opentext™

OpenText™ eDirectory CE 24.4 (v9.3)

Guide d'optimisation

Janvier 2025

Mentions légales

Copyright 2009 - 2024 Open Text Tous droits réservés.

Les seules garanties pour les produits et services d'Open Text et de ses filiales et concédants de licence (« Open Text ») sont énoncées dans les déclarations de garantie expresses accompagnant ces produits et services. Aucun élément du présent document ne doit être interprété comme constituant une garantie supplémentaire. Open Text ne pourra pas être tenu responsable des erreurs techniques ou éditoriales ou des omissions contenues dans le présent document. Les informations contenues dans le présent document sont susceptibles d'être modifiées sans préavis.

Table des matières

À propos de ce guide et de la bibliothèque 5

1 Présentation 7

Conditions préalables . . . 7

2 Sous-systèmes OpenText eDirectory 9

Base de données FLAIM . . . 9
- Point de contrôle . . . 10
- Index . . . 10
- Fichier journal de transaction individuelle . . . 11
- Mise en conteneur des attributs FLAIM . . . 11

Réserve de threads . . . 12

3 Analyse des goulots d'étranglement du système 15

Sous-système d'E/S disque . . . 15
Sous-système du processeur . . . 16
Sous-système de mémoire . . . 17
Sous-système réseau . . . 17

4 Optimisation des sous-systèmes OpenText eDirectory 19

Base de données FLAIM . . . 19
- Choix des index . . . 20
- Optimisation des mises à jour . . . 21

Réserve de threads . . . 21
ACL . . . 21
- Amélioration des recherches et des lectures d'OpenText eDirectory . . . 22
- Désactivation des modèles ACL . . . 23

Réplication . . . 25
Disque SSD . . . 26
Intervalle de mise à jour à la connexion NMAS . . . 26
Surcharge d'information SSL . . . 26
Utilitaire ICE (Import Convert and Export) . . . 26
ldif2dib . . . 26
Amélioration de la prise en charge de la taille de paquet par NCP . . . 27

5 Configuration d'OpenText eDirectory 29

Configuration du sous-système FLAIM . . . 29
- Limite de cache stricte . . . 29
- Ajustement dynamique de la limite . . . 29

Modification des paramètres de cache FLAIM . . . 30
- Modification des paramètres de cache FLAIM par l'intermédiaire d'OpenText iMonitor . . . 30

Modification des paramètres de cache FLAIM dans _ndsdb.ini 31

À propos de ce guide et de la bibliothèque

Ce guide indique comment analyser et ajuster le produit OpenText™ eDirectory CE 24.4 (v9.3) afin d'obtenir de meilleures performances dans tous les déploiements.

Pour obtenir la dernière version du présent *Guide d'optimisation d'OpenText™ eDirectory*, consultez le site Web de documentation en ligne d'OpenText eDirectory (https://www.netiq.com/documentation/edirectory-93/).

Public

Le guide est destiné aux administrateurs réseau.

Autres documents dans la bibliothèque

La bibliothèque propose les manuels suivants :

XDASv2 Administration Guide (Guide d'administration de XDASv2)

Décrit comment configurer et utiliser XDASv2 afin d'auditer OpenText eDirectory et OpenText Identity Manager.

Guide d'installation

Décrit comment installer OpenText eDirectory. Il est destiné aux administrateurs réseau.

Guide d'administration

Décrit comment gérer et configurer OpenText eDirectory.

Guide de dépannage

Décrit comment résoudre les problèmes d'OpenText eDirectory.

Guide des nouveautés

Décrit les nouvelles fonctionnalités d'OpenText eDirectory.

Ces guides sont disponibles sur le site Web de documentation d'OpenText eDirectory (https://www.netiq.com/documentation/edirectory-93/).

Pour plus d'informations sur l'utilitaire de gestion d'eDirectory, reportez-vous au Guide d'administration d'OpenText Identity Console (https://www.netiq.com/documentation/identity-console/).

1 Présentation

OpenText eDirectory 9.3 est une solution hautes performances de services d'annuaire conforme aux normes, multiplateforme, hautement évolutive et à tolérance de pannes. Ce guide fournit des informations sur l'optimisation de votre environnement OpenText eDirectory afin d'en améliorer les performances.

L'optimisation des performances est une opération complexe. Elle nécessite de bien comprendre les sous-systèmes de OpenText eDirectory et du système d'exploitation. Elle implique également de surveiller le système afin d'identifier les goulots d'étranglement et de les corriger individuellement. Il arrive fréquemment que les ressources soient limitées et que l'optimisation se limite à OpenText eDirectory et au système d'exploitation.

Commencez par lire la section Conditions préalables de ce guide, essayez d'effectuer une optimisation, puis passez aux autres sections. Le chapitre Sous-systèmes OpenText eDirectory décrit les sous-systèmes principaux qui influencent les performances d'OpenText eDirectory. Le chapitre Analyse des goulots d'étranglement du système décrit les différentes ressources système et leur influence sur les performances d'OpenText eDirectory. Le chapitre Optimisation des sous-systèmes OpenText eDirectory explique comment analyser et ajuster OpenText eDirectory dans différents environnements et conditions. Enfin, le chapitre Configuration d'OpenText eDirectory explique comment configurer les différents paramètres ajustables.

Conditions préalables

Assurez-vous que les conditions préalables suivantes sont remplies avant de tenter d'ajuster le système pour améliorer ses performances :

- Une conception d'arborescence OpenText eDirectory correcte peut améliorer les performances d'OpenText eDirectory. Les considérations suivantes peuvent s'appliquer :
  - Les applications lisent toutes les informations localement, sur le serveur, sans qu'il soit nécessaire de chaîner les requêtes.
  - OpenText eDirectory gère automatiquement les références d'objet avec efficacité. Si c'est possible, les objets d'un serveur ne doivent pas faire référence à d'autres objets qui ne figurent pas localement sur ce serveur, car la maintenance des références d'objets non locaux peut être plus longue. Si de telles références sont présentes, il est nécessaire de maintenir les liens en amont. Cette opération devient fastidieuse dans les déploiements à grande échelle.
  - Si vous avez besoin d'un groupe d'au moins 10 000 membres, il est recommandé d'utiliser des groupes dynamiques. Cela permet d'éviter la surcharge d'information associée à la maintenance des références pour de telles quantités d'utilisateurs. Choisissez la configuration des groupes dynamiques avec précaution, car l'utilisation de plusieurs groupes dynamiques dont les critères de recherche sont incorrects peut surcharger le serveur et réduire ses performances globales. Si une opération de recherche est très longue, cela peut signifier que l'index choisi n'est pas efficace. Réduisez les groupes standard (statiques) utilisés, car ils peuvent augmenter la navigation dans l'arborescence lors de la connexion.

- Utilisez les listes ACL efficacement. Utilisez par exemple l'ayant droit [Ceci] et assignez-le au niveau du conteneur au lieu d'utiliser un modèle d'ACL qui s'attribue des droits à lui-même. Plus le nombre d'ACL est faible, meilleures sont les performances. Pour plus d'informations sur les ACL, reportez-vous à la section « Droits OpenText eDirectory » du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*.
- Répartissez la charge sur plusieurs serveurs de répliques.
- Une arborescence bien conçue réduit la navigation dans l'arborescence, mais cette opération reste encore parfois nécessaire. Vous pouvez prendre en compte le « coût de renvoi avancé » du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*.
- Si les connexions sont lentes, vous pouvez désactiver les mises à jour à la connexion. Il existe différentes méthodes pour désactiver les mises à jour à la connexion pour les connexions NDS et NMAS (NetIQ Modular Authentication Service). Il est toutefois important de bien en comprendre les implications en matière de sécurité (http://www.novell.com/documentation/nmas33/admin/data/bg8dphs.html).

- Vérifiez l'état de santé par le biais de iMonitor. Pour plus d'informations, reportez-vous à la section « Affichage de l'état de santé du serveur OpenText » du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*. Vérifiez ce qui suit :
  - L'heure est synchronisée sur tous les serveurs de répliques.
  - La santé des processus en arrière-plan et de synchronisation des répliques est bonne.

2 Sous-systèmes OpenText eDirectory

Cette section traite des sous-systèmes OpenText eDirectory.

- « Base de données FLAIM » page 9
- « Réserve de threads » page 12

Base de données FLAIM

OpenText eDirectory utilise la base de données FLAIM. La base de données FLAIM (Flexible Adaptable Information Manager) est utilisée pour les informations traditionnelles, volatiles et complexes. Ce moteur de base de données très évolutif prend en charge plusieurs opérations de lecture et un modèle de simultanéité à une seule opération d'écriture. Les opérations de lecture ne bloquent pas les opérations d'écriture et, réciproquement, les opérations d'écriture ne bloquent pas les opérations de lecture.

Physiquement, le gestionnaire FLAIM organise les données en blocs. Certains de ces blocs sont généralement conservés en mémoire. Ils représentent le cache de blocs. Quant au cache d'entrées (parfois appelé cache d'enregistrements), il met en cache des entrées logiques de la base de données. Les entrées sont construites à partir des éléments du cache de blocs. Le gestionnaire FLAIM conserve les tables de hachage des deux caches. La taille du compartiment de hachage est ajustée périodiquement en fonction du nombre d'éléments.

Par défaut, OpenText eDirectory utilise une taille de blocs de 4 Ko. Pour la mise en cache de l'intégralité de la base de données DIB, la taille du cache de blocs doit être égale à celle du fichier DIB ; la taille du cache d'entrées doit être deux à quatre fois supérieure à celle du fichier DIB.

Lors de la récupération d'une entrée, le gestionnaire FLAIM recherche d'abord l'entrée dans le cache d'entrées. Si l'entrée existe, la lecture à partir du cache de blocs n'est pas nécessaire. Lors de la récupération d'un bloc à partir du disque, le gestionnaire FLAIM recherche d'abord le bloc dans le cache. Si le bloc existe, la lecture du disque n'est pas nécessaire.

Lors de l'ajout ou de la modification d'une entrée, les blocs correspondant à cette entrée ne sont pas validés directement sur le disque, si bien que le disque et la mémoire peuvent ne pas être synchronisés. Toutefois, les mises à jour apportées à l'entrée sont consignées dans le fichier journal de transaction individuelle. Ce fichier journal permet de récupérer des transactions après une défaillance du système.

L'algorithme de remplacement LRU (Least Recently Used) est utilisé pour remplacer des éléments du cache.

- « Point de contrôle » page 10
- « Index » page 10
- « Fichier journal de transaction individuelle » page 11
- « Mise en conteneur des attributs FLAIM » page 11

Point de contrôle

Un point de contrôle adapte la version de la base de données sur disque à l'état de la base de données en mémoire (en cache). Le gestionnaire FLAIM peut réaliser un point de contrôle pendant l'activité de mise à jour minimale de la base de données. Il s'exécute toutes les secondes et écrit les blocs altérés (cache altéré) sur le disque. Les blocs qui sont modifiés dans le cache, mais ne sont pas encore écrits sur le disque, sont appelés « blocs altérés ». Le gestionnaire FLAIM acquiert un verrou sur la base de données et exécute la quantité maximale de travail possible jusqu'à ce que le point de contrôle se termine ou qu'un autre thread soit en attente pour mettre à jour la base de données. Afin d'éviter une trop forte désynchronisation de la base de données sur le disque, un point de contrôle est forcé dans certaines conditions, même si des threads attendent pour mettre à jour la base de données :

- Si le thread de point de contrôle ne peut pas terminer un point de contrôle dans un intervalle de temps spécifié (la valeur par défaut est de 3 minutes), il est forcé et le cache altéré est nettoyé.
- Si la taille du cache altéré est supérieure à la valeur `maxdirtycache` (si elle est définie), un point de contrôle est forcé afin de réduire la taille du cache altéré à la valeur `mindirtycache` (si elle est définie) ou à zéro.

Index

Un index est un ensemble de clés organisées de manière à accélérer considérablement la recherche de clés spécifiques. Les clés d'index sont construites par extraction du contenu d'un ou de plusieurs champs (attributs) à partir des entrées. Les index sont conservés dans le cache de blocs. Toute modification des attributs indexés nécessite la modification des blocs d'index.

OpenText eDirectory définit un ensemble par défaut d'index pour les attributs (champs) système. Les attributs système tels que `parentID` et `ancestorID` sont utilisés pour les recherches à un seul niveau et dans les sous-arborescences. Ces index ne peuvent être ni suspendus ni supprimés. L'annuaire les utilise en interne. Les index par défaut sont définis pour les attributs tels que CN, `Surname`, `Given Name`, etc. Les index peuvent être de type Presence, Value et Substring. Ces index peuvent être suspendus. Lors d'une suppression, ils sont recréés automatiquement.

Vous pouvez utiliser OpenText Identity Console ou l'utilitaire LDAP (Lightweight Directory Access Protocol) ndsindex pour créer des index. Les index (http://www.netiq.com/documentation/edir83/) sont propres au serveur.

En activant la balise du Gestionnaire de stockage (`StrMan`) dans DSTrace (ndstrace), vous pouvez afficher l'index choisi pour les recherches.

L'exemple suivant concerne le journal DSTrace d'une recherche dans une sous-arborescence à l'aide de `"cn=admin"`, CN.

```
3019918240 StrMan: Iter #b239c18 query ((Flags&1)==1) &&
((CN$217A$.Flags&8=="admin") && (AncestorID==32821))

3019918240 StrMan: Iter #b239c18 index = CN$IX$220
```

L'exemple suivant concerne le journal DSTrace d'une recherche dans une sous-arborescence à l'aide de `"Description= This is for testing"` ("Description= Ceci est un test"), `AncestorID`.

```
2902035360 StrMan: Iter #83075b0 query ((Flags&1)==1) &&
((Description$225A$.Flags&8=="This is for testing") &&
(AncestorID==32821))
```

```
2902035360 StrMan: Iter #83075b0 index = AncestorID_IX
```

Pour améliorer les performances de recherche à l'aide du tri côté serveur, utilisez l'option -a pour ajouter l'attribut `AncestorID` en préfixe à la liste des attributs transmis lors de la création d'un index.

Le tableau ci-dessous explique la tendance à augmenter la taille de la DIB lors de la création d'index simples/composés pour un certain nombre d'objets. Nous vous recommandons de vous reporter à ces données de test lors de la création d'index simples/composés :

Nombre d'objets	Augmentation de la taille de la DIB avec un index simple contenant une taille d'attribut de 20 caractères	Augmentation de la taille de la DIB avec un index simple contenant une taille d'attribut de 200 caractères	Augmentation de la taille de la DIB avec un index composé contenant une taille d'attribut de 200 caractères
23 000	24 Ko	3 Mo	7 Mo
47 000	50 Ko	6 Mo	15 Mo
0.5 millions	4 Mo	55 Mo	100 Mo
1 million	9 Mo	111 Mo	210 Mo

Fichier journal de transaction individuelle

Le gestionnaire FLAIM enregistre les opérations de chaque transaction de mise à jour dans un fichier journal de transaction individuelle (RFL). Le fichier RFL permet de récupérer les transactions en cas de défaillance du système ou lors de la restauration à partir d'une sauvegarde. Ce fichier est tronqué à la fin de chaque point de contrôle, sauf s'il est activé (`rflkeepfiles`) à l'aide d'une sauvegarde à chaud et continue (http://www.netiq.com/documentation/edir93/).

Mise en conteneur des attributs FLAIM

Pour garantir une utilisation optimale du cache d'entrée ainsi qu'une amélioration des performances des opérations de recherche d'attributs, FLAIM enregistre les attributs dont les valeurs sont plus élevées ou contenant un nombre supérieur de valeurs à un autre emplacement, à savoir, le conteneur d'attributs. Par défaut, les attributs seront déplacés automatiquement vers le conteneur lorsque l'attribut :

- contient plus de 25 valeurs ;
- sa valeur est supérieure à 2 048 octets.

Pour désactiver la mise en conteneur automatique d'attributs, ajoutez `disablemovetoattrcontainer =1` au fichier `_ndsdb.ini`, puis redémarrez OpenText eDirectory.

OpenText eDirectory vous offre la possibilité de planifier le déplacement des attributs. Vous voyez d'abord s'afficher les attributs prêts à être déplacés, puis planifiez leur transfert à votre convenance.

Pour afficher le nombre d'attributs prêts à être déplacés vers les conteneurs d'attributs, exécutez la commande `ndscheck`. Pour afficher les détails d'attributs, utilisez l'attribut **dsContainerReadyAttrs** iMonitor sur les objets serveur Pseudo dans **Configuration de l'agent**. L'utilisateur peut également trouver ces attributs qui sont marqués pour indexation dans État de santé de l'agent.

Vous pouvez démarrer la mise en conteneur des attributs à l'aide de l'option Réparation d'objet unique de `ndsrepair` pour l'objet Pseudo serveur. Pour placer un attribut dans un conteneur, entrez la commande `ndsrepair` avec le nouveau paramètre avancé `-am` suivi du nom de l'attribut comme indiqué ci-dessous :

```
ndsrepair –J <Pseudo server object ID> –Ad –AM/–am <attribute name>
```

Après avoir déplacé un attribut dans le conteneur d'attributs, OpenText eDirectory crée un index système avec le nom de l'attribut. Lorsqu'un attribut est mis en conteneur, vous ne pouvez pas le replacer dans son conteneur d'origine.

REMARQUE : si un attribut a une valeur supérieure à 2 048 octets, la mise en conteneur s'effectue toujours, mais OpenText eDirectory ne crée pas d'index système.

Réserve de threads

Pour des raisons de performances, OpenText eDirectory est en mode multithread. Dans ce mode, lorsque le système est occupé, d'autres threads sont créés pour gérer la charge, tandis que certains threads sont arrêtés afin d'éviter toute surcharge d'information supplémentaire. La création et la destruction fréquentes de threads sont à la fois coûteuses et inefficaces. Au lieu de générer dynamiquement de nouveaux threads et de les détruire à chaque tâche, un certain nombre de threads sont démarrés et placés dans une réserve. En fonction des besoins, le système alloue les threads de la réserve à plusieurs tâches. Les tâches sont stockées dans deux types de files d'attente :

- Les tâches qui doivent être planifiées immédiatement sont placées dans la file d'attente des éléments prêts.
- Les tâches qui doivent être planifiées ultérieurement sont placées dans la file d'attente des éléments en attente.

Tous les modules n'utilisent pas la réserve de threads. Le nombre réel de threads du processus est supérieur au nombre figurant dans la réserve de threads. Par exemple, le gestionnaire FLAIM gère les threads d'arrière-plan séparément.

L'exécution de la commande `ndstrace -c threads` renvoie les statistiques de réserve de threads suivantes :

- le nombre total de threads générés dynamiquement, arrêtés et inactifs ;
- le nombre total de threads de travail utilisés actuellement et le nombre maximal de threads de travail ;
- le nombre de tâches et le nombre maximal de tâches dans la file d'attente des éléments prêts ;
- le nombre minimal, maximal et moyen de microsecondes passées dans la file d'attente des éléments prêts ;
- le nombre actuel et maximal de tâches dans la file d'attente des éléments en attente.

Voici un exemple de réserve de threads :

```
Thread Pool Information
Summary      : Spawned 42, Died 5
Pool Workers : Idle 8, Total 37, Peak 37
Ready Work   : Current 0, Peak 10, maxWait 67436 us
Sched delay  : Min 14 us, Max 1052004 us, Avg: 792 us
Waiting Work : Current 17, Peak 21
```

Il existe des paramètres de réserve de threads :

- **n4u.server.max-threads:** nombre maximal de threads qui peuvent être disponibles dans la réserve.
- **n4u.server.idle-threads:** nombre maximal de threads inactifs qui peuvent être disponibles dans la réserve.
- **n4u.server.start-threads:** nombre de threads démarrés.

Exécutez les commandes `ndsconfig get` et `ndsconfig set` pour obtenir et définir la taille de la réserve de threads.

3 Analyse des goulots d'étranglement du système

Plusieurs ressources système influencent les performances d'OpenText eDirectory. En outre, la mise à niveau vers la dernière version du système d'exploitation améliore les performances.

- « Sous-système d'E/S disque » page 15
- « Sous-système du processeur » page 16
- « Sous-système de mémoire » page 17
- « Sous-système réseau » page 17

Sous-système d'E/S disque

Le sous-système de disque est le goulot d'étranglement le plus courant. Les E/S restent pendant une période relativement prolongée dans les longues files d'attente, ce qui entraîne une forte utilisation des disques et une inactivité des cycles de processeur. Utilisez l'outil iostat pendant les périodes de charge maximale prévue pour déterminer les indicateurs de temps de réponse moyen.

Les opérations de lecture, d'écriture et de mise à jour sur le disque peuvent être séquentielles ou aléatoires. Les lectures et les mises à jour aléatoires constituent le modèle d'accès le plus courant dans les déploiements OpenText eDirectory.

Voici quelques solutions pour les workloads aléatoires :

- Augmentez la mémoire RAM. Cela permet de mettre en cache les données utilisées fréquemment ou faisant l'objet d'une lecture anticipée au niveau de la couche du système de fichiers. Le fichier DIB peut également être mis en cache dans le sous-système FLAIM.
- Utilisez des volumes dédiés pour le fichier DIB. Les performances du système de fichiers sont améliorées pour les volumes créés près de l'axe. Utilisez des volumes dédiés pour les journaux RFL et d'autres journaux.
- Les disques présentent des temps de latence toujours plus importants en raison de leur fragmentation. Ils doivent par conséquent être défragmentés.
- Ajoutez d'autres lecteurs pour les journaux RFL FLAIM. Ce type de consignation peut être effectué sur des disques à grande vitesse.
- Utilisez un environnement RAID 10(1+0) avec un plus grand nombre de disques durs.

Les fichiers créés par OpenText eDirectory peuvent atteindre 4 Go. Les systèmes de fichiers optimisés pour gérer des fichiers volumineux sont efficaces avec OpenText eDirectory.

- Pour Solaris™, le système de fichiers VxFS Veritas* est un système de fichiers basé sur les extensions et dans lequel les métadonnées sont optimisées pour les fichiers volumineux. Le système de fichiers UFS est indirectement basé sur les blocs, car les métadonnées du système de fichiers sont stockées dans un grand nombre de blocs. Elles peuvent même être découpées dans le cas de fichiers volumineux, ce qui ralentit l'exécution d'UFS pour ces fichiers.
- Pour Linux™, le système de fichiers Reiser est un système de fichiers de journalisation rapide, plus performant que le système de fichiers ext3 sur les ensembles DIB volumineux. Toutefois, le mode de journalisation d'écriture différée du système ext3 est réputé pour s'adapter aux performances du système de fichiers Reiser, même si le mode de classement par défaut fournit une meilleure cohérence des données. Le système de fichiers de journalisation XFS hautes performances peut gérer les fichiers volumineux et proposer des transferts de données plus fluides. OpenText eDirectory 9.3 est pris en charge sur les plates-formes SLES 15 32 bits et 64 bits avec système de fichiers XFS.

Le gestionnaire FLAIM prend en charge les blocs de 4 Ko et 8 Ko. La taille par défaut est de 4 Ko. C'est la taille par défaut des blocs sous Linux (`tune2fs -l device`). En revanche, sur Solaris, le système de fichiers UFS est créé avec une taille de blocs par défaut de 8 Ko (`df -g mountpoint`). Si la taille des blocs FLAIM est inférieure à celle des blocs du système de fichiers, des blocs peuvent être écrits partiellement. Si la taille des blocs de base de données est supérieure à celle des blocs du système de fichiers, chaque lecture et écriture de bloc est divisée en une série d'opérations d'E/S physiques distinctes. Par conséquent, vous devez faire en sorte que la taille des blocs FLAIM soit toujours identique à celle des blocs du système de fichiers.

La taille des blocs ne peut être contrôlée que lors de la création du fichier DIB. Ajoutez une ligne « blocksize = 8192 » au fichier `_ndsdb.ini` pour créer le fichier DIB avec une taille de blocs de 8 Ko.

Le choix de la taille des blocs dépend de la taille moyenne des enregistrements FLAIM dans les déploiements. Il est nécessaire d'effectuer des tests empiriques sur l'ensemble de données de test adéquat afin de déterminer la taille idéale des blocs dans votre déploiement.

Sous-système du processeur

OpenText eDirectory est construit sur une architecture fortement évolutive. L'augmentation du nombre de processeurs améliore les performances. On observe une amélioration du débit jusqu'à ce que le douzième processeur au moins subisse une forte charge. Cependant, lorsque la charge sur le système augmente, cette amélioration est soumise aux performances des autres ressources. Les serveurs sont souvent sous-configurés en matière de disques et de mémoire. Vous devez ajouter d'autres processeurs uniquement dans les circonstances suivantes :

- Si la charge moyenne utilise plus de 75 % de la capacité des processeurs. Si l'utilisation de l'UC est inférieure à 75 %, le fait d'ajouter des processeurs n'améliore pas nécessairement les performances.
- Si vous constatez une augmentation satisfaisante des performances.

Si OpenText eDirectory est configuré avec un nombre de threads trop élevé, une quantité considérable du temps processeur est consacrée à la commutation de contexte. Dans ce cas, la réduction du nombre de threads peut améliorer le débit.

Sous-système de mémoire

Les performances des applications serveur peuvent être considérablement améliorées lorsque la mémoire RAM augmente. La mise en cache de la base de données OpenText eDirectory dans le système de fichiers ou dans le cache FLAIM peut améliorer les performances lors des opérations de recherche et de modification. Toutefois, vous ne pouvez pas mettre en cache l'intégralité du fichier DIB dans les déploiements à grande échelle. Évitez l'échange de pages, même si cela implique la réduction de la taille des caches d'entrées et de blocs FLAIM. Utilisez l'outil vmstat pour trouver plus d'informations sur le sous-système de mémoire.

Lorsqu'OpenText eDirectory utilise de la mémoire, chaque thread de la réserve utilise 1 Mo de mémoire RAM pour sa pile. Par défaut, la taille du cache FLAIM est de 200 Mo.

Plusieurs modules chargeables sont démarrés en même temps qu'OpenText eDirectory, mais l'architecture de modules chargeables d'OpenText eDirectory permet de réduire l'encombrement de mémoire du processus, car les modules non utilisés (SecretStore, LDAP ou eMBox, par exemple) ne sont pas chargés. En outre, les produits tels qu'IDM comportent certains modules qui s'exécutent à l'intérieur d'OpenText eDirectory.

La mémoire utilisée par OpenText eDirectory peut augmenter. Même si de la mémoire est libérée par un processus OpenText eDirectory, elle peut ne pas être transmise à la réserve libre du système, car le gestionnaire de mémoire utilisé en interne par OpenText eDirectory essaie d'optimiser les allocations de mémoire pour l'avenir. C'est l'une des raisons pour lesquelles il n'est pas recommandé de configurer FLAIM de manière dynamique. Utilisez l'outil Top pour trouver la taille approximative de mémoire virtuelle du processus ndsd dans votre déploiement.

La quantité de mémoire maximale qui peut être allouée à un processus est limitée de différentes manières. Une certaine quantité de mémoire RAM est utilisée par le système d'exploitation et d'autres processus du système. Le système d'exploitation peut imposer des limites à la mémoire RAM physique utilisée par un processus.

Sous-système réseau

Les déploiements courants ont une bande passante suffisante pour gérer les pics de charge réseau. Une bande passante adéquate permet de réduire les erreurs, les collisions et les pertes de paquets. Utilisez l'outil netstat pour déterminer les statistiques réseau.

Plusieurs systèmes d'exploitation fournissent des paramètres TCP/IP adaptables pour l'optimisation des serveurs utilisant le réseau de manière intensive. Pour plus d'informations, reportez-vous à la documentation des systèmes d'exploitation.

Si le réseau est à l'origine du goulot d'étranglement, vous devez augmenter la bande passante. La configuration d'un réseau privé dédié entre les serveurs d'applications et le serveur OpenText eDirectory peut également permettre de réduire l'encombrement du réseau.

4 Optimisation des sous-systèmes OpenText eDirectory

Cette section présente les informations suivantes :

- « Base de données FLAIM » page 19
- « Réserve de threads » page 21
- « ACL » page 21
- « Réplication » page 25
- « Disque SSD » page 26
- « Intervalle de mise à jour à la connexion NMAS » page 26
- « Surcharge d'information SSL » page 26
- « Utilitaire ICE (Import Convert and Export) » page 26
- « ldif2dib » page 26
- « Amélioration de la prise en charge de la taille de paquet par NCP » page 27

Base de données FLAIM

Le dimensionnement du cache est sans doute le facteur ayant la plus forte incidence sur les performances globales d'OpenText eDirectory. Plus le nombre d'éléments (blocs et entrées) pouvant être mis en cache est important, meilleures sont les performances globales. Le pourcentage d'occurrences d'entrées ou de blocs trouvées dans le cache est appelé rapport d'occurrence. Un rapport élevé se traduit par de meilleures performances. OpenText iMonitor permet d'afficher le rapport d'occurrence.

Le cache de blocs s'avère particulièrement utile dans les opérations de mise à jour. Le cache d'entrées est particulièrement utile pour les recherches d'entrées basées sur une étendue définie. Toutefois, les recherches portant sur un niveau et sur une sous-arborescence utilisent le cache d'entrées et le cache de blocs. Le cache de blocs permet de récupérer les index. Créez le type d'index approprié ; pour plus d'informations, consultez la section « Choix des index » page 20.

Une panne du cache de blocs peut entraîner une opération de lecture de disque. Les lectures de disque sont toujours coûteuses, mais elles peuvent être évitées si un bloc est récupéré à partir du cache du système de fichiers.

La quantité de mémoire nécessaire pour mettre dans le cache de blocs l'intégralité de la base de données équivaut presque à la taille de la base de données sur le disque. Quant à la quantité de mémoire requise dans le cache d'entrées pour cette même base de données, elle équivaut à deux à quatre fois la taille de la base de données sur le disque. Lorsque la mémoire système est inférieure, essayez d'utiliser un cache d'entrées plus petit et un cache de blocs ou de système de fichiers beaucoup plus grand.

Si les lectures sont localisées sur un ensemble d'entrées de l'annuaire, vous devez augmenter le cache d'entrées jusqu'à améliorer le rapport d'occurrence.

Si le modèle de lecture est totalement aléatoire et que le fichier DIB est beaucoup plus volumineux que la mémoire RAM disponible, le cache de blocs ou de système de fichiers doit être plus grand que le cache d'entrées.

Toute méthode utilisée pour ajuster OpenText eDirectory dans le but d'améliorer les performances doit faire l'objet de tests empiriques. Pour les environnements à forte activité de recherche, un rapport d'occurrence de 2:1 dans le cache de blocs est un rapport correct. Vérifiez que la mémoire disponible est suffisante pour d'autres processus. Évitez l'échange de pages, même si cela implique la réduction de la taille des caches FLAIM.

Étant donné que le gestionnaire FLAIM préalloue la mise en cache, la mémoire allouée au cache OpenText eDirectory n'est jamais fragmentée par le gestionnaire de mémoire natif du système d'exploitation.

Choix des index

Les index sont conçus pour améliorer les performances des recherches sur un niveau ou dans une sous-arborescence. Les groupes dynamiques utilisent également des recherches sur un niveau ou dans une sous-arborescence. Les index ne sont pas utilisés pour les recherches basées sur une étendue définie.

Étant donné qu'un index Presence ne fait pas de différence entre les valeurs présentes et non présentes (supprimées), il est utilisé principalement en interne. Si les applications effectuent une recherche de type Presence, cet index n'est jamais utilisé, si bien qu'aucun index Presence ne doit être créé pour les applications.

Les applications peuvent créer un index Value pour un attribut, ce qui est suffisant pour la plupart des recherches. Le gestionnaire FLAIM peut utiliser un index Value pour effectuer des recherches de type Presence et Substring dans les attributs.

Un index Substring peut ralentir considérablement les mises à jour effectuées sur un attribut. Le nombre de blocs d'index requis pour prendre en charge un index Substring est assez important par rapport à l'index Value. Cela signifie que leur mise en cache nécessite un cache de blocs plus important. Créez un index Substring uniquement lorsque cela est nécessaire. Un index Value devrait suffire pour la plupart des recherches. Toutefois, si des recherches Substring ne donnent pas de performances acceptables avec un index Value, vous pouvez créer un index Substring sur ces attributs.

Si l'exécution d'une recherche est longue malgré l'index choisi, vous pouvez intégrer un index Value plus récent sur l'un des attributs de filtre de la recherche. Choisissez l'attribut qui produit les meilleurs résultats lors de l'indexation.

Optimisation des mises à jour

Le cache de blocs s'avère particulièrement utile dans les opérations de mise à jour. Les index se trouvent également dans le cache de blocs. Les index accélèrent les recherches, mais s'ils sont trop nombreux, le serveur consacre ses ressources à leur gestion. Les index sont modifiés si les valeurs d'attribut sont modifiées, ajoutées ou supprimées. Pendant les opérations importantes de téléchargement, les index peuvent être désactivés pour que le téléchargement soit plus rapide.

Le fait d'avoir l'annuaire RFL sur un disque et l'annuaire DIB sur un autre améliore les performances.

La limite acceptable en matière de temps de réponse pour une opération de mise à jour peut être contrôlée à l'aide de la commande `maxdirtycache`. Par exemple, si la limite acceptable de temps de réponse du serveur est de 5 secondes et que la vitesse d'écriture aléatoire sur le disque est de 20 Mo par seconde, la commande `maxdirtycache` doit être définie comme suit : 20 x 5 = 100 Mo. Assurez-vous que le cache de blocs peut contenir ces blocs altérés dans la mémoire. Pour plus d'informations, reportez-vous au « Modification des paramètres de cache FLAIM dans _ndsdb.ini » page 31.

Réserve de threads

Par défaut, le nombre maximal de threads qui peuvent être disponibles en réserve est de 256. Ce nombre devrait suffire pour la plupart des déploiements. Il peut être porté à 512 threads dans les déploiements de grande envergure. Vous devez augmenter le nombre de threads en réserve dans les cas suivants :

- si le nombre de threads inactifs est souvent nul ;
- si la durée moyenne passée par une tâche dans la file d'attente des éléments prêts est élevée et en hausse ;
- si le nombre de tâches dans la file d'attente des éléments prêts est élevé et en hausse.

Augmentez encore le nombre maximum de threads si les performances du serveur augmentent. L'utilisation de l'UC devrait également augmenter.

Pour plus d'informations sur l'affichage des statistiques de la réserve de threads, consultez la section « Affichage des statistiques des réserves de threads » du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*.

ACL

- « Amélioration des recherches et des lectures d'OpenText eDirectory » page 22
- « Désactivation des modèles ACL » page 23

Amélioration des recherches et des lectures d'OpenText eDirectory

Les résultats d'une recherche LDAP dans OpenText eDirectory dépendent du nombre d'attributs renvoyés pour un utilisateur (`inetOrgPerson`).

Lorsqu'un objet est créé dans OpenText eDirectory, les listes ACL par défaut peuvent être ajoutées à l'objet. Cela dépend des modèles d'ACL dans la définition de schéma de la classe d'objet à laquelle appartient cet objet. Par exemple, dans la configuration par défaut d'`inetOrgPerson`, jusqu'à six ACL peuvent être ajoutées sur l'objet Utilisateur. Lorsqu'une requête de recherche LDAP renvoie cet objet Utilisateur avec tous les attributs, elle met légèrement plus de temps à retourner cet objet au client qu'à retourner cet objet Utilisateur sans attributs ACL.

Les ACL par défaut peuvent être désactivées, mais les administrateurs peuvent ne pas souhaiter les désactiver, car elles sont nécessaires à l'amélioration du contrôle d'accès. Vous pouvez néanmoins améliorer les performances de recherche en n'en faisant pas la requête ou en les marquant comme étant des attributs filtrés et lus. Ces modifications n'altèrent pas les applications, car la plupart d'entre elles utilisent des privilèges efficaces et ne comptent pas sur des listes ACL spécifiques.

Non-requête d'ACL : Un attribut ACL n'est pas requis par plusieurs applications, si bien que les applications peuvent être modifiées pour requérir des attributs spécifiques qui concernent l'application. Cela se traduit par de meilleures performances de la recherche LDAP.

Marquage d'un attribut ACL comme étant lu et filtré : Si une application ne peut pas être modifiée, l'administrateur peut utiliser la commande `arf_acl.ldif` pour marquer l'attribut ACL comme étant lu et filtré. Lorsque la liste ACL est marquée comme étant un attribut lu et filtré, le serveur ne renvoie pas l'attribut sur l'entrée si tous les attributs font l'objet d'une requête. Toutefois, si la recherche LDAP renvoie des attributs opérationnels ou si la requête porte spécifiquement sur des attributs ACL, l'attribut marqué est retourné. Le fichier `rrf_acl.ldif` peut servir à désactiver le drapeau de lecture et filtrage sur un attribut ACL. Ces LDIF ont une incidence sur l'attribut ACL sur le schéma ; par conséquent, seul un utilisateur avec des droits de superviseur sur la racine de l'arborescence peut les étendre.

Par défaut, les listes ACL ne sont pas marquées comme étant lues et filtrées, si bien qu'aucune amélioration des performances n'est constatée pour les requêtes qui renvoient tous les attributs.

Le tableau suivant représente l'emplacement des fichiers `arf_acl.ldif` et `rrf_acl.ldif` dans différentes plates-formes.

Plate-forme	Emplacement
Linux	◆ `/opt/novell/eDirectory/lib64/nds-schema/`
Windows	◆ `<emplacement_dézippé>\eDirectory\windows\x64\NDSonNT\ndsnt\nds`

Désactivation des modèles ACL

Vous pouvez désactiver les modèles ACL (Access Control List - liste de contrôle d'accès) pour accroître les performances de chargement par lots. Certaines ACL risquent alors d'être manquantes, mais vous pouvez résoudre ce problème en ajoutant les ACL nécessaires au fichier LDIF ou en les appliquant ultérieurement.

1. Exécutez la commande suivante :

   ```
   ldapsearch -D cn_of_admin -w password -b cn=schema -s base
   objectclasses=inetorgperson
   ```

   Cette commande donne le résultat suivant :

   ```
   dn: cn=schema
   ```

   ```
   objectClasses: (2.16.840.1.113730.3.2.2 NAME 'inetOrgPerson' SUP
   organizationalPerson STRUCTURAL MAY (groupMembership $ ndsHomeDirectory
   $ loginAllowedTimeMap $ loginDisabled $ loginExpirationTime $
   loginGraceLimit $ loginGraceRemaining $ loginIntruderAddress $
   loginIntruderAttempts $ loginIntruderResetTime $
   loginMaximumSimultaneous $ loginScript $ loginTime $
   networkAddressRestriction $ networkAddress $ passwordsUsed $
   passwordAllowChange $ passwordExpirationInterval $
   passwordExpirationTime $ passwordMinimumLength $ passwordRequired $
   passwordUniqueRequired $ printJobConfiguration $ privateKey $ Profile $
   publicKey $ securityEquals $ accountBalance $ allowUnlimitedCredit $
   minimumAccountBalance $ messageServer $ Language $ UID $
   lockedByIntruder $ serverHolds $ lastLoginTime $ typeCreatorMap $
   higherPrivileges $ printerControl $ securityFlags $ profileMembership $
   Timezone $ sASServiceDN $ sASSecretStore $ sASSecretStoreKey $
   sASSecretStoreData $ sASPKIStoreKeys $ userCertificate
   $ nDSPKIUserCertificateInfo $ nDSPKIKeystore $ rADIUSActiveConnections
   $
   rADIUSAttributeLists $ rADIUSConcurrentLimit $ rADIUSConnectionHistory
   $ rADIUSDefaultProfile $ rADIUSDialAccessGroup $ rADIUSEnableDialAccess
   $ rADIUSPassword $ rADIUSServiceList $ audio $ businessCategory $
   carLicense $ departmentNumber $ employeeNumber $ employeeType $
   givenName $ homePhone $ homePostalAddress  $ initials $ jpegPhoto $
   labeledUri $ mail $ manager $ mobile $ pager $ ldapPhoto $
   preferredLanguage $ roomNumber $ secretary $ uid $ userSMIMECertificate
   $ x500UniqueIdentifier $ displayName $ userPKCS12) X-NDS_NAME 'User' X
   -NDS_NOT_CONTAINER '1' X-NDS_NONREMOVABLE '1' X-NDS_ACL_TEMPLATES
   ('2#subtree#[Self]#[All Attributes Rights]' '6#entry#[Self]#loginScript'
   '1#subtree#[Root Template]#[Entry Rights]'
   '2#entry#[Public]#messageServer' '2#entry#[Root Template]#groupMembership'
   '6#entry#[Self]#printJobConfiguration' '2#entry#[Root
   Template]#networkAddress'))
   ```

2. Dans le résultat obtenu à l'étape précédente, supprimez les informations figurant en gras.
3. Enregistrez le résultat révisé sous la forme d'un fichier LDIF.
4. Ajoutez les informations suivantes dans le nouveau fichier LDIF :

   ```
   dn: cn=schema
   ```

   ```
   changetype: modify
   ```

```
delete: objectclasses

objectclasses: (2.16.840.1.113730.3.2.2)

-

add:objectclasses
```

Votre fichier LDIF devrait à présent ressembler à ceci :

```
dn: cn=schema

changetype: modify

delete: objectclasses

objectclasses: (2.16.840.1.113730.3.2.2)

-

add:objectclasses

objectClasses: (2.16.840.1.113730.3.2.2 NAME 'inetOrgPerson' SUP
organizationalPerson STRUCTURAL MAY (groupMembership $ ndsHomeDirectory
$ loginAllowedTimeMap $ loginDisabled $ loginExpirationTime $
loginGraceLimit $ loginGraceRemaining $ loginIntruderAddress $
loginIntruderAttempts $ loginIntruderResetTime $
loginMaximumSimultaneous $ loginScript $ loginTime $
networkAddressRestriction $ networkAddress $ passwordsUsed $
passwordAllowChange $ passwordExpirationInterval $
passwordExpirationTime $ passwordMinimumLength $ passwordRequired
$ passwordUniqueRequired $ printJobConfiguration $ privateKey $ Profile
$
publicKey $ securityEquals $ accountBalance $ allowUnlimitedCredit $
minimumAccountBalance $ messageServer $ Language $ UID $
lockedByIntruder $ serverHolds $ lastLoginTime $ typeCreatorMap $
higherPrivileges $ printerControl $ securityFlags $ profileMembership $
Timezone $ sASServiceDN $ sASSecretStore $ sASSecretStoreKey $
sASSecretStoreData $ sASPKIStoreKeys $ userCertificate $
nDSPKIUserCertificateInfo $ nDSPKIKeystore $ rADIUSActiveConnections $
rADIUSAttributeLists $ rADIUSConcurrentLimit $ rADIUSConnectionHistory
$
rADIUSDefaultProfile $ rADIUSDialAccessGroup $ rADIUSEnableDialAccess
$ rADIUSPassword $ rADIUSServiceList $ audio $ businessCategory $
carLicense $ departmentNumber $ employeeNumber $ employeeType $
givenName $ homePhone $ homePostalAddress  $ initials $ jpegPhoto $
labeledUri $ mail
$ manager $ mobile $ pager $ ldapPhoto $ preferredLanguage $ roomNumber
$ secretary $ uid $ userSMIMECertificate $ x500UniqueIdentifier $
displayName $ userPKCS12) X-NDS_NAME 'User' X-ND S_NOT_CONTAINER '1' X
-NDS_NONREMOVABLE '1')
```

5 Saisissez la commande suivante :

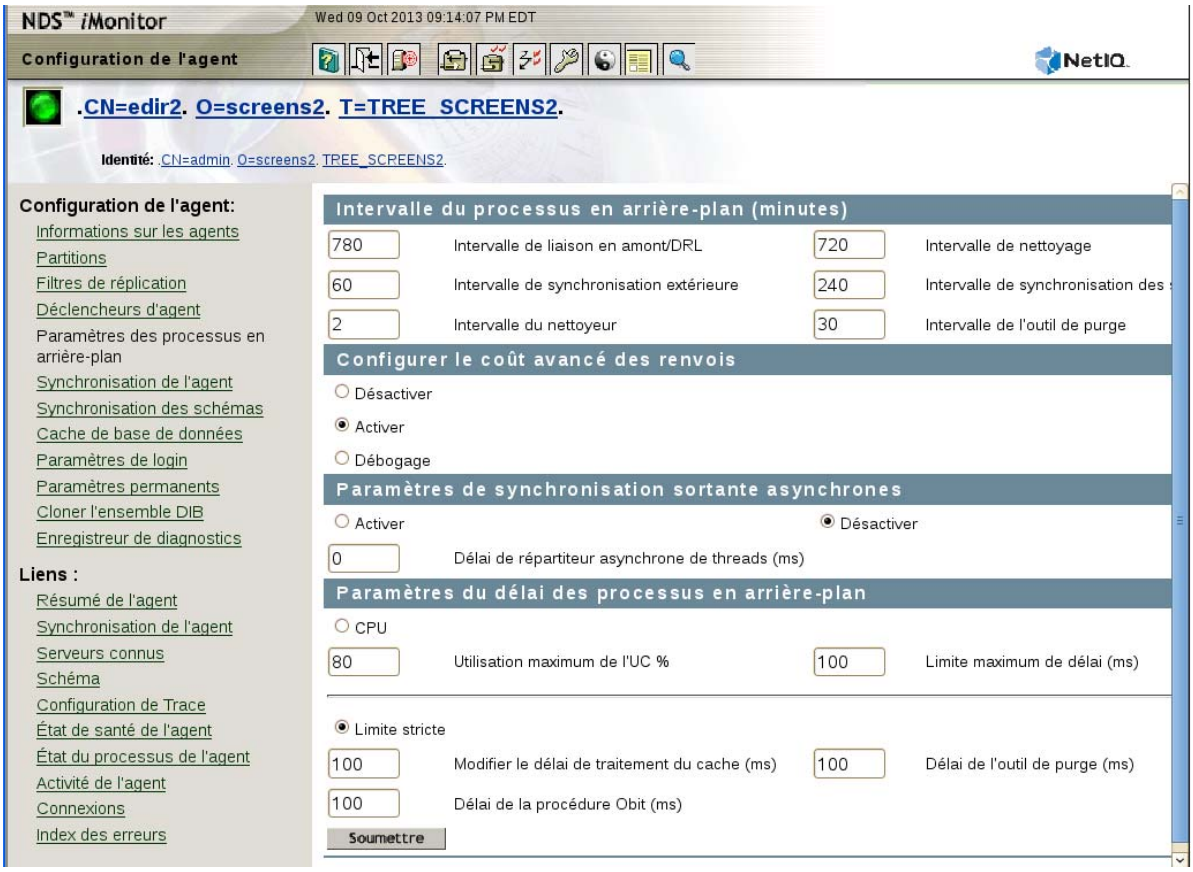
```
ldapmodify -D cn_of_admin -w password -f LDIF_file_name
```

Réplication

Dans cette version, certains processus en arrière-plan ont été redéfinis afin de répondre aux besoins des environnements dynamiques de grande envergure. Pour plus d'informations, reportez-vous à la section « Gestion des processus en arrière-plan » du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*.

Il est recommandé de définir la limite stricte sur *5 ms* et d'*activer* la synchronisation sortante asynchrone. Toutefois, si l'utilisation du processeur est élevée, augmentez la durée de mise en veille. La Figure 4-1 représente les valeurs définies pour l'option **Paramètres du délai des processus en arrière-plan**.

Figure 4-1 *Paramètres des processus en arrière-plan*



Des tests de laboratoire ont été effectués en interne sur une installation de 10 serveurs avec les paramètres suivants : Limite stricte - *0 ms*, Synchronisation sortante asynchrone - *activée* et Délai de répartiteur asynchrone de threads - *0 ms*. Les tests ont montré que la réplication est 7 fois plus rapide qu'avec les paramètres par défaut. Lors de ce test, aucune autre opération du client n'a été effectuée.

REMARQUE : afin d'optimiser les performances de vos systèmes grâce à ces améliorations de l'évolutivité, vous devez utiliser OpenText eDirectory 9.1 ou une version ultérieure sur tous les serveurs. Même si l'anneau de répliques comprend des versions plus anciennes, les performances sont améliorées.

Disque SSD

Cette version prend en charge les disques SSD Enterprise qui améliorent les opérations d'E/S. Le Tableau 4-1 page 26 indique l'amélioration des performances de réparation sur SSD dans notre configuration de test :

Tableau 4-1 *Performances de réparation*

Taille du fichier DIB (Go)	Disque dur (durée en minutes)	SSD (durée en minutes)	% d'amélioration
11	80	53	33,75
24	277	169	38,98
34	542	296	45,38
75	1 383	618	55,31
98	3 171	1 023	67,73

Intervalle de mise à jour à la connexion NMAS

Pour plus d'informations, reportez-vous à la section « Utilisation des attributs sasUpdateLoginInfo et sasUpdateLoginTimeInterval » du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*.

Surcharge d'information SSL

Le protocole LDAP sur SSL ajoute une charge supplémentaire sur le processeur en raison de ses exigences de codage. Une étude des performances menée en laboratoire indique des rapports d'occurrence 10 % supérieurs en raison de la surcharge d'information liée au codage.

Utilitaire ICE (Import Convert and Export)

L'utilitaire ICE (Import Convert and Export) d'OpenText utilise un protocole de mise à jour par lots optimisé appelé LBURP pour télécharger les données dans OpenText eDirectory. Ce protocole est beaucoup plus rapide que le téléchargement de données grâce à la simple commande `ldapmodify`. Pour plus d'informations, reportez-vous à la section Utilitaire du chargement par lots hors ligne du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*.

ldif2dib

Pour optimiser les performances d'OpenText pendant le téléchargement par lots hors ligne à l'aide de l'utilitaire ldif2dib, reportez-vous à la section Ajustement de ldif2dib du *Guide d'administration d'OpenText™ eDirectory CE 24.4 (v9.3)*.

Amélioration de la prise en charge de la taille de paquet par NCP

Pour communiquer entre les différents serveurs, OpenText eDirectory utilise le protocole NCP (Netware Core Protocol) comme protocole de communication. Dans les versions précédentes, la taille de paquet maximale autorisée par NCP était de 64 Ko, ce qui limitait le débit maximal lors du transfert des données via ce protocole. Cette version permet de traiter via NCP une taille de paquet allant jusqu'à 1 Mo, ce qui permet à OpenText eDirectory de synchroniser jusqu'à 1 Mo en un seul paquet. OpenText eDirectory démarre la synchronisation avec une taille de paquet de 64 Ko et augmente la taille de paquet en fonction des données restantes à synchroniser. Cela améliore considérablement les performances de réplication. Si la version de vos deux serveurs est 9.2, aucune configuration supplémentaire n'est nécessaire pour bénéficier de cette amélioration.

5 Configuration d'OpenText eDirectory

Cette section présente les informations suivantes :

- « Configuration du sous-système FLAIM » page 29
- « Modification des paramètres de cache FLAIM » page 30

Configuration du sous-système FLAIM

Afin de répondre à une grande variété de déploiements et configurations, OpenText eDirectory fournit deux mécanismes pour contrôler la consommation de la mémoire cache. Ces mécanismes sont mutuellement exclusifs.

- « Limite de cache stricte » page 29
- « Ajustement dynamique de la limite » page 29

Limite de cache stricte

Vous pouvez indiquer une limite de mémoire stricte de l'une des manières suivantes :

- sous la forme d'un nombre d'octets fixe ;
- sous la forme d'un pourcentage de mémoire physique ;
- sous la forme d'un pourcentage de mémoire physique disponible.

Lorsqu'une limite stricte est spécifiée à l'aide de la deuxième ou de la troisième méthode, elle est toujours convertie en un nombre d'octets fixe. Cela signifie que, pour la deuxième méthode, le nombre d'octets correspond au pourcentage de mémoire physique détecté au démarrage d'OpenText eDirectory. Pour la troisième méthode, le nombre d'octets correspond au pourcentage de mémoire physique disponible détecté au démarrage d'OpenText eDirectory.

Ajustement dynamique de la limite

Dans le cadre de l'ajustement dynamique, OpenText eDirectory ajuste régulièrement sa consommation de mémoire en fonction de la quantité de mémoire consommée par les autres processus. L'ajustement dynamique de mémoire fonctionne bien dans les scénarios classiques, mais ce mécanisme n'est pas recommandé pour optimiser les performances d'OpenText eDirectory sur les plates-formes Linux en raison des différences importantes entre les modèles d'utilisation de la mémoire et les allocateurs de mémoire sur les plates-formes Linux.

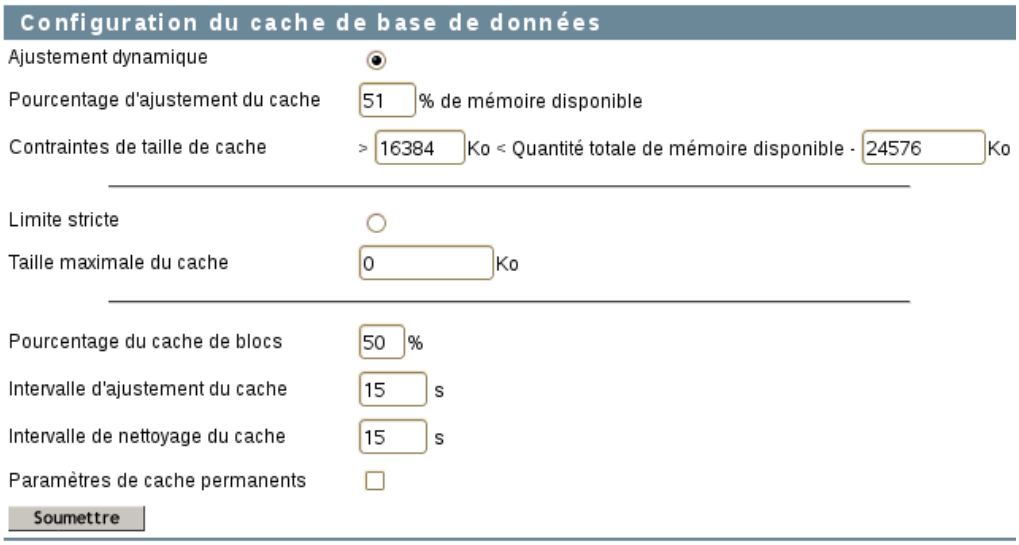
Modification des paramètres de cache FLAIM

- « Modification des paramètres de cache FLAIM par l'intermédiaire d'OpenText iMonitor » page 30
- « Modification des paramètres de cache FLAIM dans _ndsdb.ini » page 31

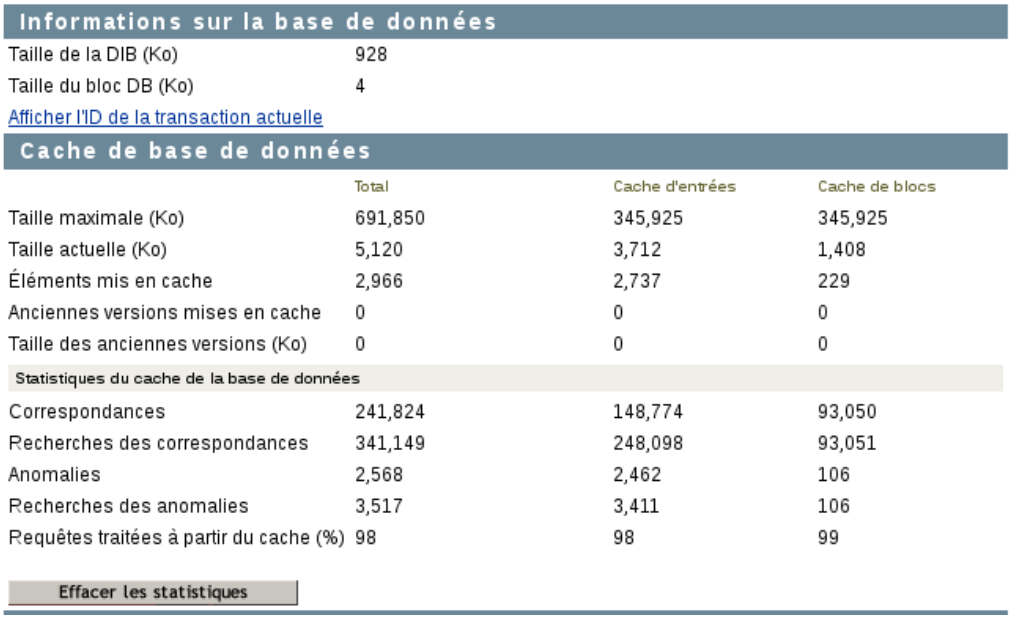
Modification des paramètres de cache FLAIM par l'intermédiaire d'OpenText iMonitor

Vous pouvez utiliser OpenText iMonitor pour effectuer les opérations suivantes :

- afficher ou modifier les paramètres du cache ;



- surveiller les statistiques relatives au cache.



Reportez-vous au cache de base de données dans Configuration de l'Agent d'OpenText iMonitor pour obtenir les informations ci-dessus.

Informations sur le cache de base de données	Description
Taille maximale	Taille maximale en ko que peut avoir le cache spécifié.
Taille actuelle	Taille actuelle en ko du cache spécifié.
Éléments mis en cache	Nombre d'éléments contenus dans le cache spécifié.
Anciennes versions mises en cache	Nombre d'anciennes versions contenues dans le cache spécifié. Les anciennes versions des éléments contenus dans le cache sont conservées par souci de cohérence des transactions de lecture dans la base de données. Autrement dit, si un thread se trouve dans une transaction de lecture et un autre dans une transaction d'écriture, les anciennes versions des blocs modifiés par l'opération d'écriture sont conservées à l'intention de l'utilisateur qui effectue la lecture. Cet utilisateur a ainsi la garantie d'obtenir des résultats cohérents durant toute sa transaction de lecture, même si des modifications sont effectuées dans l'intervalle.
Taille des anciennes versions	Taille en ko des anciennes versions des éléments contenus dans le cache.
Correspondances	Nombre d'accès réussis à un élément à partir du cache spécifié.
Recherches des correspondances	Nombre d'éléments examinés avant un accès réussi à un élément à partir du cache spécifié. Le rapport de recherche de correspondances permet de mesurer l'efficacité de la recherche dans le cache. Normalement, il doit avoisiner 1:1.
Anomalies	Nombre de fois où un élément introuvable dans le cache a dû être obtenu dans un cache de niveau inférieur ou à partir du disque.
Recherches des anomalies	Nombre d'éléments examinés avant qu'il soit établi que l'élément voulu ne se trouvait pas dans le cache spécifié. Le rapport de recherche d'anomalies permet de mesurer l'efficacité de la recherche dans le cache. Normalement, il doit avoisiner 1:1.

Modification des paramètres de cache FLAIM dans _ndsdb.ini

Les paramètres de cache FLAIM et d'autres configurations de FLAIM peuvent être définis en modifiant le fichier `_ndsdb.ini` qui se trouve dans le répertoire du fichier DIB. Redémarrez OpenText eDirectory une fois le fichier `_ndsdb.ini` modifié.

Vous pouvez définir la limite d'ajustement dynamique ou la limite de cache stricte. Les options de cache sont répertoriées ci-dessous. Vous pouvez indiquer plusieurs options, dans l'ordre de votre choix, en les séparant par une virgule. Elles sont toutes facultatives.

- **DYN ou FIXE** - Ajustement dynamique d'une limite ou d'une limite stricte.
- **% : percentage** - Pourcentage de mémoire disponible ou physique à utiliser.

- **DISPO ou TOTAL** - Le pourcentage indique la mémoire disponible ou la mémoire physique totale. Il s'applique uniquement à la limite stricte et est ignoré pour la limite d'ajustement dynamique. Les limites d'ajustement dynamique sont en effet toujours calculées en fonction de la mémoire physique disponible. Par défaut, il s'agit de DISPO.
- **MIN:octets** - Nombre minimum d'octets.
- **MAX:octets** - Nombre maximum d'octets.
- **LEAVE:octets** - Nombre minimum d'octets à laisser.

  Par exemple :

  ```
  cache=HARD,%:75, MIN:200000000
  ```

  ```
  cache=500000000
  ```

- **preallocatecache: true/false** - Avec ce paramètre, OpenText eDirectory préalloue la quantité de mémoire spécifiée par la limite de cache stricte.
- **rfldirectory -** Un autre chemin d'accès peut être spécifié pour les fichiers RFL.
- **cpinterval** - Nombre de secondes après lequel FLAIM force un point de contrôle. La valeur par défaut est 3 minutes.
- **maxdirtycache** - Nombre maximal d'octets de cache altéré.
- **lowdirtycache** - Nombre minimal d'octets de cache altéré.
- **blockcachepercent** - Pourcentage du cache FLAIM utilisé pour le cache de blocs.
- **cacheadjustinterval** - Intervalle en secondes d'ajustement dynamique du cache.
- **cachecleanupinterval** - Intervalle en secondes de nettoyage des anciennes versions des entrées et des blocs dans le cache.