880
argos	Serveur mail	/var/spool/mail /var/indexes	Sur 3 mois : – 1 Full/month – 1 Diff/week – 1 Increm/day	Sur un mois : – Full :80 – Diff :5*4 – Increm :4*26 – Total :204
git	Serveur git	/gitorious /etc/httpd/conf.d/ gitorious.conf	1 Full/day	
lerne	cups	/etc/cups.d/	Full/month	
oumix	Proxy Web	/etc/httpd/conf/httpd /var/www/html/lix	1 Full/month	
tissot	vpn	/etc/openvpn/	1 Full/month	
augias	LDAP	/etc/dirsrv /var/lib/dirsrv	1 Full/day	
hyppolite	Jabber	/etc/ejabberd	1 Full/month	
cerbere	Samba	/etc/samba	Sur 3 mois : – 1 Full/month – 1 Diff/week – 1 Increm/day	Sur un mois – Full :200

FIGURE 3.1 – Tableau des machines à sauvegarder

3.1.3 Plan de sauvegarde

D'après la liste des machines à sauvegarder, on a en tout trois types de sauvegarde à adopter ; ces types de sauvegarde constituent des calendriers à suivre permettant ainsi une sauvegarde automatique et régulière.

Le premier type de sauvegarde est un cycle mensuel et consiste à faire :

– Une sauvegarde complète (full) les premiers dimanches du mois, à 23h05, elle est conservée pendant un mois.
– Une sauvegarde différentielle (differential) hebdomadaire les autres dimanches, à 23h05, conservée pendant quatre (4) semaines.
– Une sauvegarde incrémentale (incremental) journalière tous les autres jours, à 23h05, conservée pendant vingt six (26) jours.

Le deuxième type de sauvegarde consiste à faire une sauvegarde totale tous les mois au premier dimanche, à 23h05.

Et le troisième type de sauvegarde consiste à faire une sauvegarde totale tous les jours à 23h05.

Les niveaux et les calendriers de sauvegarde sont faits à partir de l'importance des données à sauver, la fréquence des modifications apportées à l'emplacement à sauver et de la taille de l'emplacement à sauver.

3.1.4 Calcul d'espace de sauvegarde

Pendant la sauvegarde, Bacula crée un fichier catalogue qui contient toutes les informations liées aux travaux de sauvegarde. Ce fichier est aussi sauvegardé et sa taille, souvent de 1 à 5Mo, dépend de la quantité de données sauvées.

Cette partie consiste à détailler les calculs de sauvegarde de chaque machine. Il y a en tout neuf (9) machines qui sont des serveurs :

1. Serveur :
   – hydra
   – serveur de fichiers NFS
   – emplacement : /export

   On suppose que les données sont fréquemment modifiées à raison de 15Go par jour. La sauvegarde adopte le premier type vu ci-dessus.
   – Sauvegardes incrémentales : 26(jours)x15(Go) = 390Go
   – Sauvegardes différentielles : 4(semaines)x20(Go) = 80Go
   – Sauvegardes complètes : 410Go
   – Espace total : 880Go soit environ 900Go

   Ce cycle sera conservé pendant 3 mois.
2. Serveur :
   – argos
   – serveur mail
   – emplacements :
     – /var/spool/mail
     – /var/indexes

   On suppose que les données sont fréquemment modifiées à raisons de 4Go par jour. La sauvegarde est du premier type vu ci-dessus.
   – Sauvegardes incrémentales : 26(jours)x4(Go) = 104Go
   – Sauvegardes différentielles : 4(semaines)x5(Go) = 20Go
   – Sauvegardes complètes : 80Go

- Espace total : 204Go

Ce cycle sera conservé pendant 3 mois.

3. serveur :
   - git
   - serveur git
   - emplacements :
     - /gitorious (projets en ligne)
     - /etc/httpd/conf.d/gitorious.conf

   Le type de sauvegarde est le troisième vu ci-dessus.

   Sauvegarde complète tous les jours.
4. serveur :
   - lerne
   - serveur cups (impression)
   - emplacement : /etc/cups.d

   Le type de sauvegarde est le deuxième vu ci-dessus.

   Sauvegarde complète tous les mois.
5. Serveur :
   - oumix
   - serveur proxy web
   - emplacements :
     - /etc/httpd/conf/httpd
     - /var/www/htm/lix

   Le type de sauvegarde est le deuxième vu ci-dessus.

   Sauvegarde complète tous les mois
6. Serveur :
   - tissot
   - serveur vpn
   - emplacement : /etc/openvpn

   Le type de sauvegarde est le deuxième vu ci-dessus.

   Sauvegarde complète tous les mois.
7. Serveur :
   - augias
   - serveur LDAP
   - emplacements :
     - /etc/dirsrv
     - /var/lib/dirsrv

   Le type de sauvegarde est le troisième vu ci-dessus.

   Sauvegarde complète tous les jours.
8. Serveur :
   - hyppolite
   - serveur messagerie instantanée

   Le type de sauvegarde est le deuxième vu ci-dessus.

   Sauvegarde complète tous les mois.

9. Serveur :
   – cerbere
   – serveur de partage de dossiers et d'imprimantes

   Le type de sauvegarde est le premier vu ci-dessus.

   Sauvegardes complètes : 200Go

   Cette sauvegarde sera conservée pendant 3 mois.

Pour plus d'informations, voir dans les annexes la partie *Fichiers de configuration, bacula-dir.conf*.

A chaque sauvegarde, le directeur crée un nouveau fichier de catalogue qu'il sauvegarde et supprime après le *job*.

Grâce au catalogue on a un historique journalier des 30 derniers jours, un historique hebdomadaire des 4 dernières semaines et un historique mensuel.

3.2 installation sur serveurs physiques : srv0 et srv4

3.2.1 Installation

On dispose de deux machines-serveurs physiques, situés au sous-sol dans la salle des serveurs du laboratoire, qui vont permettre d'administrer les différents services de Bacula. Ces machines, avec le client sont configurées sur une architecture différente de celle rencontrée dans la deuxième du fait que les services sont installés et séparés, comme le montre la figure suivante :

Les flèches numérotées :
– 1re : Connexion du directeur au client et au service de stockage.
– 2e : Connexion entre client et service de stockage, le sens de connexion dépend du *job* lancé : sauvegarde ou restauration.
– 3e : Connexion connexion du service de stockage aux périphériques de stockage.

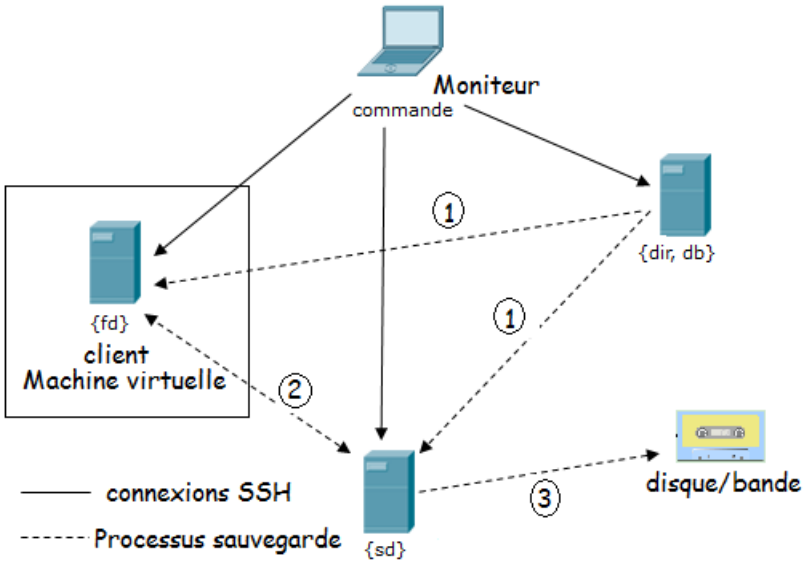


FIGURE 3.2 – Architecture de base pour la sauvegarde

Le premier serveur appelé svr0, de nom de domaine *svr0.lix.polytechnique.fr* et d'une adresse IP *129.104.11.180*, dispose du système d'exploitation *Scientific Linux release 6.0 (Carbon)*. On installe sur cette machine le projet Bacula de version encore plus récente que celle utilisé pour les tests. Il s'agit de *Bacula version 5.0.0 (26 January 2010)*.

Les services Bacula qui fonctionnent sur cette machine sont le directeur qui est normalement unique et le démon *bacula-fd* ou *file daemon*. Aussi la base des données qui gère les informations de Bacula est installée sur le serveur svr0, ici on utilise toujours *PostgreSQL* qui efficace et tout à fait adapté au projet Bacula en entreprise.

Le deuxième serveur appelé svr4 de nom de domaine *svr4.lix.polytechnique.fr*, d'adresse IP *129.104.11.180* dispose aussi du système d'exploitation *Scientific Linux release 6.0 (Carbon)*. On installe aussi sur cette machine Bacula de la même version.

Les services qui fonctionnent sont le démon *bacula-sd* ou *storage daemon* et le démon *bacula-fd*. Ainsi c'est sur ce serveur qu'on va vérifier les bandes de stockage et piloter le robot de sauvegarde qui permet de vérifier l'accès aux périphériques de stockage et leur liaison aux ordinateurs.

3.2.2 Configuration

Pour l'installation de Bacula sur ces machines, on va installer différents paquets en exécutant dans un terminal *shell* :

```
Sur le serveur svr0, installer:
yum install bacula-director-common.x86_64     #le service directeur
yum install bacula-client.x86_64              #le service client
yum install bacula-console.x86_64             #le programme bconsole
yum install bacula-docs.x86_64                #la documentation bacula
yum install bacula-console-bat.x86_64         #interface graphique d'administration
yum install bacula-director-postgresql.x86_64 #bacula-dir avec le support PostgreSQL
yum install postgresql-server.x86_64          #Pour créer et exécuter la base de données
yum install bacula-traymonitor.x86_64         #Connexion en mode graphique

On installe sur svr4:
yum install bacula-storage-common.x86_64      #le service de stockage
```

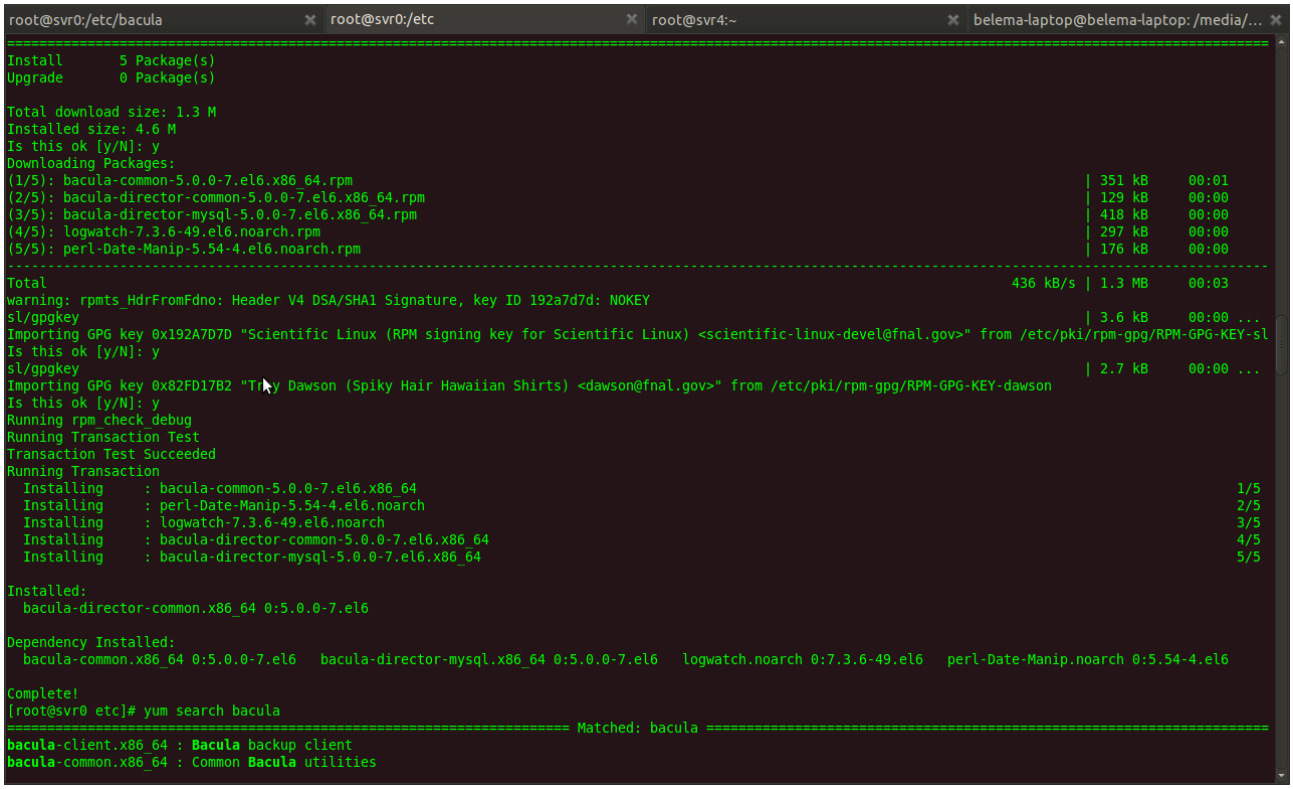


FIGURE 3.3 – Exemple : installation du service directeur.

Après l'installation de Bacula et de tous les programmes dont il dépend (base de données, les clés publiques...), on va configurer la base de données, le service *bacula-directeur* sur la machine-serveur *svr0*, le service de stockage sur la machine-serveur *svr4* ainsi que le service *bacula-file* sur les différentes machines dont on souhaite sauver les données.
Dans la *partie 2*, la configuration de la base de donnée commence en la démarrant avec la commande */etc/init.d/postgresql start*, mais avec la version récente de PostgreSQL(8.4.7-1.el6_0.1) il faut d'abord créer le répertoire qui va contenir les informations du fonctionnement de la base de données, avant de la démarrer.

```
Aller dans l'environnement bash-4.1$ en se connectant sous l'utilisateur
```

```
postgres et créer le répertoire avec les informations de fonctionnement.
[root@svr0 bacula]# su -- postgres
bash-4.1$ /usr/bin/initdb -D /var/lib/pgsql/data
The files belonging to this database system will be owned by user "postgres".
This user must also own the server process.

The database cluster will be initialized with locale en_US.UTF-8.
The default database encoding has accordingly been set to UTF8.
The default text search configuration will be set to "english".

fixing permissions on existing directory /var/lib/pgsql/data ... ok
creating subdirectories ... ok
selecting default max_connections ... 100
selecting default shared_buffers ... 24MB
creating configuration files ... ok
creating template1 database in /var/lib/pgsql/data/base/1 ... ok
initializing pg_authid ... ok
initializing dependencies ... ok
creating system views ... ok
loading system objects' descriptions ... ok
creating conversions ... ok
creating dictionaries ... ok
setting privileges on built-in objects ... ok
creating information schema ... ok
vacuuming database template1 ... ok
copying template1 to template0 ... ok
copying template1 to postgres ... ok

WARNING: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the -A option the
next time you run initdb.

Success. You can now start the database server using:

    /usr/bin/postgres -D /var/lib/pgsql/data
or
    /usr/bin/pg_ctl -D /var/lib/pgsql/data -l logfile start

bash-4.1$
```

On pourra continuer la suite de la configuration comme indiqué dans la *partie2*.
Après avoir configuré correctement tous les services en suivant la même méthode que lors du test dans la *partie 2*, on pourra lancer le programme *bconsole* afin de commander interactivement les services Bacula.

3.3 Vérification des bandes de stockage

Comme mentionné un peu plus haut, on effectuera la vérification des bandes de stockage à partir de la machine svr4. De cette manière on peut vérifier l'état des bandes de stockage et contrôler le lecteur de bande grâce au programme ***btape*** dédié à Bacula. On peut également piloter le robot de sauvegarde inclus qui permet d'automatiser le processus de stockage. Configurer le lecteur de bande, revient aussi à spécifier la taille des block de données sur les bandes.

Grâce au programme *btape*, on peut déterminer la meilleure configuration d'un lecteur de bande ; *btape* ne fonctionne uniquement qu'avec les périphériques de stockage sur bande.

On dispose aussi de ***mtx***, programme de contrôle pour auto-changeur de bande avec une interface de type SCSI, pour piloter le robot de sauvegarde ; ***tapeinfo*** pour avoir des informations caractéristiques des périphériques de stockage qui peuvent permettre de savoir s'il s'agit d'une bande de stockage, d'un lecteur de disk ou d'un changeur, et aussi le type d'interface de ces périphériques (scsi) ; et ***loaderinfo*** pour avoir des informations sur les éléments de stockage fournies par le lecteur de bandes (leur nombre). D'autres commandes sont disponibles en tapant la commande **-help** de ***btape***.

On utilise pour stocker les données l'auto-changeur *Powervault TL2000* pouvant contenir jusqu'à 24 cartouches de bandes de type LTO-4 disposant d'un espace de sauvegarde de 1.6To chacune.Cet auto-changeur est supporté par le projet Bacula d'où le fait de configurer le lecteur de bande avec le programme *btape*. On dispose donc d'un espace de stockage suffisamment élevé dans le magasin des cartouches pour stocker nos données. Le point d'entrée de périphérique */dev/nst0* correspond au lecteur de bande de l'auto-changeur et le point d'entrée de périphérique */dev/changer-sg4* correspond au robot de sauvegarde.

On va procéder à un exemple d'utilisation du programme *btape* ainsi que du programme *mtx* ; la commande *help* permet d'afficher les commandes disponibles dans ces programmes ; avant de commencer il faut toujours arrêter le service de stockage (bacula-sd) :

1. Test *mtx* : Ce programme peut permettre d'afficher l'état des *slots* qui contiennent les bandes et de charger une bande dans le lecteur de bande :

```
[root@svr4 /]# mtx -f /dev/changer-sg4 load 1
Loading media from Storage Element 1 into drive 0...done
[root@svr4 /]# mtx -f /dev/changer-sg4 status
  Storage Changer /dev/changer-sg4:1 Drives, 24 Slots ( 1 Import/Export )
Data Transfer Element 0:Full (Storage Element 1 Loaded):VolumeTag = 000001L4
      Storage Element 1:Empty
      Storage Element 2:Full :VolumeTag=000002L4
      Storage Element 3:Full :VolumeTag=000003L4
      Storage Element 4:Full :VolumeTag=000004L4
      Storage Element 5:Full :VolumeTag=000005L4
      Storage Element 6:Full :VolumeTag=000006L4
      Storage Element 7:Full :VolumeTag=000007L4
      Storage Element 8:Full :VolumeTag=000008L4
      Storage Element 9:Full :VolumeTag=000009L4
      Storage Element 10:Full :VolumeTag=000010L4
      Storage Element 11:Full :VolumeTag=000011L4
      Storage Element 12:Full :VolumeTag=000012L4
      Storage Element 13:Full :VolumeTag=000013L4
      Storage Element 14:Full :VolumeTag=000014L4
      Storage Element 15:Full :VolumeTag=000015L4
      Storage Element 16:Full :VolumeTag=000016L4
      Storage Element 17:Full :VolumeTag=000017L4
      Storage Element 18:Full :VolumeTag=000018L4
      Storage Element 19:Full :VolumeTag=000019L4
      Storage Element 20:Full :VolumeTag=000020L4
      Storage Element 21:Full :VolumeTag=000021L4
      Storage Element 22:Full :VolumeTag=000022L4
      Storage Element 23:Full :VolumeTag=000023L4
      Storage Element 24 IMPORT/EXPORT:Full :VolumeTag=000024L4
[root@svr4 /]#
```

FIGURE 3.4 – Résultat de commande mtx.

2. Test *btape* :

Ce programme peut permettre de configurer la vitesse du lecteur de bande et marquer la fin des fichiers écris (End of Files EOF) afin de permettre d'écrire des fichiers issus de plusieurs jobs. On va lancer un test général des fonctions de la bande utilisée par Bacula :

```
[root@svr4 /]# btape -v /dev/nst0
Tape block granularity is 1024 bytes.
btape: butil.c:285 Using device: "/dev/nst0" for writing.
01-Jun 00:23 btape JobId 0: 3301 Issuing autochanger "loaded? drive 0" command.
01-Jun 00:23 btape JobId 0: 3302 Autochanger "loaded? drive 0" result is Slot 1.
btape: btape.c:476 open device "TL2000" (/dev/nst0): OK
*test

=== Write, rewind, and re-read test ===

I'm going to write 10000 records and an EOF
then write 10000 records and an EOF, then rewind,
and re-read the data to verify that it is correct.

This is an *essential* feature ...

btape: btape.c:1148 Wrote 10000 blocks of 64412 bytes.
btape: btape.c:608 Wrote 1 EOF to "TL2000" (/dev/nst0)
```

FIGURE 3.5 – Commande btape1.

On teste aussi l'auto-changeur :

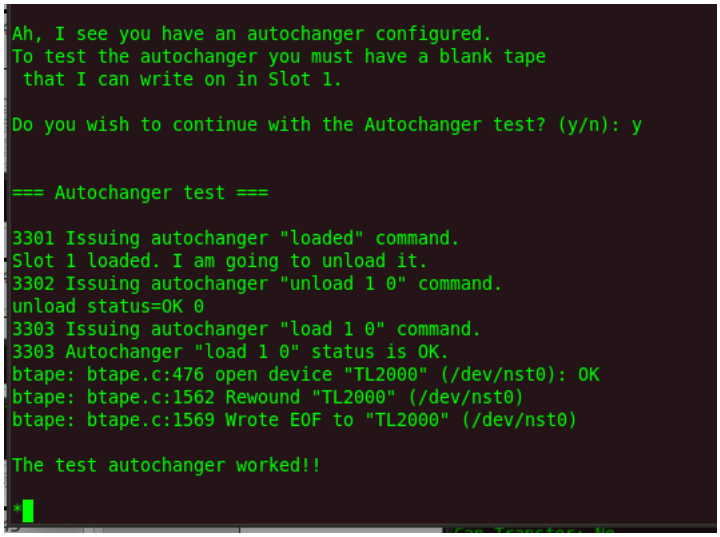


FIGURE 3.6 – Commande btape2.

3.4 Sauvegarde et restauration d'une machine

3.4.1 Gestion de l'espace de sauvegarde

Avant de lancer les sauvegardes, on configure les paramètres liés à la ressource ***Pool*** dans le fichier de configuration du directeur afin de gérer au mieux l'espace de stockage.
En effet, dans la ressource ***Pool***, on peut définir la taille maximale du *pool* avec deux options qui sont, ***Maximum Volume Bytes*** qui permet de déterminer la taille maximale d'un volume du pool, elle sera par défaut la même pour les volumes qui seront créés (on peut modifier la taille selon le volume) ; et ***Maximum Volume*** qui définit le nombre maximal de volumes de stockage que peut contenir un pool.
Ensuite il y a aussi une autre option ***Volume Retention*** qui permet de spécifier la durée minimale de conservation d'un volume du pool avant qu'il ne soit réutilisé, il sera toujours au moins le double de l'intervalle de temps qui sépare deux sauvegardes complètes. Ce délai sera le même par défaut pour tous les volumes (on peut modifier sa valeur selon le volume). Ainsi Bacula va recycler un volume si sa période d'utilisation est dépassée et qu'il est plein, en l'écrasant et en effaçant automatiquement les fichiers sur le volume du catalogue.
Enfin il y a différentes manières de créer un volume : au moment de la sauvegarde, il n'y a pas de volume dans le pool ou le périphérique de stockage n'a pas de nom, alors on le crée avec la commande *label* dans la console ; on peut ajouter un volume au pool sans donner de nom explicitement à un volume physique avec la commande *add* dans la console ; on peut également nommer, dans l'environnement *btape*, un volume correspondant à une bande avec la commande *btape*, l'inconvénient en est que ce volume ne sera pas ajouté dans un pool et ce ne sera possible que lorsqu'on exécute un *job* et qu'on doit explicitement spécifier un pool à utiliser. (voir figures *Volume de stockage de fichiers* et *Commande label sous btape*). La méthode recommandée pour nommer un volume physique et l'ajouter à un pool est celle qui consiste à le faire dans la console avec la commande *btape*. Après chaque modification dans un pool il faut exécuter la commande **update pool** pour mettre à jour les informations du pool

dans la base des données.

```
*label
Enter Volume Name: vol001
Wrote Volume label for volume "vol001".
*readlabel
btape: btape.c:528 Volume label read correctly.

Volume Label:
Id                : Bacula 1.0 immortal
VerNo             : 11
VolName           : vol001
PrevVolName       :
VolFile           : 0
LabelType         : PRE_LABEL
LabelSize         : 172
PoolName          : Default
MediaType         : LTO-4
PoolType          : Backup
HostName          : svr4.lix.polytechnique.fr
Date label written: 02-Jun-2011 11:37
```

FIGURE 3.7 – Commande *label* sous *btape*.

La figure suivante permet de visualiser une gestion possible d'un espace de stockage.

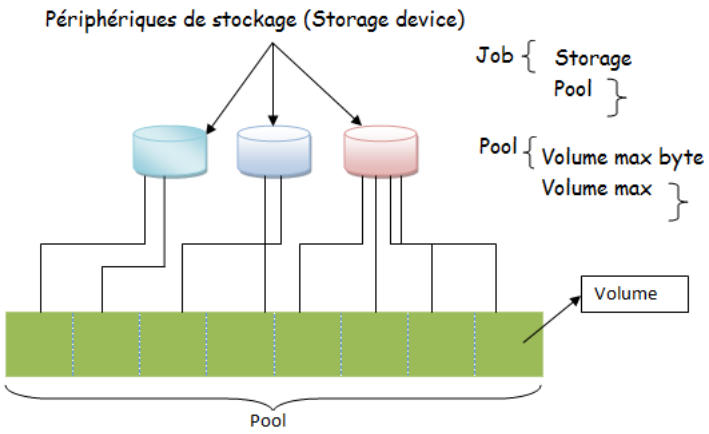


FIGURE 3.8 – Gestion d'espace de stockage.

En pratique nous avons utilisé la commande *add* du programme *bconsole* pour créer 24 volumes dans le pool *Default*. Ensuite, dans l'environnement du programme *btape*, nommer les 24 bandes de la bibliothèque de l'auto-changeur *TL2000* de la même manière que les volumes du pool. Ainsi lorsqu'une bande sera utilisée, *Bacula* va chercher dans le catalogue un volume disponible du pool dont le nom correspond à celui de la bande pour éviter que *Bacula* ne demande de créer un nouveau volume à chaque fois qu'une bande sera utilisée. Enfin on s'assure que le numéro des *slots* corresponde bien à ceux de l'auto-changer ; *Bacula* ne pourra pas utiliser un volume

de l'auto-changeur si le nom du volume et son numéro de *slot* ne sont pas cohérents avec les informations du pool dans le catalogue.
Ces modifications sont possibles dans la console du programme *bconsole* grâce à la commande *update* et sélectionner les paramètres des volumes *Volumes Parameters*.

3.4.2 Sauvegarde

La machine cliente étant une machine se trouvant sur le réseau, il est important de vérifier les règles **IPTABLES** avant d'exécuter un job (voir la partie 2.8).

On vérifie d'abord les bandes de stockage, dans ce cas-ci on sauvegarde des fichiers sur un disk dur interne géré par le système de fichiers sur le serveur *svr4*. La figure suivante indique l'arborescence du volume de stockage dans le système de fichier (*/tmp*)

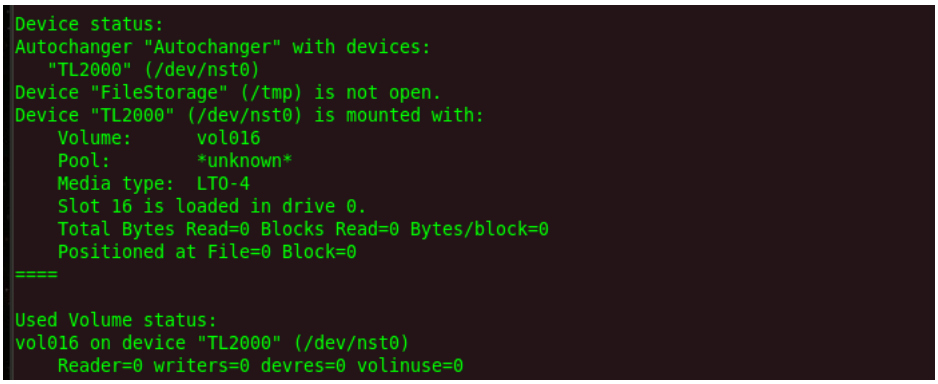


FIGURE 3.9 – Volume de stockage de fichiers

On vérifie que la base données est démarrée, ensuite on démarre *bacula-dir*, sur la machine-serveur *svr0*.

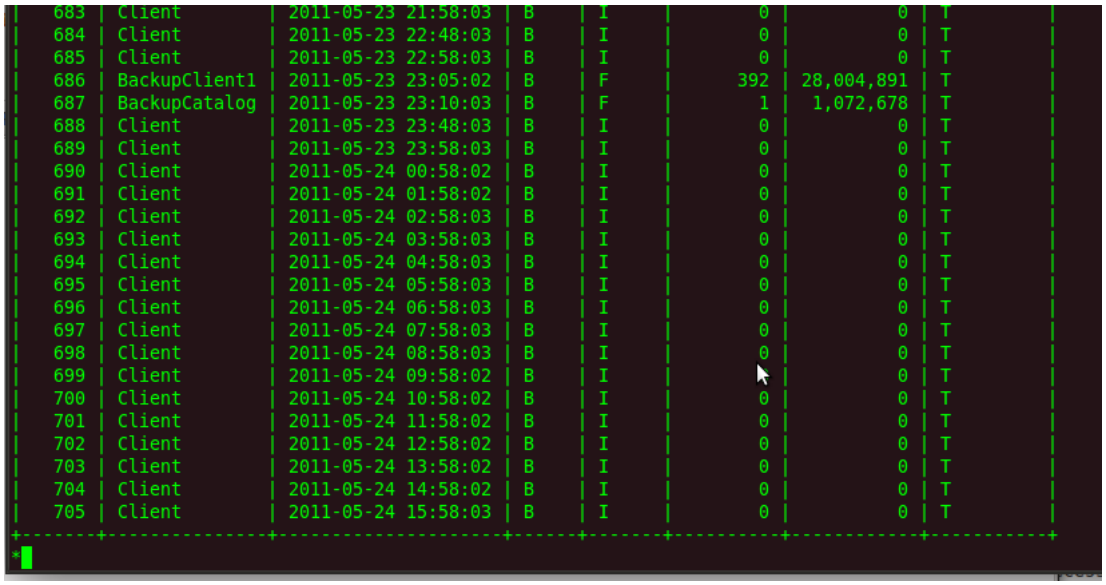
Sur la machine *svr4* on démarre le service *bacula-sd* et sur la machine cliente on démarre le service *bacula-fd*.

La machine qu'on va sauvegarder est la machine virtuelle cliente utilisée pour le test c'est-à-dire qu'on va sauvegarder les mêmes fichiers (*/tests*, sauf qu'on va changer les paramètres de connexion au directeur (adresse, mot de passe..)). La version de Bacula installé sur la machine cliente est plus ancienne et c'est celle utilisée pour le test c'est-à-dire la 2.0.3 (06 Mars 2007), sachant que les services directeur et de stockage de version récente supportent les clients de version plus ancienne.

Dans le fichier de configuration du directeur on retrouve les mêmes paramètres de configuration du client que ceux du test de la partie 2.7, sauf le calendrier de sauvegarde qui dans ce cas est un cycle de sauvegardes incrémentales séparées de 1 heure chacune.

```
#Le calendrier de sauvegarde
```

```
Schedule {
  Name = "HourlyCycle"
  Run = Incremental hourly at  18:58
}
```



683	Client	2011-05-23 21:58:03	B	I	0	0	T	
684	Client	2011-05-23 22:48:03	B	I	0	0	T	
685	Client	2011-05-23 22:58:03	B	I	0	0	T	
686	BackupClient1	2011-05-23 23:05:02	B	F	392	28,004,891	T	
687	BackupCatalog	2011-05-23 23:10:03	B	F	1	1,072,678	T	
688	Client	2011-05-23 23:48:03	B	I	0	0	T	
689	Client	2011-05-23 23:58:03	B	I	0	0	T	
690	Client	2011-05-24 00:58:02	B	I	0	0	T	
691	Client	2011-05-24 01:58:02	B	I	0	0	T	
692	Client	2011-05-24 02:58:03	B	I	0	0	T	
693	Client	2011-05-24 03:58:03	B	I	0	0	T	
694	Client	2011-05-24 04:58:03	B	I	0	0	T	
695	Client	2011-05-24 05:58:03	B	I	0	0	T	
696	Client	2011-05-24 06:58:03	B	I	0	0	T	
697	Client	2011-05-24 07:58:03	B	I	0	0	T	
698	Client	2011-05-24 08:58:03	B	I	0	0	T	
699	Client	2011-05-24 09:58:02	B	I	0	0	T	
700	Client	2011-05-24 10:58:02	B	I	0	0	T	
701	Client	2011-05-24 11:58:02	B	I	0	0	T	
702	Client	2011-05-24 12:58:02	B	I	0	0	T	
703	Client	2011-05-24 13:58:02	B	I	0	0	T	
704	Client	2011-05-24 14:58:02	B	I	0	0	T	
705	Client	2011-05-24 15:58:03	B	I	0	0	T	

FIGURE 3.10 – Sauvegarde d'un cycle de 1 heure

3.4.3 Restauration de la machine

Avant de restaurer les fichiers sauvés, on va détruire la machine virtuelle, puis la recréer, car dans les cas extrêmes comme les incendies, on aura besoin de réinstaller les machines, mais souvent on a à faire à des cas moins graves comme la mauvaise manipulation d'une commande système comme *rm ** dans un répertoire important.
Pour la restauration on va ensuite réinstaller *Bacula backup client* et donc principalement le service *bacula-fd* en suivant la même méthode que dans la partie *installation de bacula* ; on a pas de modification à faire du côté du directeur étant donné que la machine à restaurer était déjà client de Bacula.
La procédure de restauration est la même que celle vue dans la partie 2.7 *premier test de sauvegarde*.
Voici un extrait de la restauration de fichiers dans la console :

```
JobName:         RestoreFilesdist
Bootstrap:       /var/lib/bacula/svr0.lix.polytechnique.fr-dir.restore.1.bsr
Where:           /tmp/bacula-restores
Replace:         always
FileSet:         Full Set client
Backup Client:   client-fd
Restore Client:  client-fd
Storage:         File
When:            2011-05-27 15:16:16
Catalog:         MyCatalog
Priority:        10
Plugin Options:  *None*
OK to run? (yes/mod/no):  yes
Job queued. JobId=782
```

3.5 Intégrer tous les clients

Il s'agit maintenant d'installer le service *Bacula-fd* sur chaque machine dont les fichiers seront sauvegardés et indiquer les paramètres adéquats pour la connexion avec les autres services.
Ensuite, dans le fichier de configuration du directeur, on va définir pour chaque *client*, un *JobDefs* et un *Job* pour la sauvegarde à exécuter, un *job* pour la restauration des données qui sera adapté par le programme Console qu'on utilise à chaque client dont on souhaite restaurer les fichiers, ce job est unique pour tous les autres jobs de sauvegarde, les clients et les ressources *Storage* ; un *Fileset* pour les emplacements à sauvegarder, programmer les calendriers adaptés à chaque type de sauvegarde et définir les clients à sauvegarder avec la ressource *Client*.
On définit ensuite les règles *IPTABLES* qui permettent d'ouvrir les connexions entre les machines, en n'autorisant que les connexions sur les ports TCP de Bacula et le port 22 de SSH.

On s'assure que les périphériques de stockage sont connectés, que la base de données est démarrée, et on redémarre les services. Ainsi avec la commande *status dir* dans la console, on devrait voir dans la liste des *jobs* qui sont programmés pour les sauvegardes.

3.6 Mise en place de la console Bat

Afin d'avoir une administration transparente et conviviale, avec plus de fonctionnalités notamment comme des graphes, on va mettre en place l'interface graphique d'administration Bat (Bacula Administration Tool) de version *bacula-bat-5.0.0 26 junuary 2010*. L'avantage en est que l'on ne tape des commandes mais on clique juste sur des boutons pour par exemple programmer un *job* ; la figure suivante nous présente l'interface bat qu'on utilise :

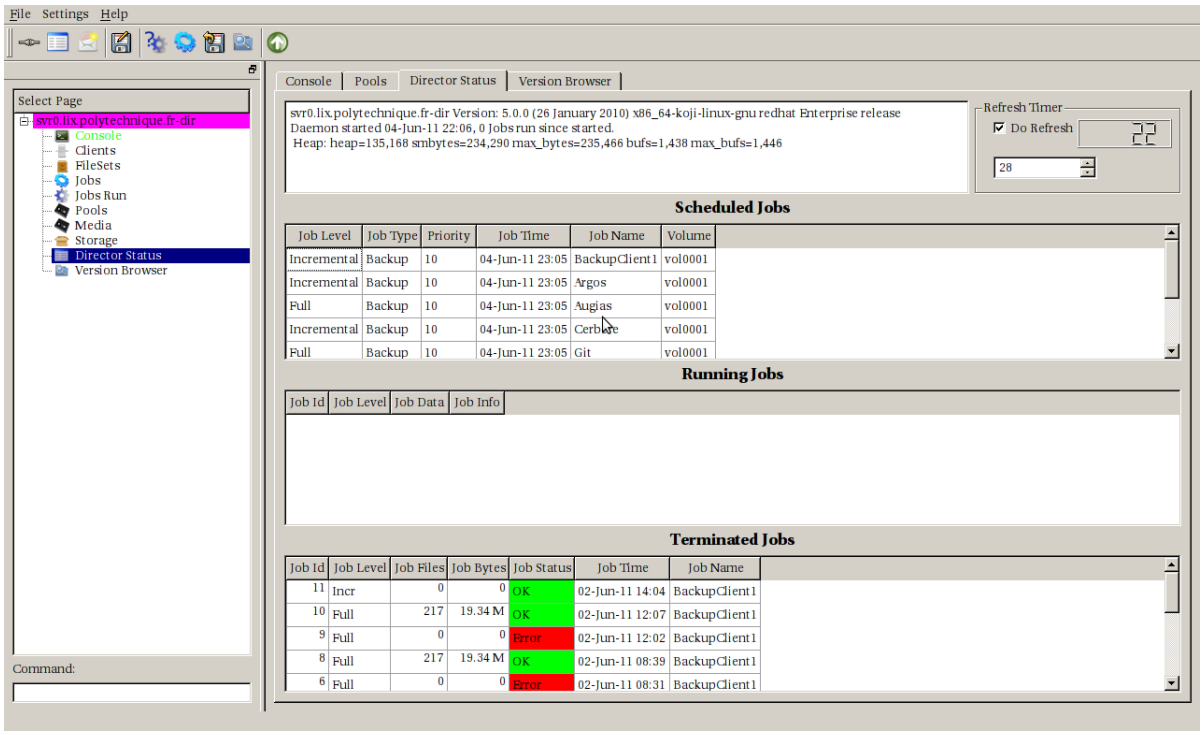


FIGURE 3.11 – Bat :Bacula Admin Tool

3.7 Avantages du projet Bacula

Bacula dispose de nombreux avantages sur d'autres logiciels de sauvegarde, du fait qu'il y a un client pour chaque machine on peut sauvegarder et restaurer des clients de tous types avec l'assurance que tous les attributs de fichiers sont convenablement sauvegardés et restaurés. Il prend en charges les sauvegardes multi-volumes c'est-à-dire qu'on a un vaste choix de supports de stockage ; Une base de données complète aux standards SQL de tous les fichiers sauvegardés. Ceci permet une vue en ligne des fichiers sauvegardés sur n'importe quel volume et une destruction automatique des anciens enregistrements, ce qui simplifie l'administration de le base de données. On peut configurer dans un *job* l'arrêt et le démarrage des applications avec les outils natifs du système sauvegardé, avec la directive *ClientRunBeforeJob* dans un job[1].

1. Voir annexes *B.1 Configuration du directeur*

Le projet Bacula est supporté par tous les systèmes d'exploitation souvent utilisés (Windows, UNIX/Linux) [2]. Enfin Bacula est aussi rapide que la grande application commerciale majeure.

3.8 Bilan

En fin de cette partie, les principales tâches de cette partie ont été réalisées.

La réelle difficulté dans la mise en œuvre de Bacula réside dans l'étape de stockage des données car il faut avoir une bonne configuration et administration surtout du système d'automatisation de bandes (Powervault TL2000) et des bandes de stockage qu'on a à notre disposition. En effet le type du périphérique qu'on utilise requiert une configuration adéquate en faisant attention aux options à spécifier pour permettre sa prise en charge par les services de stockage et du directeur.
Bacula est aussi difficile à mettre en œuvre du fait qu'il est doté de beaucoup d'options et de détails dont il faut tenir compte pour mettre en place un bon système de sauvegarde.
Bacula a été classé parmi les meilleures solutions *open source* non seulement du fait de sa fiabilité et de son efficacité, mais aussi de par sa grande flexibilité et son infinité de fonctionnalités. Pourtant, Bacula n'a pas que des avantages suivant ce que l'on veut faire.

2. Voir annexes *Sauvegarde Windows*

Retour d'expériences

Pendant la période de mon stage, j'ai reçu la mission de mettre en œuvre c'est-à-dire d'installer, de configurer et de déployer *Bacula* dans le réseau informatique du LIX, dont l'objectif est de sauvegarder et de restaurer d'une manière efficace, rapide et centralisée les données des ordinateurs du réseau qui sont principalement les serveurs. Ainsi j'ai d'abord évaluer le fonctionnement de base du logiciel dans une architecture constituée de machines virtuelles avant de le déployer et d'explorer ses fonctionnalités sur les machines physiques du LIX.

Au cours de la formation de l'IUT j'ai acquis des connaissances en réseaux et en informatique que j'ai réinvesties durant le stage de dix semaines au laboratoire, notamment dans la notion de Client/Serveur car Bacula est composé de différents services dont le directeur, unique q'on installe sur un serveur et les services clients installés sur les machines à sauvegarder dont les données sauvées sont spécifiées par le directeur. Cette notion permet aussi d'approfondir les connaissances dans l'architecture des réseaux car pour communiquer sur un réseau on se sert des protocoles dont le modèle TCP/IP, en effet Bacula utilise des port TCP/IP bien définis (par exemple le port 9103 pour le directeur). Je me suis aussi familiarisé avec le monde UNIX/Linux sur des machines utilisant les distributions Scientific Linux 6.0 carbon ou Cent OS 5.5, uniques systèmes sur lesquels j'ai travaillé durant le stage. J 'ai aussi développé quelques facultés linguistiques du fait que la documentation technique de Bacula est en anglais et j'ai acquis de nouvelles méthodes de travail comme l'utilisation du langage LaTeXpour la rédaction de mon rapport. Enfin ce stage m'as permis d'avoir une vision panoramique des réseaux.

Je suis pleinement satisfait du travail réalisé car il m'a permis de découvrir nouvelles compétences, notamment celle de sauvegarde des systèmes informatiques ; car Bacula permet non seulement de sauvegarder des données sur un système de stockage mais permet également de sauvegarder d'une manière sécurisée grâce à un algorithme de cryptage, rapide et organisée du fait que les tâches soient séparées ; l'une des étapes importantes dans un processus de sauvegarde reste la gestion de l'espace de stockage et plus particulièrement des périphériques de stockage qui peuvent être un appareil sophistiqué comme le *Powervault TL2000* utilisé. Bacula permet aussi de recevoir soit dans la console ou sur un compte de messagerie, en temps réel, les informations liées au déroulement des jobs ou des messages d'erreurs de fonctionnement tel qu'un problème de connexion ; ce qui permet de résoudre à distance et via SSH, rapidement les problèmes. Je n'ai pas pu mettre en œuvre l'application *bweb* qui est une interface web d'administration, mais ce problème est comblé par l'autre interface Bat qui remplit toutes les fonctions à savoir du *bweb* et de la console. Enfin Bacula est l'un des projets les pl :us réussis du monde des solutions open source car il est extrêmement

flexible du fait qu'il fonctionne sur tout type de système couramment utilisé et qu'il dispose de nombreuses fonctionnalités ; c'est aussi une solution qui est adaptée aux entreprises car aussi rapide que la solution majeure de sauvegarde en entreprise. Mon travail pourra être repris.

Je me suis intégré assez facilement dans la vie du laboratoire. J'ai eu de bonnes relations pédagogiques avec mon maître de stage qui a envie d'apprendre et avec quelques chercheurs du laboratoire. J'ai pu participer à une activité de foot-ball avec certains chercheurs du laboratoire. La dynamique du laboratoire pousse à travailler sérieusement.

Ce stage m'a permis de développer des compétences tant du point de vue relationnel que professionnel. En effet tout au long du stage, en m'appropriant le sujet j'ai été maître et donc responsable du travail à réaliser ; ensuite le cadre et l'environnement du travail, différent de l'entreprise, m'ont permis d'avoir plus de confiance en moi. Aussi le fait d'effectuer ce stage dans un autre environnement géographique m'a permis de développer de l'autonomie et mon sens de responsabilité.
Disposant de bonnes bases pour poursuivre mes études, je compte, après le DUT, continuer dans une formation d'ingénieur par apprentissage : en informatique et réseaux, domaines dans lesquels je veux m'impliquer et approfondir certaines notions abordées ; ou en électronique, formation qui offre l'opportunité de découvrir les nouvelles technologies ; ou encore continuer dans la formation de l'IUP STRI (Systèmes de Télécommunications et Réseaux Informatiques).

Index

Bibliographie

[1] `http://www.bacula.org/fr/`.

[2] Administration et développement des systèmes UNIX. *GNU LINUX MAGAZINE/France*, (N138) :41–45, Mai 2011.

[3] Bacula. Documentation. *fichier main.pdf*, 428 pages.

[4] `http://fedoraproject.org/wiki/EPEL`. Dépôt de dossiers rpm.

[5] `http://sourceforge.net/`. Logiciels libres open source.

[6] `http://wiki.lix.polytechnique.fr/FrontPage`.