draft-bannister-dbis-passwd-05

dbis : passwd et group

Ce document étend DBIS pour supporter les bases passwd et group. Ces schémas de base devraient être compatible avec NIS, mais stockés dans des entrées X.500 pour qu'ils puissent être résolus avec le protocole LDAP. Une base passwd représente les comptes de connexion utilisateurs sur les systèmes UNIX et dérivés et une base group représente des groupes d'utilisateurs.

Ce document décris les maps de configuration pour les bases de données passwd et group, et les entrées de base référencée par ces maps.

Maps de configuration

Toutes les base décrites dans ce document utilisent les maps de configuration standard définies dans draft-bannister-dbis-mapping. Additionnellement, les entrées dbisMapConfig pour les bases passwd et group devraient être assignées avec l'objectClass dbisPasswdConfig et dbisGroupConfig, respectivement.

Il est recommandé que l'entrée dbisMapConfig pour une base passwd ou group ait l'attribut dbisMapFilter définis en accord avec la table suivante :

```
Database______dbisMapFilter
passwd________objectClass=posixUserAccount
group_________objectClass=posixGroupAccount
```

Exemple d'entrées de map de configuration

L'exemple suivant donne un exemple d'entrée de map de configuration pour une base passwd :

```
dn: cn=passwd,en=sales.corp,ou=domain-mappings,o=infra
objectClass: top
objectClass: dbisMapConfig
objectClass: dbisPasswdConfig
cn: passwd
dbisMapDN: cn=passwd,ou=dbis,o=infra
dbisMapFilter: objectClass=posixUserAccount
dbisMapGecos: displayName
profileTTL: 900
description: Primary passwd database
```

L'exemple suivant donne un exemple d'entrée de map de configuration pour une base group :

```
dn: cn=group,en=sales.corp,ou=domain-mappings,o=infra
objectClass: top
objectClass: dbisMapConfig
objectClass: dbisGroupConfig
cn: group
dbisMapDN: cn=group,ou=dbis,o=infra
dbisMapFilter: objectClass=posixGroupAccount
profileTTL: 900
description: Primary group database
```

base passwd

Une base de données passwd contient les champs suivants :

- Nom d'utilisateur
- Mot de passe de l'utilisateur
- Identifiant numérique de l'utilisateur
- Identifiant numérique du groupe primaire de l'utilisateur
- description de l'utilisateur
- Chemin du répertoire personnel de l'utilisateur
- Chemin du shell de connexion de l'utilisateur.

ObjectClass

Une entrée dbisMapConfig pour une base passwd devrait avoir l'objectClass dbisPasswdConfig. Une entrée passwd devrait être définie par une entrée LDAP avec l'objectClass posixUserAccount. Vu que c'est une classe auxiliaire, elle doit également avoir une classe structurelle assignée qui n'est pas définis dans ce document, par exemple inetOrgPerson.

dbisPasswdConfig

La classe dbispasswdConfig est définie comme suit :

```
objectclass ( 1.3.6.1.4.1.23780.219.1.8 NAME 'dbisPasswdConfig' DESC 'DBIS passwd configuration map' SUP
dbisMapConfig STRUCTURAL MUST dbisMapGecos MAY dbisOverlayDN )
```

posixUserAccount

La classe posixUserAccount est définie comme suit :

```
objectclass ( 1.3.6.1.4.1.23780.219.1.9 NAME 'posixUserAccount' DESC 'User account with POSIX attributes' SUP
top AUXILIARY MUST ( en $ uidNumber $ gidNumber $ homeDirectory ) MAY ( authPassword $ userPassword $
loginShell $ disableObject ) )
```

Attributs

dbisMapGecos

le champ gecos maintient traditionnellement le nom complet de l'utilisateur et parfois d'autres informations descriptive sur le compte. informations qui sont mieux stockées dans des attributs nommés plus spécifiquement. Vu qu'il y a une variété de manière de stocker ces informations déjà disponibles, ce document ne définis pas de champs additionnel pour les informations gecos, mais l'attribut dbisMapGecos doit être assigné à une entrée dbisPasswdConfig et qui maintient le nom de l'attribut à utiliser pour fournir les informations gecos. Il est définis :

```
attributetype ( 1.3.6.1.4.1.23780.219.2.13 NAME 'dbisMapGecos' DESC 'Source attribute for gecos field'
EQUALITY caseIgnoreIA5Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

L'objectClass posixUserAccount est auxiliaire et doit toujours être associé avec une autre classe structurelle. Une telle classe est inetOrgPerson. Si les comptes utilisateurs on la class inetOrgPerson, le displayName peut être une valeur approprié pour l'attribut dbisMapGecos.

dbisOverlayDN

Un ou plusieurs DN identifiant la base de recherche pour les entrées overlay sont stockés dans l'attribut dbisOverlayDN qui peut être assigné à une entrée dbisPasswdConfig :

```
attributetype ( 1.3.6.1.4.1.23780.219.2.14 NAME 'dbisOverlayDN' DESC 'DN of search base for DBIS overlay
entries' EQUALITY distinguishedNameMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.12 )
```

en

Le nom du compte utilisateur est stocké dans l'attribut en qui est définis dans draft-bannister-dbis-mapping. L'attribut en doit être associé avec une entrée posixUserAccount et devrait former le RDN.

uidNumber

L'UID est stocké dans l'attribut uidNumber qui doit être assigné à l'entrée posixUserAccount :

```
attributetype ( 1.3.6.1.1.1.1.0 NAME 'uidNumber' DESC 'An integer uniquely identifying a user in an
administrative domain' EQUALITY integerMatch ORDERING integerOrderingMatch SYNTAX
1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
```

gidNumber

Le GID primaire est stocké dans l'attribut gidNumber qui doit être assigné à une entrée posixGroupAccount :

```
attributetype ( 1.3.6.1.1.1.1.1 NAME 'gidNumber' DESC 'An integer uniquely identifying a group in an
administrative domain' EQUALITY integerMatch ORDERING integerOrderingMatch SYNTAX
1.3.6.1.4.1.1466.115.121.1.27 SINGLE-VALUE )
```

homeDirectory

Le chemin du répertoire personnel de l'utilisateur est stocké dans l'attribut homeDirectory qui doit être assigné à une entrée posixUserAccount :

```
attributetype ( 1.3.6.1.1.1.1.3 NAME 'homeDirectory' DESC 'The absolute path to the home directory' EQUALITY
caseExactIA5Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

authPassword

Le mot de passe chiffré de l'utilisateur est stocké dans l'attribut authPassword qui est définis dans la rfc3112, et qui peut être assigné à une entrée posixUserAccount.

Bien qu'un DUA peut implémenter tout schéma de mot de passe d'authentification supporté par le DSA, il doit supporter le schéma CRYPT pour la compatibilité, qui est une implémentation de l'algorithme de chiffrement traditionnel UNIX. Cependant, il est recommandé qu'un schéma plus sécurisé soit utilisé.

Si l'attribut authPassword a plusieurs valeurs, le DUA devrait sélectionner un mot de passe basé sur le schéma d'authentification le plus fort qu'il supporte et l'utiliser pour l'authentification. Si l'authentification échoue, le DUA ne devrait pas tenter d'utiliser d'autres valeurs. Si l'attribut n'utilise pas de schéma supporté par le DUA, alors le DUA ne devrait pas réussir l'authentification.

Si une entrée posixUserAccount n'a pas d'attribut authPassword ou userPassword, le compte est blocké. Un DUA ne devrait pas réussir une authentification sur un compte blocké.

Le transfert des valeurs authPassword est fortement découragé quand le service de transport sous-jacent ne peut pas garantir la confidentialité.

userPassword

Pour des raisons de compatibilité, le mot de passe chiffré de l'utilisateur peut alternativement être stocké dans l'attribut userPassword qui peut être assigné à une entrée posixUserAccount.

Il est prévu pour supporter les configurations existantes uniquement et ne devrait pas être utilisé pour les nouvelles entrées, qui devraient utiliser authPassword.

La représentation chaîne de l'attribut userPassword devrait matcher la grammaire suivante, spécifiée en notation ABNF :

```
scheme = "crypt" / "md5" / "sha" / "ssha" / altscheme
altscheme = "x-" keystring
userPassword = LCURLY scheme RCURLY cryptpass
```

où "cryptpass" représente le mot de passe hashé par l'algorithme spécifié. Si le schéma est "sha", le SHA-1 du mot de passe est calculé, et le mot de passe chiffré devrait être encodé en base64.

Bien qu'un DUA peut implémenter n'importe quel schéma supporté par le DSA, il doit supporter le schéma "crypt" pour des raisons de compatibilité. Cependant, il est recommandé qu'un schéma plus sécurisé soit utilisé.

Si l'attribut userPassword a plus d'un seul valeur, le DUA devrait sélectionner un mot de passe basé sur le schéma d'authentification le plus fort qu'il supporte. Si l'authentification échoue, le DUA ne devrait pas tenter d'utiliser d'autres valeurs. Si l'attribut n'utilise pas de schéma supporté par le DUA, l'authentification doit échouer.

loginShell

Le chemin du répertoire personnel de l'utilisateur est stocké dans l'attribut loginShell qui peut être assigné à une entrée posixUserAccount :

```
attributetype ( 1.3.6.1.1.1.1.4 NAME 'loginShell' DESC 'The path to the login shell' EQUALITY
caseExactIA5Match SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 SINGLE-VALUE )
```

Si le loginShell est manquant, cet utilisateur ne sera pas capable de se connecter sur un hôte ou un service qui nécessite un shell unix.

disableObject

Un compte utilisateur peut être désactivé en définissant l'attribut disableObject à TRUE. Si une entrée est désactivée, le DUA devrait se comporter comme si cet utilisateur n'existait pas. Le DUA peut optionnellement fournir un mécanisme séparé pour lister les entrées désactivées, mais elles doivent être clairement marquées désactivées pour ne pas créer de confusion.

Exemple d'entrée passwd

L'exemple suivant est un exemple d'entrée posixUserAccount au format LDIF :

```
dn: en=mark,ou=passwd,ou=sales,o=infra
objectClass: top
objectClass: inetOrgPerson
objectClass: posixUserAccount
cn: Mark
sn: Bannister
displayName: Bannister, Mark
en: mark
uidNumber: 101
gidNumber: 900
homeDirectory: /home/mark
loginShell: /bin/bash
```

groupes

Une base group contient les détails suivants :

- Le nom du groupe
- Le mot de passe du groupe
- Un identifiant numérique pour le groupe
- La liste des membres

ObjectClass

Une entrée dbisMapConfig pour une base group devrait avoir l'objectClass dbisGroupConfig. Une entrée group devrait être définie par une entrée avec l'objectClass posixGroupAccount.

dbisGroupConfig

La classe dbisGroupConfig est définie comme suit :

```
objectclass ( 1.3.6.1.4.1.23780.219.1.11 NAME 'dbisGroupConfig' DESC 'DBIS group configuration map' SUP
dbisMapConfig STRUCTURAL MAY dbisOverlayDN )
```

posixGroupAccount

La classe posixGroupAccount est définie comme suit :

```
objectclass ( 1.3.6.1.4.1.23780.219.1.12 NAME 'posixGroupAccount' DESC 'Group account with POSIX attributes'
SUP top STRUCTURAL MUST ( en $ gidNumber ) MAY ( authPassword $ userPassword $ exactUser $ uniqueMember $
description $ manager $ disableObject ) )
```

en

Le nom du compte groupe est stocké dans l'attribut en. Cet attribut doit être associé avec une entrée posixGroupAccount et devrait former le RDN.

gidNumber

Le GID est socké dans l'attribut gidNumber qui doit être assigné à une entrée posixGroupAccount

authPassword

Le mot de passe chiffré du groupe est stocké dans l'attribut authPassword qui peut être assigné à une entrée posixGroupAccount. Toutes les considérations liées à l'attribut authPassword données dans la section de même nom pour la base passwd s'applique également aux entrées posixGroupAccount. Si un attribut authPassword est définis, l'utilisateur doit fournir le bon mot de passe avant que le DUA autorise la bascule dans ce groupe

userPassword

Pour des raisons de compatibilité, le mot de passe chiffré du groupe peut alternativement être stocké dans l'attribut userPassword et peut être assigné à une entrée posixGroupAccount. Toutes les considérations liées à l'attribut userPassword données dans la section de même nom pour la base passwd s'applique également aux entrées posixGroupAccount.

exactUser

Une liste d'un ou plusieurs noms de comptes qui sont membres du groupe sont stockés dans les attributs exactUser qui peuvent être assignés à une entrée posixGroupAccount :

```
attributetype ( 1.3.6.1.4.1.23780.219.2.26 NAME 'exactUser' DESC 'One or more user account names' EQUALITY
caseExactMatch SUBSTR caseExactSubstringsMatch SYNTAX 1.3.6.1.4.1.1466.115.121.1.15{32768} )
```

uniqueMember

Pour des raisons de compatibilité, les membres du groupe peuvent alternativement être stockés dans l'attribut uniqueMember et peut être assigné à une entrée posixGroupAccount. Il est prévu pour supporter les configurations existantes uniquement et ne devrait pas être utilisé pour les nouvelles entrées. Un DUA doit supporter les 2 format.

Les membres référés par l'attribut uniqueMember devrait être assumés avoir été présentés via la map de configuration existante, même s'ils sont présentés dans un DN de base différent. L'attribut uniqueMember n'est donc pas prévu pour référencer les utilisateurs ou groupes qui sont définis avec un schéma différent. L'attribut exactUser ne souffre pas de ce problème.

description

L'attribut description peut être associé avec une entrée posixGroupAccount pour fournir une description arbitraire de l'entrée.

manager

L'attribut manager peut être associé avec une entrée posixGroupAccount pour fournir un ou plusieurs DN d'individus, groupes ou systèmes qui sont responsables de la maintenance de l'entrée.

disableObject

Un goupe peut être désactivé en définissant l'attribut disableObject à TRUE. Si une entrée est désactivée, le DUA devrait se comporter comme si le groupe n'existait pas. Le DUA peut optionnellement fournir un mécanisme séparé pour lister les entrées désactivées, mais doivent les marquer clairement comme désactivé pour éviter toute confusion.

Exemple d'entrée groupe

L'exemple suivant est une entrée posixGroupAccount au format LDIF :

```
dn: en=finance,ou=group,ou=sales,o=infra
objectClass: top
objectClass: posixGroupAccount
en: finance
gidNumber: 152
exactUser: mark
exactUser: julie
exactUser: stephen
exactUser: nathan
```

Overlays

Les overlays fournissent les entrées passwd et group alternatifs qui peuvent écraser l'UID, GID, Home ou shell pour les groupes d'hôtes qui partagent une map de configuration. C'est utile pour fusionner 2 domaines DBIS avec des ID qui se chevauchent en permettant une période de transition quand les hôtes et services du domaine d'origine peut continuer à utiliser leur ID et shell originaux. Un DUA devrait implémenter les overlays.

Considérons l'exemple où UserA et UserB ont les UID 100 et 101, et le shell /bin/sh sur HostA et HostB, mais nécessitent les UID 1000 et 1001 et le shell /bin/csh sur HostC et HostD. Les 4 hôtes sont membre du même domaine DBIS. Les overlays permettent ce type de configuration. Un DUA devrait traiter les overlays après toute transformation définie dans la map de configuration.

ObjectClass

Le DN top-level sous lequel rechercher les entrées overlay devraient être définis par l'attribut dbisOverlayDN qui est associé avec une entrée dbisPasswdConfig ou dbisGroupConfig. Les entrées overlay doivent résider sous ce DN s'ils sont utilisé par un DUA.

Les entrées overlay pour la base passwd sont identifiées par l'objectClass dbisPasswdOverlay. Les entrées overlay pour la base group sont identifiés par l'objectClass dbisGroupOverlay.

dbisPasswdOverlay

La classe dbisPasswdOverlay est définie comme suit :

```
objectclass ( 1.3.6.1.4.1.23780.219.1.13 NAME 'dbisPasswdOverlay' DESC 'User account overlay entry' SUP top
STRUCTURAL MUST en MAY ( uidNumber $ homeDirectory $ loginShell $ description $ disableObject ) )
```

dbisGroupOverlay

La classe dbisGroupOverlay est définie comme suit :

```
objectclass ( 1.3.6.1.4.1.23780.219.1.14 NAME 'dbisGroupOverlay' DESC 'Group account overlay entry' SUP top
STRUCTURAL MUST en MAY ( gidNumber $ description $ disableObject ) )
```

Attributs

en

L'attribut en doit être assigné à une entrée dbisPasswdOverlay ou dbisGroupOverlay et est utilisé pour identifier l'entrée posixUserAccount ou posixGroupAccount. Si les attributs en correspondent exactement, ou si c'est un dbisPasswdOverlay et qu'il n'y a pas de correspondance exacte mais une entrée pas défaut existe identifié par un attribut en contenant un asterisk, les attributs fournis par l'overlay devraient remplacer ceux dans l'entrée originale.

Quand un DUA recherche une entrée posixUserAccount ou posixGroupAccount qui a une configuration d'overlay, il devrait également rechercher une entrée dbisPasswdOverlay ou dbisGroupOverlay.

Si une entrée par défaut est trouvée (par ex : en=*), l'attribut uidNumber est ignoré s'il est assigné à l'objet dbisPasswdOverlay.

uidNumber

Un UID alternatif à utiliser pour un compte utilisateur correspondant peut être stocké dans l'attribut uidNumber qui peut être associé avec une entrée dbisPasswdOverlay

gidNumber

Un GID alternatif à utiliser pour un groupe correspondant peut être stocké dans l'attribut gidNumber qui peut être associé avec une entrée dbisGroupOverlay

HomeDirectory

Un répertoire personnel alternatif à utiliser pour un compte utilisateur correspondant est stocké dans l'attribut homeDirectory qui peut être associé avec une entrée dbisPasswdOverlay.

loginShell

Un shell de connexion alternatif à utiliser pour un compte utilisateur correspondante est stocké dans l'attribut loginShell qui peut être associé avec une entrée dbisPasswdOverlay.

description

L'attribut description peut être associé avec une entrée dbisPasswdOverlay ou dbisGroupOverlay pour fournir une description arbitraire de l'entrée.

disableObject

Une entrée overlay peut être désactivée en définissant l'attribut disableObject à TRUE. Si une entrée est désactivé, le DUA devrait se comporter comme si l'overlay n'existait pas. Le DUA peut optionnellement fournir un mécanisme séparé pour lister les entrées désactivées, mais ils doivent les marquer clairement comme désactivé pour éviter toute confusion.

Exemple d'entrées overlay

L'exemple suivant montre un entrée dbisPasswdOverlay au format ldif, et les entrée dbisMapConfig correspondantes. Dans cet exemple, l'utilisateur "julie" qui se log sur des hôtes qui font partie du netgroup "sales-merger" aura un UID alternatif 5001 et un shell "/bin/sh". Si "julie" se logs sur un autre hôte, elle aura son UID et loginShell normal :

```
dn: cn=passwd,en=sales.corp,ou=domain-mappings,o=infra
objectClass: top
objectClass: dbisMapConfig
objectClass: dbisPasswdConfig
cn: passwd
dbisMapDN: cn=passwd,ou=dbis,o=infra
dbisMapFilter: objectClass=posixUserAccount
dbisMapGecos: displayName
notNetgroup: sales-merger
profileTTL: 900
description: Primary passwd database

dn: cn=passwd2,en=sales.corp,ou=domain-mappings,o=infra
```

```
objectClass: top
objectClass: dbisMapConfig
objectClass: dbisPasswdConfig
cn: passwd2
dbisMapDN: cn=passwd,ou=dbis,o=infra
dbisMapFilter: objectClass=posixUserAccount
dbisMapGecos: displayName
dbisOverlayDN: ou=passwd,ou=overlays,ou=sales-merger,o=infra
profileTTL: 900
description: Primary passwd database for Sales merger

dn: en=julie,ou=passwd,ou=overlays,ou=sales-merger,o=infra
objectClass: top
objectClass: dbisPasswdOverlay
en: julie
uidNumber: 5001
loginShell: /bin/sh
```

L'exemple suivant montre un dbisGroupOverlay qui modifie le GID pour le groupe "finance" quand il est utilisé dans une entrée de map de configuration :

```
dn: en=finance,ou=group,ou=overlays,ou=sales-merger,o=infra
objectClass: top
objectClass: dbisGroupOverlay
en: finance
gidNumber: 7308
```

Syntaxe d'attributs

Les syntaxes suivantes sont utilisée par les attributs définis dans ce document :

```
Syntax OID - - - - - - - - - - Value - - - - - - Reference
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
1.3.6.1.4.1.1466.115.121.1.12 DN - - - - - - - - [RFC4517]
1.3.6.1.4.1.1466.115.121.1.15 Directory String -[RFC4517]
1.3.6.1.4.1.1466.115.121.1.26 IA5 String - - - -[RFC4517]
1.3.6.1.4.1.1466.115.121.1.27 Integer - - - - - [RFC4517]
- - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
```

Notes d'implémentation

Mappage des champs compatibles NIS

Tous les champs requis pour le format des bases NIS passwd et group existent dans ce schéma et peuvent être mappés aux types d'attribut en utilisant les production ABNF décrites dans draft-bannister-dbis-netgroup.

passwd

Les champs de la base passwd sont mappés comme suit :

```
user = en
password = %x78 ; lowercase "x", see below
uid = uidNumber
gid = gidNumber
gecos = dbisMapGecos ; derived, see below
homedir = homeDirectory
loginshell = loginShell

passwd-entry = user COLON password COLON uid COLON gid COLON gecos COLON homedir COLON loginshell
```

- password vaux "x" qui signifie traditionnellement que le mot de passe est stocké dans la base shadow. Cependant, c'est dûs aux permission plus stricts du fichier shadow, rendant les mots de passe plus sécurisés. LDAP n'a pas ce problème, l'attribut authPassword est associé avec l'objectClass posixUserAccount. Un implémenteur peut cependant optionnellement reporter le mot de passe chiffré dans les entrées passwd NIS, ou pas du tout. Pour des raisons de sécurité il est recommandé que les utilisateurs ne puissent pas afficher leur mot de passe chiffré à d'autres utilisateurs ou groupe.
- gecos est déterminé en recherchant l'attribut définis par l'attribut dbisMapGecos donné dans l'entrée de map de configuration.

group

Les champs de la base group sont mappés comme suit :

```
group = en
password = %x2a ; asterisk "*", see below
gid = gidNumber
users = exactUser ; derived, see below

group-entry = group COLON password COLON gid COLON users
```

- Pour des raisons de sécurité il est recommandé que le mot de passe soit mis à "*" et pas de authPassword.
- La liste des utilisateur est une liste séparé par des virgules d'attributs exactUser.

Filtres de recherche communs

Cette section fournis des exemples de filtres de recherche LDAP pour obtenir des entrées de base avec des critères d'entrée communément utilisés.

Pour simplifier les exemples, toutes les base sont assumée être définis avec seulement une entrée de map de configuration. Cependant, une implémentation doit le supporter, augmentant le nombre d'opérations nécessaires pour localiser les entrées de base dans le scope.

Le DN de base utilisé dans les opérations de recherche décrites dans cette section viennent de l'attribut dbisMapDN assigné à l'entrée dbisMapConfig. Noter qu'une entrée dbisMapConfig peut en avoir plusieurs.

Quand il apparaît dans les filtres de recherche ci-dessous, le texte "dbisMapFilter" réfère à la valeur assignée à l'attribut de même nom dans l'entrée dbisMapConfig correspondante. Noter que les bases passwd et group ont des entrées dbisMapConfig différentes. Les noms d'attributs utilisés dans ces filtres de recherche peuvent être modifiés par l'attribut dbisMapAttr assignés à l'entrée dbisMapConfig.

Pour localiser la map de configuration pour un domain DBIS donné, rechercher les entrées sous l'entrée dbisDomainObject :
(&(objectClass=dbisPasswdConfig)(!(disableObject=TRUE)))
(&(objectClass=dbisGroupConfig)(!(disableObject=TRUE)))
Les entrées passwd et group sont énumérées en appliquant le filtre dbisMapFilter suivant :

(&(dbisMapFilter)(!(disableObject=TRUE)))
Si une entrée passwd et group est connue pas "name", sa définition est localisée en utilisant le filtre de recherche suivant :
(&(dbisMapFilter)(!(disableObject=TRUE))(en=name))
Si une entrée passwd a l'uid "uid", sa définition est localisée en utilisant le filtre suivant :
(&(dbisMapFilter)(!(disableObject=TRUE))(uidNumber=uid))
Si une entrée group a le gid "gid", sa définition est localisée en utilisant le filtre suivant :
(&(dbisMapFilter)(!(disableObject=TRUE))(gidNumber=gid))