VEILLE TECHNOLOGIQUE SUR LES OUTILS DE SAUVEGARDES

Table des matières

Introduction 3
1)Compréhension du cahier des charges 3
1.1) Méthode prises en charges de sauvegarde prises en charge 3
1.1.1) Sauvegarde Total 4
1.1.2) Sauvegarde différentielle 4
1.1.3) Sauvegarde incrémentielle 4
1.1.4) Règle de sauvegarde 3-2-1 5
1.1.5) Durée de rétention 6
1.1.6) Rotation GFS (Grandfather – Father – Son) 6
1.2) Gestion des scripts avant/après la sauvegarde (pre/post-job). 7
1.2.1) Script Pre-JOB 7
1.2.2) Script Post-Job 7
1.3) Paramétrage des jobs 8
1.4) Rapport de sauvegarde automatisé et possibilité de rotation/conservation des logs. 8
1.5) Support de différents protocoles pour la cible de sauvegarde 9
1.5.1) RSYNC 9
1.5.2) SSH 9
1.5.3) SFTP (SSH File Transfer Protocol) 10
1.5.4) SMB 10
1.5.5) NFS (Network File System) 11
1.6) Nécessité ou non d'installer un agent sur les postes à sauvegarder. 11
1.7) Modes de transfert 12
1.7.1) Mode Push 12
1.7.2) Mode Pull (la destination récupère) 12
2) Recherche et sélection des outils 13
2.1) BorgBackup 13
2.2) Bacula 13
2.3) UrBackup 14
2.4)Rclone 14
2.5) Amanda / Zmanda 15
2.6) Duplicity 15
2.7) Sélection des 3 solutions qui me semble les plus pertinentes 15
3)Analyse détaillée par critères 16
4) **Synthèse et préconisations** 21
4.1) la solution la plus adaptée au contexte 21

4.2) indice de pondération.....21
4.3) Dans quels cas chaque solution est préférable : 22
Liens : 22

Introduction

Dans le contexte actuel des systèmes d'information, la sauvegarde des données représente un enjeu critique pour assurer la continuité de service et la résilience face aux sinistres, aux pannes matérielles ou aux cyberattaques (ransomwares). L'objectif de cette veille technologique est d'identifier, d'analyser et de comparer trois solutions professionnelles de sauvegarde de données spécifiquement adaptées à l'environnement GNU/Linux.

Cette démarche vise à sélectionner les outils les plus pertinents pour répondre aux exigences précises d'un cahier des charges donné, en privilégiant la robustesse, la conformité (notamment à la règle 3-2-1 et au RGPD), la gestion des infrastructures complexes et l'intégration dans un écosystème d'entreprise.

Démarche Suivie

La réalisation de ce dossier s'est articulée autour des étapes structurées suivantes :

1. J'ai commencé par définir toutes les demandes du cahier des charges pour pouvoir être plus pertinent dans ma réponse.
2. J'ai ensuite cherché puis sélectionné 6 outils de sauvegardes sous GNU/Linux en relevant leurs fonctionnalités, popularité, documentation, et compatibilité avec le cahier des charges. Et parmi ces 6 j'ai sélectionné les 3 qui me semblé être les plus pertinents
3. Ensuite j'ai fait une **Analyse Détaillée :** Ces trois outils ont fait l'objet d'une analyse comparative objective, critère par critère, afin de mettre en lumière leurs fonctionnalités, leurs forces et leurs limites.
4. **J'ai ensuite fait une Synthèse et Préconisations :** j'ai dit lequel des 3 me semblé le mieux et pourquoi ? Puis j'ai fait un tableau comparatif intégrant un indice de pondération qui a permis de dégager la solution la plus adaptée dans chaque contexte professionnel (PME/grande entreprise) et de définir les cas d'usage privilégiés pour chacune des solutions sélectionnées.

Ce document propose donc une évaluation argumentée pour guider le choix d'une solution de sauvegarde professionnelle sous GNU/Linux.

1)Compréhension du cahier des charges

1.1) Méthode prises en charges de sauvegarde prises en charge

Il existe différente méthode de sauvegarde : totale, différentielle, incrémentielle, conformité à la règle 3-2-1, durée de rétention, rotation GFS

1.1.1) Sauvegarde Total

La sauvegarde **complète (ou totale)** copie **l'intégralité** des données à chaque fois.

Avantages :

- Restaurations très rapides (tout est dans un seul backup).
- Très simple à comprendre.

Inconvénients :

- Très long à faire.
- Très gros en taille.

1.1.2) Sauvegarde différentielle

Après une sauvegarde totale, une sauvegarde différentielle copie **toutes les modifications depuis la dernière Full**.

Exemple :
Dimanche : Full
Lundi : Diff = changements depuis dimanche
Mardi : Diff = changements depuis dimanche (Lundi + Mardi)
Mercredi : Diff = changements depuis dimanche (Lundi + Mardi + Mercredi)

Pour restaurer :
Full **+** dernière sauvegarde différentielle.

Avantage : restauration plus rapide que l'incrémentielle.

Inconvénient : les fichiers différentiels grossissent chaque jour.

1.1.3) Sauvegarde incrémentielle

Principe :
Après une Full, la sauvegarde incrémentielle copie **uniquement les changements depuis la dernière sauvegarde (qu'elle soit Full ou incrémentielle)**.

Exemple :

- Dimanche : Full
- Lundi : Incrémentielle = changements depuis dimanche
- Mardi : Incrémentielle = changements depuis lundi

- Mercredi : Incrémentielle = changements depuis mardi

Pour restaurer : Full + la chaîne complète des incréments (Lundi, Mardi, Mercredi).

Avantages :

- Très rapide.
- Très faible espace utilisé.

Inconvénients :

- Restauration plus longue.
- Si un incrément est corrompu, la chaîne est cassée.

1.1.4) Règle de sauvegarde 3-2-1

C'est la règle standard pour **sécuriser les données**.

3 copies de vos données

- 1 originale
- 2 copies supplémentaires

2 types de supports différents
Exemples :

- Disque dur + cloud
- NAS + bande
- PC + clé USB

1 copie hors site

- Sur un autre lieu physique
- Sur le cloud
- Chez un prestataire

Pour résister à :

- Panne matérielle
- Virus / ransomware
- Incendie, vol, inondation
- Erreur humaine

1.1.5) Durée de rétention

C’est-à-dire : combien de temps **l'on conserve** les sauvegardes.

Exemples courants :

- **Rétention courte** : 7 jours
- **Rétention moyenne** : 30 jours
- **Rétention longue** : 1 an
- **Archivage** : 10 ans (pour les documents légaux)

Les entreprises fixent la rétention selon :

- Le besoin de restauration
- Les obligations légales
- L’espace disponible

1.1.6) Rotation GFS (Grandfather – Father – Son)

Son = sauvegardes **journalières**
Souvent incrémentielles.

Father = sauvegardes **hebdomadaires**
Souvent des différentielles.

Grand Father = sauvegardes **mensuelles**
Souvent des Full.

Exemple de rotation :

- **Daily** (sons) : conserver 7 jours
- **Weekly** (fathers) : conserver 4 semaines
- **Monthly** (grandfathers) : conserver 12 mois
- Parfois **Yearly** : conserver 3 à 10 ans

Avantage :

Elle assure toujours :

- Des sauvegardes récentes
- Des sauvegardes stables

- Des points de restauration anciens

1.2) Gestion des scripts avant/après la sauvegarde (pre/post-job).

1.2.1) Script Pre-JOB

Le **script pré-job** est exécuté juste **avant** que la sauvegarde commence.

À quoi ça sert ?

À **préparer l'état du système** pour que la sauvegarde se déroule correctement.

Exemples d'utilisation :

- **Arrêter un service** avant de sauvegarder ses fichiers (ex : base de données, logiciel métier)
- **Mettre une base en mode « freeze »** (backup mode).
- **Vider un cache ou un dossier temporaire** pour réduire la taille de la sauvegarde.
- **Monter un lecteur réseau** avant de copier les données.
- **Débrancher une machine virtuelle d'un réseau** avant snapshot.
- **Envoyer un message dans un canal (ex : Slack/Discord)** pour dire « la sauvegarde commence ».

1.2.2) Script Post-Job

Le script post-job s'exécute une fois que la sauvegarde est terminée (que ce soit succès ou échec selon la configuration).

À quoi ça sert

À **remettre le système en état** ou effectuer des actions complémentaires.

Exemples d'utilisation :

- **Redémarrer un service** arrêté dans le pre-job.
- **Nettoyer les fichiers temporaires** créés pour la sauvegarde.
- **Envoyer une notification** pour dire que la sauvegarde a eu lieu(Discord, Slack,etc...)
- **Synchroniser le backup vers un stockage secondaire** (ex : copier sur NAS / cloud).
- **Remonter un volume chiffré** utilisé pour sauvegarder.
- **Relancer une opération de maintenance** (ex : ré indexation d'une base).

1.3) Paramétrage des jobs

Quand on crée un job de sauvegarde, la première étape est de **définir ce que l'on veut sauvegarder**.

Cela peut être :

- Un disque entier
- Des dossiers spécifiques
- Des fichiers précis
- Un partage réseau
- Les données d'un utilisateur
- Un volume logique ou un répertoire monté (GNU/Linux)

1.4) Rapport de sauvegarde automatisé et possibilité de rotation/conservation des logs.

Après chaque job de sauvegarde, la plupart des logiciels génèrent **automatiquement un rapport**.

Ce rapport sert à vérifier si **la sauvegarde s'est bien passée**, s'il y a eu des erreurs, et combien de données ont été sauvegardées.

Que contient généralement un rapport ?

1. **Nom du job**
2. **Heure de début**
3. **Heure de fin**
4. **Durée totale**
5. **Statut**
6. **Quantité de données sauvegardées**
7. **Vitesse de transfert**
8. **Erreurs détectées**
9. **Actions réalisées (pre/post-job)**
10. **Destination des sauvegardes**

Le but : **pouvoir contrôler automatiquement chaque sauvegarde**, sans aller à la main vérifier les dossiers.

Rotation / Conservation des logs :

Comme les sauvegardes tournent chaque jour, les logs augmentent très vite.
Pour éviter de remplir le disque avec des milliers de fichiers, les logiciels utilisent un système de rotation des logs.

Rotation des logs = suppression automatique des anciens rapports.

Elle peut être réglée selon :

- Durée

-Quantité

-Type de rotation (journalière, hebdomadaire, etc...)

1.5) Support de différents protocoles pour la cible de sauvegarde

1.5.1) RSYNC

Ce que c'est :

Un protocole Linux de synchronisation intelligent :
Un protocole GNU/Linux de synchronisation intelligente : il transfère uniquement les différences entre les fichiers

Avantages :

- Très rapide (envoie seulement ce qui change)
- Idéal pour les sauvegardes **incrémentielles**
- Très utilisé pour **Linux → NAS**
- Peut fonctionner via SSH pour être sécurisé

Inconvénients :

- Pas natif sous Windows
- Fonctionne surtout pour les fichiers (pas pour images disque)

1.5.2) SSH

Ce que c'est

Un protocole sécurisé utilisé pour se connecter à distance sur un serveur Linux.

Il chiffe tout : identifiants, données, transferts, etc...

Avantages

- Très sécurisé
- Permet d'exécuter des commandes (scripts pre/post-job)
- Base de SFTP et RSYNC sécurisé

Inconvénients

- Moins rapide que NFS/SMB en local
- Config nécessaire côté Linux

1.5.3) SFTP (SSH File Transfer Protocol)

Ce que c'est

Un protocole de transfert de fichiers basé sur SSH → **donc sécurisé**.

Avantages

- Sécurisé
- Facile à configurer
- Parfait pour les sauvegardes via Internet
- Grande compatibilité (Windows, Mac, Linux)

Inconvénients

- Un peu plus lent que RSYNC ou NFS
- Moins efficace pour les gros volumes

1.5.4) SMB

Ce que c'est

Le protocole de partage de fichiers **Windows**.
Utilisé pour accéder à un partage réseau du type :

\\NAS\Backup\

Avantages

- Très simple sous Windows
- Compatible avec tous les NAS

- Rapide en réseau local (LAN)

Inconvénients

- Pas chiffré par défaut → à éviter sur Internet
- Moins performant que NFS sous Linux

1.5.5) NFS (Network File System)

Ce que c'est

Le protocole de partage de fichiers **pour Linux / UNIX**.

Avantages

- Très rapide en LAN
- Très léger (moins de surcharge que SMB)
- Idéal pour les serveurs Linux

Inconvénients

- Pas sécurisé si utiliser sur Internet
- Plus complexe à configurer que SMB

1.6) Nécessité ou non d'installer un agent sur les postes à sauvegarder.

Qu'est-ce qu'un agent ?

C'est un **petit logiciel** installé directement sur chaque machine à sauvegarder (PC, serveur, VM).
Il permet au serveur de sauvegarde de communiquer plus finement avec le poste.

Avantages

1. **Meilleure performance**
2. **Sauvegarde cohérente des applications**
3. **Sauvegarde d'image système complète**
4. **Moins de dépendance au réseau**
5. **Gestion plus fine**

Inconvénients

- Installation sur chaque poste
- Maintenance (mises à jour de l'agent)
- L'agent consomme quelques ressources
- Moins adapté si on a des centaines de postes très divers sans gestion centralisée

-

1.7) Modes de transfert

1.7.1) Mode Push

Principe

C'est **la machine à sauvegarder** (la source) qui **pousse** elle-même ses données vers la destination (serveur de sauvegarde, NAS, etc.).

Exemple simple

Ton PC envoie ses fichiers vers un NAS via RSYNC ou SFTP.

Avantages

- Simple à mettre en place sur les petits environnements.
- Bonne maîtrise depuis le poste source.
- Moins de charge sur le serveur de sauvegarde.

Inconvénients

- Il faut souvent configurer une tâche sur chaque poste.
- Il faut ouvrir des accès réseau vers la destination (risque sécurité).
- Si beaucoup de postes existent, l'administration devient lourde.

1.7.2) Mode Pull (la destination récupère)

Principe

C'est **le serveur de sauvegarde** qui **vient chercher** les données directement sur les machines à sauvegarder.

Exemple simple

Le serveur lance un RSYNC vers le PC et copie les fichiers.

Avantages

- Centralisation totale : toute la logique est sur le serveur de sauvegarde.
- Plus facile à gérer à grande échelle (une seule configuration).
- Souvent plus sécurisé : les postes n'ont pas besoin d'avoir accès à la destination.

Inconvénients

- Le serveur doit avoir accès aux postes → parfois compliqué (pare-feu, droits...).
- Nécessite souvent l'installation d'un agent ou au moins un service accessible.
- Peut charger fortement le serveur si beaucoup de postes sont sauvegardés.

2) Recherche et sélection des outils

2.1) BorgBackup

Description / points forts : outil d'archivage dédupliquant., chiffrement intégré, compression, très efficace pour backups quotidiens.

Popularité **/ doc** : grande communauté Linux, doc complète sur ReadTheDocs.

Compatibilité cahier des charges :

- Méthodes : incrémentiel
- Pre/post : pas de « hooks » intégrés avancés mais on utilise couramment des wrappers / borgmatic pour pre/post jobs.
- Paramétrage des jobs
- Rapports/logs : sorties détaillées et possibilité de script pour rotation
- Protocoles : fonctionne sur dépôt local ou via SSH
- Agent : **non** (fonctionne en local ou via SSH ; on installe borg uniquement sur les sources qu'on veut sauvegarder).
- Mode : **push** typique, mais peut pull via SSH.

2.2) Bacula

Description / points forts :gère grandes infrastructures, catalogues, bandes/disques, planification avancée.

Popularité / doc : très répandu en entreprise, doc et manuels complets ; édition Enterprise disponible.

Compatibilité :

- Méthodes : full/diff/incr (planification flexible).
- Pre/post : supporte scripts/commandes pré/post job via directives (très complet).
- Paramétrage des jobs Include/exclude : fine granularité (filesets, patterns).
- Rapports/logs : reporting riche, rotation et archivage configurables (catalogue & logs).
- Protocoles : client Bacula sur postes (TCP), stockage sur disque/tape/NFS/SMB selon configuration.

- Agent : **oui** .architecture client/serveur.
- Mode : **pull** classique mais flexible.

2.3) UrBackup

Description / points forts : client/serveur facile à déployer, combine image disque (Windows) + sauvegarde de fichiers.

Popularité / doc : populaire pour petites/ moyennes infra ; doc d'administration disponible.

Compatibilité :

- Méthodes : fichiers incrémentiels rapides + image
- Pre/post : client/serveur permet actions pré/post via scripts (selon version).
- Paramétrage des jobs : sélection de dossiers, filtres, exclusions.
- Rapports/logs : interface web montre l'état ; logs stockés côté serveur .
- Protocoles : client dédié communique avec serveur UrBackup.
- Agent : **oui** sur poste client
- Mode : **pull**. Architecture client/serveur.

2.4)Rclone

- **Description / points forts** : « rsync pour le cloud » — sync/copy vers 70+ backends ,très scriptable.
- **Popularité / doc** : très populaire pour migration & sauvegarde cloud ; doc officielle complète. Rclone+1
- **Compatibilité** :
    - Méthodes : sync / copy / move ; peut implémenter incrémentaux via --backup-dir
    - Pre/post : scriptable (cron/systemd) → pre/post via wrapper.
    - Paramétrage des jobs : filtres et patterns très puissants.
    - Rapports/logs : sorties, logs, options de verbosité ; rotation via --backup-dir ou wrappers + logrotate.
    - Protocoles : prise en charge native de très nombreux backends (SFTP, S3, WebDAV, etc.).
    - Agent : **non**
    - Mode : **push** (source envoie) ou **pull** (serveur peut lancer rclone vers sources si accessibles).

2.5) Amanda / Zmanda

- **Description / points forts** : système réseau client-serveur historique (AMANDA), peut sauvegarder vers disque/tape/cloud.
- **Popularité / doc** : utilisé en milieu universitaire/PME, docs et guides (Zmanda fournit support pro).
- **Compatibilité** :
    - Méthodes : full/incrémentielle archive vers formats traditionnels.
    - Pre/post : gestion via scripts et configuration
    - Paramétrage des jobs : règles d'exclusion.
    - Rapports/logs : login et rotation paramétrables (selon install + logrotate).
    - Protocoles : utilise ssh/rsync/tape backends selon config.
    - Agent : **client/server** (agent léger sur hôtes).
    - Mode : **pull** (serveur orchestre) typique.

2.6) Duplicity

- **Description / points forts** : client open-source orienté utilisateur, sauvegardes chiffrées, beaucoup de backends cloud
- **Popularité / doc** : communauté active sur GitHub ; facile à prendre en main pour postes & petits serveurs
- **Compatibilité** :

    -Méthodes : incrémental, compression, chiffrement.

    -Pre/post : on peut appeler scripts externes depuis système (wrappers) ; pas de hook natif complexe.

    -Include/exclude : patterns et filtres.

    - rapports/logs : logs et notifications ; rotation via config et scripts.

    - Protocol: SFTP, WebDAV, S3, Azure, many cloud providers.

    - Agent : **non** (client installé sur poste qui pousse les données).

    -Mode : **push** (client envoie).
- **Remarque** : idéal pour postes/PME avec destinations cloud grand public.

2.7) Sélection des 3 solutions qui me semble les plus pertinentes

BACULA :

Cet outil me semble pertinents pour les moyennes ou grandes entreprises, il est utile pour gère les grandes infrastructures,

Duplicity :

Ducplicity quand à lui est pratique pour les petites entreprises, il est léger et simple à utilisé

UrBackup :

Très facile à déployé avec une interface web moderne facile à utilisa et util pour les PME

3)Analyse détaillée par critères

Solutions	BACULA	DUPLICITY	URBACKUP
Méthodes de sauvegarde prises en charge	**FULL, Différentielle, incrémentielle**	**Full initiale + incrémentiel sur la durée**	**FULL, incrémentielle**
La conformité à la règle 3-2-1, duré de rétention, configurable, rotation GFS.	**Conforme à la règle 3-2-1, et Excellent support GFS**	**Conforme à la règle 3-2-1, Duré de rétention oui. Mais pas de rotation GFS natif**	**Excellent support GFS, Durée de rétention bien intégrée et rotation ZFS partiellement supporté**
Gestion des scripts pre/post-job	**Support exellent**	**Duplicity n'intègre pas nativement un système pre/post-job**	**Oui — intégré dans l'interface**
Paramétrage des fichiers/dossiers à inclure/exclure	**Excellent — très complet et avancé**	**Très bon — simple et efficace**	**Très bon - simple**
Génération et rotation des rapports/logs	**Excellent — complet et professionnel**	**Bon — via stdout, logs externes et scripts**	**Très bon — intégré et visualisable via l'interface web**

Protocoles supportés pour la cible	**NFS en natif , SSH/SFTP / RSYNC disponible via script et SMB disponbile via montage Samba**	**SSH/SFTP/RSYNC/FTP disponible nativement NFS et SMB disponible si monté localement**	**NFS/SMB = natif mais SSH/SFTP/RSYNC non disponible**
Agent	**Agent requis**	**Agentless**	**Agent requis**
Mode de transfert	**PULL uniquement**	**PUSH uniquement**	**Principalement PULL mais peut faire du PUSH**
Performances et optimisation	**Excellente gestion débit avec le Storage daemons Compression et excellent chiffrement. Mais la Déduplication n'est pas intégrée nativement**	**On peut avoir de la compression et des chiffrements mais pas de déduplication et de gestion de débit**	**La gestion de débit, compressions, chiffrement et déduplication sont possible**
Gestion de la restauration en analysant la granularité de la restauration	**Fichier individuel Oui. Répertoires complets : Oui, Système complet / bare-metal : Oui, Bases de données applicatives : Oui**	**Fichier individuel Oui. Répertoires complets : Oui Système complet : non natif Bases de données applicatives : nécésite un dump préalable (exemple MySQL dump)**	**Fichier individuel Oui. Répertoires complets : Oui, Système complet: Oui, Bases de données applicatives : non**

Tolérance aux erreurs et reprise sur incident	**Bacula est très robuste pour les environnements professionnels : reprise fiable + alertes complètes.**	**Duplicity est moins tolérant aux interruptions, surtout sur de gros fichiers ou sur réseau instable.**	**UrBackup est très tolérant aux erreurs sur LAN et WAN.**
Journalisation et traçabilité	**Bacula est idéal pour les environnements professionnels**	**Duplicity est suffisant pour des besoins simples.**	**UrBackup offre une traçabilité pratique et accessible**
Conformité au RGPD	**Chiffrement en transit : oui Chiffrement au repos : Oui**	**Chiffrement en transit : oui Chiffrement au repos : Oui**	**Chiffrement en transit :oui Chiffrement au repos : Oui mais optionel**
Gestion des sauvegardes distantes ou hybrides, analyse de l'intégration avec des stockages cloud et compatibilité multi-sites.	**Sauvegardes distantes / Multi-sites, Intégration Cloud, Compatibilité OVH Cloud**	**Sauvegardes distantes / Multi-sites Intégration Cloud. Natif en S3. Intégration Cloud**	**Sauvegardes distantes / Multi-sites, Intégration Cloud**

Gestion des utilisateurs de la plateforme de sauvegarde, intégration LDAP/Active Directory, politiques de permissions.	**Gestion des utilisateurs : Très granulaire, très complet Intégration LDAP : Compatible entreprises multisites Politiques de permissions : Politique de segmentations, Possibilité d'empêcher un opérateur de lancer certains jobs**	**Gestion des utilisateurs : Non Intégration LDAP : Non Politique de permissions : aucune politique de permissions**	**Gestion des utilisateurs : Oui Intégration LDAP : pas nativement Politique de permissions : Quelques rôles préconfigurés, correcte pour une PME**
Niveau d'intégration avec des outils de supervision, possibilité d'alerter via mail, Slack, Zabbix, Prometheus, etc.	**possibilité d'alerter via mail: Oui Slack: pas natif mais possible. Zabix : intégration exellente Prometheus: Très bon niveau**	**Possibilité d'alerter via mail: non natif mais possible Slack: possible uniquement via script Zabix : pas natif mais possible Prometheus: pas d'export officielle**	**Possibilité d'alerter via mail: Oui Slack: pas natif mais facile à mettre en place Zabix : intégration exellente Prometheus: oui**

Automatisation et planification, gestionnaire de tâche intégré, gestion des dépendances entre jobs, mise en file d'attente.	**Planification nativement intégré, Gestion des dépendances entre jobs: oui mise en file d'attente : oui système professionnelle complet Automatisation avancée:Bonne**	**planification nativement intégré, Gestion des dépendances entre jobs: pas de gestion native mise en file d'attente : Pas de file d'attente système professionnelle complet Automatisation avancée: possible mais limité**	**planification nativement intégré, Gestion des dépendances entre jobs: non mise en file d'attente :oui système professionnelle complet Automatisation avancée:Bonne**
Scalabilité de la solution	**Bacula est conçu dès l'origine pour les grandes infrastructures.**	**Pas optimal pour grandes entreprises.**	**Scalable pour structures multi-agences, mais pas datacenters.**
Facilité de prise en main	**Complexe à prendre en main.**	**Facile à prendre en main**	**très facile à prendre en main**
Documentation et support communautaire	**Bonne documentation**	**Documentation complète**	**Bonne documentation, communauté active**
Courbe d'apprentissage	**Courbe d'apprentissage raide**	**Courbe d'apprentissage faible à moyenne**	**Courbe d'apprentissage très faible**
Intégrabilité dans l'infrastructure existante	**Très intégrable dans les infrastructures existantes**	**Très scriptable et automatisable, idéal pour serveurs individuels ou petites infrastructures**	**Bonne intégrabilité pour PME et petites infrastructures**

4) **Synthèse et préconisations**

4.1) la solution la plus adaptée au contexte

Selon moi la solution la plus adapté au contexte est URBACKUP. Il est Idéal pour les PME, écoles, TPE, ou environnements multipostes de taille moyenne.

Une Solution simple à déployer et à maintenir, robuste et sécurisée pour des volumes moyens.

Il a de grands avantages :

- **Une Interface web moderne et intuitive : très facile à prendre en main.**
- **Une Automatisation intégrée : planification simple, pre/post-job.**
- **Déduplication et optimisation : réduit l'espace utilisé et le temps de sauvegarde.**
- **Bonne intégration supervision : Zabbix, Prométheums, notifications email.**
- **Gestion multi-clients possible jusqu'à 200–1000 clients selon architecture.**

Mais il à également certains inconvénients :

Dépendances entre jobs limitées : pas adapté à orchestrations complexes multisites.

Moins scalable que Bacula pour très grandes infrastructures (>1000 clients ou plusieurs centaines de To).

Intégration LDAP/AD non native (possible via contournement).

4.2) indice de pondération

Critère	Bacula	UrBackup	Duplicity
Méthodes / GFS	5/5	4/5	3/5
Automatisation	5/5	4/5	3/5
Paramétrage fichiers	5/5	4/5	3/5
Logs / Reports	5/5	4/5	2/5
Protocoles	5/5	4/5	4/5
Agent	4/5	5/5	5/5
Push/Pull	5/5	4/5	5/5
Performance / Chiffrement	5/5	4/5	3/5
Restauration	5/5	4/5	2/5
Supervision	5/5	4/5	2/5
Scalabilité	5/5	4/5	2/5
Prise en main	1/5	5/5	4/5
Total	50/55	45/55	30/55

4.3) Dans quels cas chaque solution est préférable :

Bacula :

- Bacula est préférable lorsque notre infrastructure est grande, structurée ou multisites

-lorsqu'on a des exigences avancées

-Quand on a besoin d'automatisation complexe

-Quand on exige une supervision et une traçabilité avancées

UrBackup :

-UrBackup est préférable lorsqu'on souhaite une solution simple, centralisée et efficace

-Notre contexte est une PME, une école ou un service informatique municipal

-lorsque l'on privilégie le confort d'utilisation

-Quand notre budget est limité

Duplicity :

-Duplicity est préférable quand nous avons besoin de sauvegarder un petit nombre de serveurs ou postes

-Lorsqu'on veut une solution scriptable et très léger

-quand les volumes de données sont modestes

-quand notre priorité est la simplicité et le coût minimal

Liens :

https://www.malekal.com/meilleurs-logiciels-sauvegarde-linux/

https://fr.ubunlog.com/serveur-client-du-syst%C3%A8me-de-sauvegarde-urbackup/

https://www.urbackup.org/administration_manual.html

https://doc.ubuntu-fr.org/duplicity

https://doc.ubuntu-fr.org/bacula

https://www.bacula.org/documentation/documentation/