Manuel d’administration du serveur ownCloud

version 9.1

Les développeurs d’ownCloud

21 October 2017

Sommaire

Table des matières

Introduction au manuel d'administration du serveur ownCloud 9.1

Introduction

Bienvenue dans le guide d'administration du serveur ownCloud. Ce guide décrit les tâches d'administration d'ownCloud, la solution de partage et de synchronisation open source. ownCloud contient le serveur ownCloud, qio fonctionne sous GNU/Linux, des applications clientes pour Microsoft Windows, Mac OS X et Linux, et des clients mobiles pour les systèmes d'exploitation Android et Apple.

Les versions actuelles des manuels d'ownCloud sont disponibles en ligne sur doc.owncloud.org et doc.owncloud.com.

Le serveur ownCloud est disponible en trois éditions :

- le serveur libre supporté par la communauté. C'est la base serveur pour toutes les éditions ;
- l'Édition standard pour les clients qui veulent un support payant pour le serveur, sans les applications Entreprise ;
- l'Édition Entreprise qui fournit un support payant pour le serveur et les applications Entreprise.

Consulter *Nouveautés pour les administrateurs dans ownCloud 9.1* pour plus d'informations sur les différentes éditions d'ownCloud.

Vidéos et blogs d'ownCloud

Consulter le canal officiel d'ownCloud et le canal communautaire des ownClouders sur YouTube pour des tutoriels, aperçus ou des vidéos de conférences.

Visiter ownCloud Planet pour des nouvelles et les blogs des développeurs.

Public visé

Ce guide est pour les utilisateurs qui veulent installer, administrer et optimiser leur serveur ownCloud. Pour en apprendre plus sur l'interface utilisateur Web d'ownCloud, sur les clients pour PC et mobiles, veuillez consulter les manuels respectifs :

- Manuel utilisateur d'ownCloud
- Client pour ordinateur ownCloud
- Application Android ownCloud
- Application iOS ownCloud

Notes de version ownCloud 9.1

Changements dans la version 9.1

Général

- Les tâches de fond (cron) peuvent maintenant être exécutées en parallèles.
- Les notifications de mise à jour dans le client via l'API. Vous pouvez maintenant être notifié dans votre client pour ordinateur des mises à jour disponibles pour le cœur et les applications. Les notifications sont rendues disponibles par l'API de notifications.
- Support multi-bucket pour l'intégration des primary objectstore.
- Le support pour Internet Explorer en dessous de la version 11 n'est plus assuré.
- Les liens symboliques pointant en dehors du répertoire de données ne sont plus autorisés. Veuillez consulter *Configuration du stockage externe (interface graphique)* et *Local*.

- Suppression des commandes `dav:migrate-calendars` et `dav:migrate-addressbooks` pour `occ`. Les utilisateurs prévoyant de mettre à jour à partir de ownCloud 9.0 ou d'une version précédente vers ownCloud 9.1 doivent s'assurer que leurs agendas et carnets d'adresses soient correctement migrés **avant** de continuer la mise à jour vers la version 9.1.

Authentification

- Authentification par plugin : plugin système gérant différents schémas d'authentification.
- Authentification basée sur les jetons.
- Possibilité d'invalider des sessions.
- Liste de navigateurs/équipements connectés dans la page des paramètres personnels. Permet à l'utilisateur de déconnecter les navigateurs en session ou les équipements.
- Des jetons/mots de passe spécifiques par équipement peuvent être générés dans la page des paramètres personnels et révoqués.
- Désactivation des utilisateurs et révocation automatique de leurs sessions.
- Détection des utilisateurs LDAP désactivés ou des changements de mots de passe et révocation de leurs sessions.
- Connexion à l'aide de l'adresse électronique.
- Option de configuration pour forcer la connexion par jeton en dehors de l'interface Web.
- Plugin système d'authentification à deux facteurs.
- Commande OCC ajoutée pour activer/désactiver (temporairement) l'authentification à deux facteurs pour des utilisateurs spécifiques.

> ***Note***
>
> les versions actuelles pour ordinateurs et pour mobiles ne gèrent pas encore l'authentification à deux facteurs, ce sera ajouté pour plus tard. Il est déjà possible de générer un mot de passe spécifique pour un équipement et de la saisir avec les versions de clients actuelles.

Application Fichiers

- Possibilité d'afficher/masquer les fichiers cachés.
- Enregistrement de l'ordre de tri.
- Liens permanents pour les partages internes.
- Trace visuelle lors du glissement de fichiers dans l'application Fichiers.
- Défilement de la liste des fichiers lors du glissement de fichiers dans l'application.
- Estimation de la progression du téléversement de fichiers.

Partage fédéré

- Possibilité de créer des partages fédérés avec des permissions CRUDS.
- Le re-partage de partage fédéré ne crée plus de chaîne de partages mais connecte le serveur du propriétaire du partage au destinataire du re-partage.

Stockage externe

- Gestion de la compatibilité d'encodage UTF-8 NFD pour les noms de fichiers NFD stockés directement sur les stockages externes (nouvelle option de montage dans la page d'administration des stockages externes).
- Liens directs vers les pages de configuration pour paramétrer une application GDrive ou Dropbox avec ownCloud.
- Améliorations des performances et de l'utilisation mémoire pour GDrive avec le téléchargement par flux ou par morceaux.
- Améliorations des performances et de l'utilisation mémoire pour Dropbox avec le téléchargement par flux.

- Mise à jour de la bibliothèque GDrive library limitant les erreurs de limites de débit.

Additions mineures

- Gestion des feuilles de style d'impression.
- La mise à jour par ligne de commande sera maintenant suggérée si l'instance est trop grosse pour éviter de potentiels dépassements de temporisation.
- La mise à jour par le Web sera désactivée si LDAP ou Shibboleth sont installés.
- Le processus de mise à jour de l'application/base de données affiche désormais plus d'informations.
- Ajout de l'option occ files:scan --unscanned pour analyser uniquement les dossiers qui n'ont pas été explorés sur les stockages externes.
- Le TTL du cache peut maintenant être configuré.
- Ajout d'avertissements pour les trabsactions de base de données mal configurées, aidant à empêcher les problèmes « database is locked ».
- Utilisation d'un cache mémoire pour réduire l'utilisation de la mémoire, spécialement pour les tâches de fond et le scanner de fichiers.
- Possibilité de se connecter avec une adresse électronique.
- Respect de la propriété CLASS dans les evénements d'agenda.
- Possibilité d'exporter un carnet d'adresses en utilisant VCFExportPlugin.
- Les anniversaires sont aussi générés sur la base des agendas partagés.

Pour les développeurs

- Nouvelle terminaison DAV avec un nouveau protocole destiné à résoudre beaucoup de problèmes comme les temporisations (pas encore utilisé par les clients).
- Nouvelle propriété webdav pour les permissions de partage.
- Les étapes de réparation en arrière-plan peuvent être spécifiées dans info.xml.
- Les tâches de fond (cron) peuvent maintenant être déclarées dans info.xml.
- Les applications peuvent maintenant définir les étapes de réparation au moment de l'installation/désinstallation.
- Export des images de contacts à l'aide du plugin Sabre.
- Le plugin de navigateur Sabre DAV est disponible en mode debug pour permettre un développement plus facile autour de webdav.

Dette technique

- Auto-chargement forcé de PSR-4 pour OCet OCP, facultatif pour OCA. Documentation sur https://doc.owncloud.org/server/9.1/developer_manual/app/classloader.html
- Nettoyage supplémentaire dans le code de partage (en cours)

Changements dans la version 9.0

La version 9.0 nécessite des fichiers .ico pour les favicônes. Ceci sera modifié dans la version 9.1 qui utilisera des fichiers .svg. Consulter Changing favicon dans le manuel du développeur.

La règle du dossier Home est mise en oeuvre dans l'application user_ldap dans les nouvelles installation d'ownCloud. Consulter *Authentification utilisateur avec LDAP*. Ceci affecte les versions ownCloud 8.0.10, 8.1.5 et 8.2.0 et suivantes.

Les applications Agenda et Contacts ont été réécrites et les accès CalDAV et CardDAV de ces applications ont été fusionnés dans le code d'ownCloud. Pendant la mise à jour, les agendas et carnets d'adresses existants sont automatiquement migrés (excepté lors de l'utilisation de `IMAP user backend`). Alternativement, lorsque la mise à jour échoue lors de l'utilisation de `IMAP user backend` ou pour tester une migration, les scripts `dav:migrate-calendars` et/ou `dav:migrate-addressbooks` sont disponibles (**seulement pour ownCloud 9.0**) avec la commande `occ`. Voir *Utilisation de la commande occ*.

La mise à jour sur des systèmes avec de grands volumes de données prendront plus de temps en raison de l'addition de la vérification des sommes de contrôles dans la base de données d'ownCloud. Consulter https://github.com/owncloud/core/issues/22747.

Les paquets GNU/Linux sont disponibles sur notre dépôt officiel . Nouveau dans la version 9.0 : les paquets séparés. `owncloud` installe ownCloud plus les dépendances, y compris Apache et PHP. `owncloud-files` installe seulement ownCloud. Ceci est utile pour les stacks LAMP personnalisés et vous permet d'installer vos propres applications et versions LAMP en évitant les conflits de paquets avec ownCloud. Consulter *Méthode d'installation préférée pour Linux*.

Une nouvelle option pour les administrateurs d'ownCloud permet d'activer ou de désactiver le partage sur des points de montage externes individuels (consulter *Options de montage*). Le partage sur de tels points de montage est désactivé par défaut.

Édition Entreprise 9.0

owncloud-enterprise packages are no longer available for CentOS6, RHEL6,

Debian7, or any version of Fedora. A new package, owncloud-enterprise-files, is available for all supported platforms, including the above. This new package comes without dependencies, and is installable on a larger number of platforms. System administrators must install their own LAMP stacks and databases. See https://owncloud.org/blog/time-to-upgrade-to-owncloud-9-0/ .

Nouveautés pour les administrateurs dans ownCloud 9.1

Consulter la page des fonctionnalités d'ownCloud 9.1 sur Github pour une liste complète des nouvelles fonctionnalités et mises à jour.

ownCloud a de nombreuses améliorations. Parmi lesquelles :

- Un système de plugin d'authentification à deux facteurs.
- Ajout d'une commande OCC pour activer/désactiver (temporairement) l'authentification à deux facteurs pour des utilisateurs spécifiques.

Note : les versions actuelles pour ordinateurs et pour mobiles ne gèrent pas encore l'authentification à deux facteurs, ce sera ajouté pour plus tard. Il est déjà possible de générer un mot de passe spécifique pour un équipement et de la saisir avec les versions de clients actuelles.

- Nouvelle option `occ`, `--unscanned`, pour scanner seulement les fichiers n'ayant pas été scannés précédemment (https://github.com/owncloud/core/pull/24702).
- Nouvelle commande `occ` pour activer/désactiver les utilisateurs.
- Nouvelle commande `occ` pour activer/désactiver l'authentification à deux facteurs pour des utilisateurs spécifiques.
- Nouvelles étiquettes de groupes pour les étiquettes de fichiers système (https://github.com/owncloud/enterprise/issues/1208)
- Nettoyeur d'URL pour les fichiers internes (https://github.com/owncloud/core/issues/11732)
- Liste déroulante de configuration pour Google Drive/Dropbox. Configuration facilitée pour les hébergeurs (https://github.com/owncloud/core/pull/22214).

Édition Entreprise

- ownCloud 9.1 Enterprise can now handle notifications directly from Windows network drives as a technology preview, making complete file scans obsolete.
- The Workflow engine represents a powerful new feature for enterprise customers. Triggers can be placed on new or modified files, while scripts of any type can be run. This allows documents to be converted automatically into PDFs or sent to a specific recipient by e-mail. In conjunction with the retention app, the file firewall and integrated tagging, it also allows numerous workflows to be addressed for integration into business processes.

Installation

Prérequis système

Mémoire

Les prérequis mémoire pour exécuter un serveur ownCloud sont très variables et dépendent du nombre d'utilisateurs et de fichiers, et du volume d'activité. ownCloud a besoin au minimum de 128 Mo RAM et nous recommandons d'utiliser au moins 512 Mo.

> **Note**
>
> *Considérations pour des environnements ayant peu de mémoire*
>
> L'analyse (scan) des fichiers est faite par tranche de 10 000 fichiers. Nos tests ont montré que l'utilisation mémoire pour l'analyse de plus de 10 000 fichiers consommait 75 Mo de mémoire additionnelle.

Configuration recommandée pour exécuter ownCloud

Pour de *meilleures performances*, *stabilité*, *support* et *intégralité des fonctionnalités*, nous recommandons :

Plateforme	Options
Système d'exploitation	Système d'exploitation Ubuntu 16.04
Base de données	MySQL ou MariaDB 5.5+ avec moteur de stockage InnoDB
Serveur Web	Apache 2.4 avec `prefork` *Module Multi-Processing (MPM)* et `mod_php`
Runtime PHP	PHP (5.4, 5.5, 5.6 et 7.0)

Plateformes supportées

Si vous ne pouvez utiliser un ou plusieurs des outils ci-dessus, les options suivantes sont également supportées.

Serveur

- Debian 7 et 8
- SUSE Linux Enterprise Server 12 et 12 SP1
- Red Hat Enterprise Linux/Centos 6.5 et 7 (**7 seulement en 64 bits**)
- Ubuntu 14.04 LTS

Serveur Web

- Apache 2.4 avec mod_php

Bases de données

- Oracle 11g (**Édition Entreprise seulement**)
- PostgreSQL

Hyperviseurs

- Hyper-V
- VMware ESX

- Xen
- KVM

PC de bureau

- Windows 7+
- Mac OS X 10.7+ (seulement en 64 bits)
- Ubuntu 16.10
- Ubuntu 16.04
- Ubuntu 14.04
- Debian 7.0
- Debian 8.0
- CentOS 7
- Fedora 24
- Fedora 25
- openSUSE Leap 42.1
- openSUSE Leap 42.2

Note

Pour les distributions Linux, nous supportons, si c'est techniquement possible, les deux dernières versions par plateforme et la version LTS précédente.

Mobile

- iOS 7+
- Android 4+

Navigateur Web

- IE11+ (sauf mode de compatibilité)
- Firefox 14+
- Chrome 18+
- Safari 5+

Consulter *Manuel d'installation pour Linux* pour les versions logicielles minimum pour installer ownCloud.

Prérequis de base de données pour MySQL / MariaDB

Ce qui suit est obligatoire si vous utilisez ownCloud avec une base de données MySQL / MariaDB :

- désactiver `BINLOG_FORMAT = MIXED` ou `BINLOG_FORMAT = ROW` (voir : *MySQL / MariaDB avec activation des journaux binaires*) ;
- utiliser le moteur de stockage InnoDB (le moteur de stockage MyISAM n'est pas supporté, voir : *MySQL / MariaDB storage engine*) ;
- utiliser le niveau d'isolation de transaction « READ COMMITED » (voir : *Moteur de stockage MySQL / MariaDB niveau d'isolation de transaction « READ COMMITED »*).

Recommandations de déploiement d'ownCloud

Quel est le meilleur moyen pour installer et maintenir ownCloud ? La réponse à cette question est : *« Ça dépend »* car chaque client ownCloud a ses propres besoins et infrastructure. ownCloud et l'ensemble LAMP sont hautement configurables, nous présenterons donc trois scénarios typiques et ferons des recommandations pour le matériel et les logiciels.

Recommandations générales

Note

Quelle que soit la taille de votre organisation, gardez toujours ceci à l'esprit : la quantité de données stockées dans ownCloud ne fera qu'augmenter. Planifiez en conséquence.

Vous pouvez envisager de définir une infrastructure pour gérer la montée en charge ou utiliser la fédération de serveurs ownCloud pour conserver des instances ownCloud de taille gérable.

- Système d'exploitation : Ubuntu 16.04 LTS.
- Serveur Web : Apache 2.4.
- Base de données : MySQL/MariaDB avec le moteur de stockage InnoDB (MyISAM n'est plus supporté, voir : *MySQL / MariaDB storage engine*)
- PHP 7.

Petits groupes de travail et départements

- **Nombre d'utilisateurs**

  Jusqu'à 150 utilisateurs.

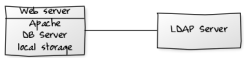
- **Haute disponibilité**

  Pas de coupure de service avec des instantanés (snapshots) de système de fichiers, interruption de service si défaillance matérielle. Plan de sauvegarde alternatif : sauvegarde de nuit avec interruption de service.

Prérequis système

Une seule machine exécutant le serveur applicatif, le serveur Web, la base de données et le stockage local.

Authentification à l'aide d'un serveur LDAP existant ou Active Directory.



- **Composants**

  Un seul serveur avec au moins deux cœurs, 16 Go de RAM, stockage local.

- **Système d'exploitation**

  Distribution Linux entreprise avec support du vendeur. Nous recommandons Ubuntu 16.04 LTS. D'autres distributions sont aussi supportées (par ex. : RedHat ou SuSE) mais elles peuvent ne pas fournir toutes les dépendances requises dans leurs dépôts officiels et il pourrait être nécessaire d'activer des dépôts tiers pour des modules comme APCu et Redis.

- **Configuration SSL**

  La terminaison SSL est réalisée dans Apache. Un certificat SSL standard est nécessaire, installé selon les recommandation de la documentation Apache.

- **Équilibrage de charge**

  Aucun.

- **Base de données**

  Nous recommandons actuellement MySQL / MariaDB, car nos clients ont de bonnes expériences avec ceux-ci en passant sur une cluster Galera pour absorber la montée en charge de la base de données. (Moteur de stockage InnoDB, MyISAM n'est pas supporté, voir *MySQL / MariaDB storage engine*).

- **Authentification**

  Authentification à l'aide d'un ou plusieurs serveurs LDAP ou Active Directory. Voir Authentification utilisateur avec LDAP pour des informations sur la configuration d'ownCloud pour utiliser LDAP et Active Directory.

- **Gestion de session**

  Gestion de session locale sur le serveur applicatif. Les sessions PHP sont stockées sur une partition tmpfs montée sur un emplacement spécifique du système d'exploitation. Vous pouvez trouver cet emplacement en exécutant la commande `grep -R 'session.save_path' /etc/php5` et ajoutez-le au fichier `/etc/fstab`. Par exemple : `echo "tmpfs /var/lib/php5/pool-www tmpfs defaults,noatime,mode=1777 0 0" >> /etc/fstab`.

- **Mémoire cache**

  Nous recommandons : - APC/APCu pour le cache local. - Redis pour le verrouillage de fichiers transactionnel et le cache distribué, sur un serveur dédié. Un système de mémoire cache améliore les performances du serveur. ownCloud sait gérer 4 systèmes de mémoire cache. Veuillez consulter Configuration de la mémoire cache pour des informations sur le choix et la configuration d'un système de mémoire cache.

- **Stockage**

  Stockage local.

Entreprises moyennes

- **Nombre d'utilisateurs**

  De 150 à 1 000 utilisateurs.

- **Haute disponibilité**

  Chaque composant est totalement redondé et peut s'arrêter sans interruption de service. Sauvegardes sans interruption de service.

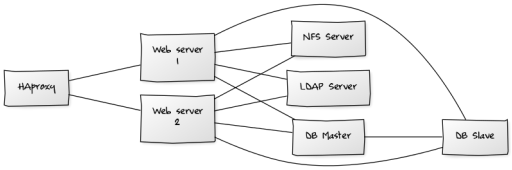
Prérequis serveurs recommandés

De 2 à 4 serveurs applicatifs.

Un cluster de deux serveurs de bases de données.

Stockage sur un serveur NFS ou un magasin d'objets compatible avec S3.

Authentification à laide d'un serveur LDAP ou Active Directory existant.



- **Composants**

  - de 2 à 4 serveurs applicatifs avec 4 sockets et 32 Go de RAM.
  - 2 serveurs de base de données avec 4 sockets et 32 Go de RAM.
  - Un équilibreur de charge HAproxy avec 2 sockets et 16 Go de RAM.
  - Un serveur de stockage NFS.

- **Système d'exploitation**

Distribution Linux entreprise avec support du vendeur. Nous recommandons Ubuntu 16.04 LTS. D'autres distributions sont aussi supportées (par ex. : RedHat ou SuSE) mais elles peuvent ne pas fournir toutes les dépendances requises dans leurs dépôts officiels et il pourrait être nécessaire d'activer des dépôts tiers pour des modules comme APCu et Redis.

- **Configuration SSL**

  La terminaison SSL est réalisée sur l'équilibreur de charge HAProxy. Un certificat standard SSL est nécessaire, installé selon la documentation HAProxy.

- **Équilibrage de charge**

  HAProxy s'exécutant sur un serveur dédié en frontal des serveurs applicatifs. L'affinité de session doit être utilisée à cause de la gestion locale de session sur les serveurs applicatif.

- **Base de données**

  Nous recommandons MySQL/MariaDB, avec un cluster Galera avec réplication maître-maître ou une configuration maître-esclave avec bascule automatique. (Moteur de stockage InnoDB, MyISAM n'est pas supporté, voir *MySQL / MariaDB storage engine*).

- **Authentification**

  Authentification à l'aide d'un ou plusieurs serveurs LDAP ou Active Directory. Voir Authentification utilisateur avec LDAP pour des informations sur la configuration d'ownCloud pour utiliser LDAP ou Active Directory.

- **LDAP**

  Des serveurs esclaves en lecture seule doivent être déployés pour chaque serveur applicatif pour une montée en charge optimale.

- **Gestion de session**

  Gestion de session locale sur le serveur applicatif. Les sessions PHP sont stockées sur une partition tmpfs montée sur un emplacement spécifique du système d'exploitation. Vous pouvez trouver cet emplacement en exécutant la commande `grep -R 'session.save_path' /etc/php5` et ajoutez-le au fichier `/etc/fstab`. Par exemple : `echo "tmpfs /var/lib/php5/pool-www tmpfs defaults,noatime,mode=1777 0 0" >> /etc/fstab`.

- **Mémoire cache**

  - APC/APCu pour le cache local.
  - Redis pour le verrouillage de fichiers transactionnel et le cache distribué, sur un serveur dédié.

  Un système de mémoire cache améliore les performances du serveur. ownCloud sait gérer 4 systèmes de mémoire cache. Veuillez consulter Configuration de la mémoire cache pour des informations sur le choix et la configuration d'un système de mémoire cache.

- **Stockage**

  Utiliser une solution NFS toute faite comme IBM Elastic Storage ou RedHat Ceph.

Grandes entreprises et fournisseurs de services

- **Nombre d'utilisateurs**

  de 5 000 à plus de 100 000 utilisateurs.

- **Haute disponiblité**

  Chaque composant est totalement redondant et peut s'arrêter sans interruption de service. Sauvegardes sans interruption de service.

Prérequis système recommandés

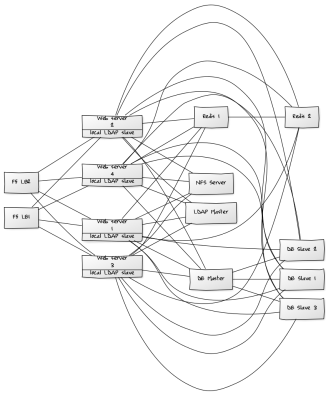
Plus de 4 serveurs applicatifs/Web.

Un cluster de deux (ou plus) serveurs de base de données.

Stockage sur un serveur NFS ou un magasin d'objets compatible S3.

Fédération de serveurs pour une configuration distribuée sur plusieurs centres de données.

Authentification à l'aide de serveurs LDAP ou Active Directory existants ou SAML.



- **Composants**
    - plus de 4 serveurs applicatifs avec 4 sockets et 64 Go de RAM
    - 4 serveurs de base de données avec 4 sockets et 64 Go de RAM
    - 2 équilibreurs de charge, comme par exemple BIG IP de F5
    - Stokage NFS
- **Système d'exploitation**

  Distribution Linux entreprise avec support du vendeur. Nous recommandons Ubuntu 16.04 LTS. D'autres distributions sont aussi supportées (par ex. : RedHat ou SuSE) mais elles peuvent ne pas fournir toutes les dépendances requises dans leurs dépôts officiels et il pourrait être nécessaire d'activer des dépôts tiers pour des modules comme APCu et Redis.
- **Configuration SSL**

  La terminaison SSL est réalisée dans l'équilibreur de charge. Un certificat standard SSL est nécessaire, installé selon la documentation de l'équilibreur de charge.
- **Équilibrage de charge**

  Des équipements redondants avec pulsation (heartbeat), par exemple F5 Big-IP. Deux équilibreurs de charge en frontal des serveurs applicatifs.
- **Base de données**

  Nous recommandons MySQL/MariaDB, avec un cluster Galera avec réplication maître-maître ou une configuration maître-esclave avec bascule automatique. (Moteur de stockage InnoDB, MyISAM n'est pas supporté, voir *MySQL / MariaDB storage engine*).
- **Authentification**

  Authentification à l'aide d'un ou plusieurs serveurs LDAP ou Active Directory, ou SAML/Shibboleth. Voir Authentification utilisateur avec LDAP et Intégration Shibboleth.)
- **LDAP**

  Des serveurs esclaves en lecture seule doivent être déployés pour chaque serveur applicatif pour une montée en charge optimale.
- **Gestion de session**

  Redis should be used for the session management storage.
- **Cache**
    - APC/APCu pour le cache local.
    - Redis pour le verrouillage de fichiers transactionnel et le cache distribué, sur un serveur dédié.

  Un système de mémoire cache améliore les performances du serveur. ownCloud sait gérer 4 systèmes de mémoire cache. Veuillez consulter Configuration de la mémoire cache pour des informations sur le choix et la configuration d'un système de mémoire cache.

- **Stockage**

  Une solution toute faite, comme IBM Elastic Storage ou RedHat Ceph doit être utilisée. Vous pouvez aussi utiliser un object store compatible S3.

Considérations sur le matériel

- Disques SSD obligatoires pour des performance d'entrées/sorties (I/O) optimales
- Partitions séparées pour le stockage et la base de données, disques SSD pour les base de données
- Plusieurs interfaces réseau pour distribuer le trafic sur plusieurs sous-réseaux

Machine unique / montée en charge

Le déploiement machine unique est largement utilisé dans la communauté.

Avantages :

- Configuration facile : pas de démon de stockage de session, utilisation de tmpfs et de mémoire cache pour améliorer les performances, stockage local.
- Pas d'inquiétude sur la latence réseau.
- Pour la montée en charge, augmenter les processeurs, la mémoire et l'espace disque.

Inconvénients :

- Pas d'option pour la haute disponiblité.
- La quantité de données tend à augmenter continuellement. Au bout du compte, une seule machine ne pourra pas absorber la montée en charge : les performances d'entrées/sorties diminuent et deviennent un goulet d'étranglement pour les téléchargements et les téléversements, même avec des disques SSD.

Déploiement pour une montée en charge

Configuration pour les fournisseurs de service :

- Round Robin DNS sur serveurs HAProxy (2-n, déchargement SSL, cache des ressources statiques)
- Distribution du trafic sur plusieurs serveurs Web pour diminuer la charge (2-n)
- Redis pour le stockage de session partagé (2-n)
- Cluster de base de données avec un seul maître et plusieurs esclaves, plus un proxy pour répartir les requêtes (2-n)
- GPFS ou Ceph via phprados (2-n, 3 pour être sûr, plus de 10 nœuds Ceph pour bénéficier de la rapidité sous la charge)

Avantages :

- Les composants peuvent être multipliés si besoin.
- Haute disponibilité.
- Tests de migration réalisés plus facilement.

Inconvénients :

- Configuration plus compliquée.
- Le réseau devient le goulet d'étranglements (10 Go Ethernet recommandé).
- Actuellement, la table de fichiers de cache de la base de données grossit rapidement, rendant les migrations pénibles si la table est altérée.

Et Nginx ?

Il peut être utilisé à la place de HAproxy pour l'équilibrage de charge.

Considérations sur les logiciels

Système d'exploitation

ownCloud est dépendant des distributions qui offrent un moyen facile d'installer les divers composants nécessaires à jour. d'expérience, nous recommandons vivevment Ubuntu 16.04 LTS, puisque toutes les dépendances requises sont contenues par défaut. Canonical, la maison mère de Ubuntu Linux, offre aussi des services et du support aux entreprises.

ownCloud a un partenariat avec RedHat et SUSE pour les clients ayant besoin d'un support commercial. CentOS est le clone gratuit de Red Hat Enterprise Linux, soutenu par la communauté. openSUSE est soutenu par la communauté et contient beaucoup des outils d'administration de SUSE Linux Enterprise Server.

Serveur Web

Si l'on compare Apache et Nginx, Apache avec mod_php est actuellement la mailleure option, car Nginx ne gère pas toutes les fonctionnalités nécessaires pour des déploiements en entreprise. mod_php est recommandé par rapport à PHP_FPM, car pour de la montée en charge, des pools PHP séparés ne sont tout simplement pas nécessaires.

Base de données relationnelles

Le plus souvent, le client a déjà une opinion sur la base de données à utiliser. Il est en général recommandé d'utiliser celle avec laquelle l'administrateur de bases de données est le plus familier. En prenant en compte ce que nous pouvons voir dans les déploiements chez nos clients, nous recommandons MySQL/MariaDB en mode maître esclave, avec un proxy MySQL en frontal pour envoyer les mises à jour au serveur maître et sélectionner les serveurs esclaves.

La deuxième meilleure option est d'utiliser PostgreSQL bien que nous ayons encore à trouver un client qui utilise une configuration maître-esclave.

Et les autres SGBD ?

- PostgreSQL est une bonne alternative à MySQL/MariaDB (les modifications des tables ne verrouillent pas les tables, ce qui rend les migrations moins pénibles), cependant, les configuration maître-esclave ne sont pas très courantes.
- Sqlite est adapté pour les tests ou pour un utilisateur unique. Il n'est pas approprié pour les systèmes en production.
- Microsoft SQL Server n'est pas une option supportée.
- Oracle est le standard de facto des grandes entreprises et est totalement supporté pour l'Édition Entreprise seulement.

..note:: Une seule base de données maître est un point de défaillance unique car pas de montée en charge possible

Quand la base maître est défaillante, un esclave peut devenir maître. Cependant, cette complexité accrue comporte des risques : un système multi-maître comporte le risque de cerveau partagé (split brain) et de verrous. ownCloud essai de résoudre le problème des verrous avec un système de verrouillage de fichiers de haut niveau.

Stockage de fichiers

Bien que beaucoup de clients commencent avec NFS, ils ont besoin tôt ou tard d'un stockage supportant la montée en charge. Actuellement, les options sont DRDB, GPFS ou GlusterFS, ou un protocole de stockage d'objet comme S3 (supporté pour l'Édition Entreprise seulement) ou Swift. S3 permet aussi l'accès à Ceph Storage.

Stockage de session

- Redis : fournit la persistance, des outils d'inspection graphique et gère le verrouillage de fichiers de haut niveau d'ownCloud.
- Si Shibboleth est un prérequis, vous devez utiliser Memcached, et il peut être aussi utilisé pour la montée en charge pour le stockage de session shibd (voir Memcache StorageService).

Références

Haute disponibilité de base de données

Amélioration des performances pour Apache et PHP

Configuration d'un serveur Redis comme gestionnaire de session pour PHP sur Ubuntu 14.04

Méthode d'installation préférée pour Linux

Pour les environnements de production, nous recommandons l'installation à partir de l'archive tar. Ceci s'applique en particulier aux scénarios où le serveur Web le stockage et la base de données sont situés sur des machines distinctes. Dans cette configuration, toutes les dépendances et les prérequis sont gérés par le gestionnaire de paquets de votre système d'exploitation, alors que le code ownCloud lui-même est maintenu dans une séquence d'étapes simples telles que documentées dans le *Manuel d'installation pour Linux* et le *Manuel de mise à jour d'ownCloud*.

Le paquet d'installation est utilisé pour les configurations avec un unique serveur.

Changements dans la version 9

Les paquets de distribution Linux (de Open Build Service) ont été divisés en plusieurs paquets pour ownCloud 9 : `owncloud`, `owncloud-deps` et `owncloud-files`.

Installer le méta-paquet `owncloud` pour obtenir une installation complète avec toutes les dépendances.

Les paquets `owncloud-files` installent seulement ownCloud, sans Apache, base de données ou dépendances PHP.

Les paquets `owncloud-deps` installent toutes les dépendances : Apache, PHP et MySQL. `owncloud-deps` n'est pas destiné à être installé seul mais en faisant partie du méta-paquet `owncloud`.

`owncloud-files` est disponible pour les distributions suivantes, mais pas `owncloud-deps`. Vous devrez installer votre propre LAMP d'abord. Ceci permet de créer votre propre LAMP personnalisé sans les conflits de dépendances du paquet ownCloud. Parcourez http://download.owncloud.org/download/repositories/9.1/owncloud/ pour trouver le paquet `owncloud-files` pour votre distribution :

- Ubuntu 14.04, 16.04
- Debian 7, 8
- RHEL 6, 7
- CentOS 6 SCL, 7
- SLES 12, 12 SP1
- openSUSE 13.2, Leap 42.1

Les paquets ownCloud avec les dépendances sont disponibles pour les distributions Linux suivantes :

- Ubuntu 14.04, 16.04
- Debian 8
- RHEL 7
- CentOS 7
- SLES 12
- openSUSE 13.2, Leap 42.1

Les dépôts pour Fedora, openSUSE Tumbleweed et Ubuntu 15.04 ont été supprimés. Si vous utilisez Fedora, utilisez l'archive tar sur votre LAMP. Les utilisateurs openSUSE peuvent se fier aux paquets LEAP pour Tumbleweed.

Suivez les instructions sur la page de téléchargement pour installer. Puis, lancez l'assistant d'installation (voir *Assistant d'installation*).

Warning

Ne déplacez pas les répertoires fournis par ces paquets car les mises à jour ne fonctionneront pas.

Consulter *Prérequis système* pour connaître les plateformes supportées et les configurations recommandées.

Dépôts : stable ou version majeure ?

Pour ownCloud 9.1, vous pouvez utiliser les dépôts suivants :

- https://download.owncloud.org/download/repositories/stable/owncloud/
- https://download.owncloud.org/download/repositories/9.1/owncloud/

Si vous utilisez le dépôt `Stable`, vous n'aurez plus besoin de le changer cer il pointe toujours sur la version stable en cours pour les versions majeures : 8.2, 9.0, etc. Les versions majeures sont indiquée par le deuxième chiffre, donc 8.0, 8.1, 8.2 et 9.0 sont des versions majeures.

Si vous souhaitez suivre une version majeure spécifique, comme la 9.0 ou la 9.1, utilisez alors ce répertoire. De cette manière, vous ne tomberez pas accidentellement sur une mise à jour pour la prochaine version majeure avant d'être prêt.

Installation d'ownCloud Édition Entreprise

Consulter *Installation et mise à jour de l'Édition Entreprise* pour des instuctions pour l'installation de l'Édition Entreprise.

Retour à une version précédente non supporté

Le retour à une version précédente n'est pas supporté et pourrait corrompre vos données ! Si vous voulez revenir à une version précédente, installer cette version puis restaurer vos données à partir d'une sauvegarde. Avant de faire cela, ouvrez un ticket support (si vous bénéficiez du support payant) ou demander de l'aide sur les forums d'ownClolud pour voir si votre problème peut être résolu sans revenir à la version précédente.

BINLOG_FORMAT = STATEMENT

Si votre installation d'ownCloud échoue et que vous voyez ceci dans votre fichier journal

```
An unhandled exception has been thrown: exception ‘PDOException’ with message
'SQLSTATE[HY000]: General error: 1665 Cannot execute statement: impossible to
write to binary log since BINLOG_FORMAT = STATEMENT and at least one table
uses a storage engine limited to row-based logging. InnoDB is limited to
row-logging when transaction isolation level is READ COMMITTED or READ
UNCOMMITTED.'
```

veuillez consulter *MySQL / MariaDB avec activation des journaux binaires*.

Guides et notes d'installation additionnels

Consulter *Assistant d'installation* pour les étapes importantes comme le choix de la base de données ou les permissions de répertoires.

Consulter *Configuration SELinux* pour des suggestions de configuration pour les distributions avec SELinux, comme Fedora et CentOS.

Si votre distribution n'est pas listée, elle maintient peut-être ses propres paquets ownCloud, ou vous pouvez préférez l'installer à partir du code source (voir *Manuel d'installation pour Linux*).

Archlinux : La version stable actuelle est dans le dépôt officiel de la communauté et d'autres paquets sont disponibles dans le dépôt utilisateur Arch.

Mageia : Le Wiki de Mageia contient une bonne page pour l'installation d'ownCloud à partir du dépôt logiciel de Mageia.

Exécution d'ownCloud dans un sous-répertoire : Si vous exécutez ownCloud dans un sous répertoire et que vous voulez utiliser des clients CalDAV ou CardDAV, assurez-vous d'avoir configuré correctement les URL *Service de découverte*.

Note pour les environnements MySQL/MariaDB : Veuillez consulter *MySQL / MariaDB avec activation des journaux binaires* sur la manière correcte de configurer votre environnement si les logs binaires sont activés.

Assistant d'installation

Démarrage rapide

Quand les prérequis d'ownCloud sont remplis et que les fichiers d'ownCloud sont installés, la dernière étape est terminer l'installation en lançant l'assistant d'nstallation. Cela se passe en trois étapes :

1. pointer le navigateur Web sur l'adresse `http://localhost/owncloud` ;
2. saisir le nom d'administrateur et le mot de passe désiré ;
3. cliquer sur **Terminer l'installation**.

Vous avez terminé et pouvez commencer à utiliser votre nouveau serveur ownCloud.

Bien sûr, il y a beaucoup plus à faire si vous voulez améliorer la sécurité et les performances. Dans les sections suivantes, nous couvrirons les étapes importantes de l'installation et de la post-installation. Veuillez noter que vous devez suivre les instructions de *Renforcer les permissions* pour utiliser la *Commande occ*.

- *Emplacement du répertoire data*
- *Choix de la base de données*
- *Domaines de confiance*
- *Renforcer les permissions*

Emplacement du répertoire data

Cliquez sur **Stockage & base de données** pour afficher des options de configuration supplémentaires pour le répertoire `data` d'ownCloud et la base de données.

Vous devriez définir votre répertoire `data` d'ownCloud en dehors de la racine Web si vous utilisez un serveur HTTP autre qu'Apache. Vous pourriez également souhaiter mettre le répertoire `data` dans un autre emplacement pour d'autres raisons (par exemple sur un serveur de stockage). Il est préférable de configurer l'emplacement de ce répertoire lors de l'installation car il est plus difficile de le déplacer ensuite. Vous pouvez le placer n'importe où. Ce répertoire doit déjà exister et être détenu par l'utilisateur HTTP (voir *Renforcement des permissions de répertoires*).

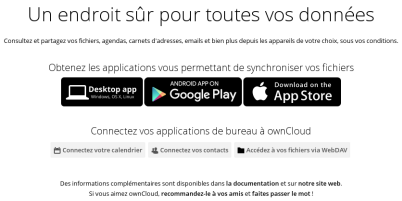
Choix de la base de données

SQLite est la base de données par défaut pour le serveur ownCloud (elle n'est pas disponible ni supportée dans l'Édition Entreprise d'ownCloud), et elle n'est adaptée que pour les tests et les environnements mono-utilisateur sans client de synchronisation. Les bases de données gérées sont MySQL, MariaDB, Oracle 11g (Édition Entreprise d'ownCloud seulement) et PostgreSQL. Nous recommandons l'utilisation de *MySQL/MariaDB*. Votre base de données et les connecteurs PHP doivent être installés avant de lancer l'assistant d'installation. Quand vous installez ownCloud à partir des paquets de votre distribution, toutes les dépendances sont satisfaites (voir *Manuel d'installation pour Linux* pour une liste détaillées des modules PHP obligatoires et facultatifs) . Vous aurez besoin du compte root de la base de données ou de tout compte ayant les droits suffisants pour créer et modifier une base de données, puis de saisit le nom que vous voulez pour la base de données ownCloud.

Après avoir saisi votre compte root ou adminiustrateur de la base de données, l'installeur crée un utilisateur de base de données spécial avec des privilèges limités à la base de données ownCloud. ownCloud n'a besoin que de l'utilisateur spécial pour ownCloud et oublie le mot de passe root de la base de données. Cet utilisateur est construit en utilisant le compte administrateur de la base de données que vous avez saisi précédemment préfixé par `oc_`. Un mot de passe aléatoire est généré. L'utilisateur de base de données ownCloud et sont mot de passe sont écrits dans le fichier `config.php`:

```
'dbuser' => 'oc_root',
'dbpassword' => 'pX65Ty5DrHQkYPE5HRsDvyFHlZZHcm',
```

Cliquez sur « Terminer l'installation » et commencez à utiliser votre nouveau serveur ownCloud.



Nous allons maintenant aborder quelques étapes pots-installation importantes.

Domaines de confiance

Toutes les URL pour accéder au serveur ownCloud doivent être mises dans une liste blanche dans votre fichier `config.php`, dans le paramètre `trusted_domains`. Les utilisateurs ne sont autorisés à se connecter que leorsque leur navigateur utilise une des adresses indiquées dans `trusted_domains`. Vous pouvez utiliser ds adresses IP ou des noms de domaine. Un configuration classique ressemble à ceci:

```
'trusted_domains' =>
  array (
   0 => 'localhost',
   1 => 'server1.exemple.com',
   2 => '192.168.1.50',
),
```

L'adresse de boucle (loopback), `127.0.0.1`, est automatiquement ajoutée à la liste blanche, de sorte que vous puissiez toujours vous connecter à partir du serveur. En cas d'utilisation d'un équilibreur de charge,, il n'y a pas de problème tant qu'il envoie l'en-tête correct `X-Forwarded-Host header`. Quand un utilisateur essaie une URL qui n'est pas dans la liste blanche, le message d'erreur suivant apparaît :

Renforcement des permissions de répertoires

Pour une sécurité renforcée, nous recommandons de définir des permissions aussi strictes que possible pour les répertoires d'ownCloud. Ceci doit être fait immédiatement après l'installation initiale at avant la configuration. Votre

utilisateur HTTP doit détenir les répertoires `config/`, `data/` et `apps/` pour que vous puissiez configurer ownCloud, créer, modifier et supprimer vos fichiers de données et installer des applications via l'interface Web d'ownCloud.

Vous pouvez trouver l'utilisateur HTTP dans le fichier de configuration de votre serveur HTTP. Ou vous pouvez utiliser *Informations et version de PHP* (recherchez la ligne **User/Group**).

- L'utilisateur et le groupe HTTP sous Debian/Ubuntu est `www-data`.
- L'utilisateur et le groupe HTTP sous Fedora/CentOS est `apache`.
- L'utilisateur et le groupe HTTP sous Arch Linux est `http`.
- Sous openSUSE, l'utilisateur HTTP est `wwwrun` et le groupe HTTP est `www`.

Note

Lors de l'utilisation de montages NFS pour le répertoire `data`, ne modifiez pas sons propriétaire par défaut. Le simple fait de monter le lecteur donnera les bonnes permissions pour ownCloud pour pouvoir écrire dans le répertoire. Changer le propriétaire comme indiqué plus haut, pourrait provoquer des problèmes si le montage NFS est perdu.

Le meilleur moyen de définir les permissions correctes et de copier et d'exécuter ce script. Remplacez la variable `ocpath` par le chemin d'accès à votre répertoire ownCloud et les variables `htuser` et `htgroup` par l'utilisateur et le groupe HTTP:

```
#!/bin/bash
ocpath='/var/www/owncloud'
htuser='www-data'
htgroup='www-data'
rootuser='root'

printf "Creating possible missing Directories\n"
mkdir -p $ocpath/data
mkdir -p $ocpath/assets
mkdir -p $ocpath/updater

printf "chmod Files and Directories\n"
find ${ocpath}/ -type f -print0 | xargs -0 chmod 0640
find ${ocpath}/ -type d -print0 | xargs -0 chmod 0750

printf "chown Directories\n"
chown -R ${rootuser}:${htgroup} ${ocpath}/
chown -R ${htuser}:${htgroup} ${ocpath}/apps/
chown -R ${htuser}:${htgroup} ${ocpath}/assets/
chown -R ${htuser}:${htgroup} ${ocpath}/config/
chown -R ${htuser}:${htgroup} ${ocpath}/data/
chown -R ${htuser}:${htgroup} ${ocpath}/themes/
chown -R ${htuser}:${htgroup} ${ocpath}/updater/

chmod +x ${ocpath}/occ

printf "chmod/chown .htaccess\n"
if [ -f ${ocpath}/.htaccess ]
 then
  chmod 0644 ${ocpath}/.htaccess
  chown ${rootuser}:${htgroup} ${ocpath}/.htaccess
fi
if [ -f ${ocpath}/data/.htaccess ]
 then
  chmod 0644 ${ocpath}/data/.htaccess
```

```
  chown ${rootuser}:${htgroup} ${ocpath}/data/.htaccess
fi
```

Si vous avez personnalisé votre installation d'ownCloud et que les chemins d'accès sont différents de l'installation standard, modifiez le script en conséquence.

Ceci détaille les modes et propriétaires recommandés pour vos fichiers et répertoires ownCloud :

- tous les fichiers doivent être accessibles en lecture et en écriture pour le propriétaire du fichier, en lecture seule pour le groupe et sans aucun droit pour le reste du monde ;
- Tous les répertoires doivent être exécutables (car les répertoires ont toujours besoin d'avoir le bit `executable` défini), accessibles en lecture et en écriture pour le propriétaire et en lecture seule pour le groupe ;
- le répertoire apps/ doit être détenu par `[HTTP user]:[HTTP group]` ;
- le répertoire config/ doit être détenu par `[HTTP user]:[HTTP group]` ;
- le répertoire themes/ doit être détenu par `[HTTP user]:[HTTP group]` ;
- le répertoire assets/ doit être détenu par `[HTTP user]:[HTTP group]` ;
- le répertoire data/ doit être détenu par `[HTTP user]:[HTTP group]` ;
- le fichier [ocpath]/.htaccess doit être détenu par `root:[HTTP group]` ;
- le fichiier data/.htaccess doit être détenu par `root:[HTTP group]` ;
- les fichiers .htaccess doivent être accessibles en lecture et en écriture pour le propriétaire et en lecture seule pour le groupe et le reste du monde.

Ce renforcement des permissions empêche la mise à jour du serveur ownCloud. Voir *Permissions pour la mise à jour* pour un script permettant de modifier rapidement les permissions pour permettre la mise à jour.

Installation d'ownCloud en ligne de commande

Il est maintenant possible d'installer ownCloud entièrement à partir de la ligne de commande. Ceci est pratique pour les opérations par script, les serveurs sans interface graphique et pour les administrateurs qui préfèrent la ligne de commande. Il y a trois étapes pour installer ownCloud en ligne de commande :

1. Téléchargez et installez ownCloud via votre gestionnaire de paquets, ou téléchargez et décompressez l'archive dans le répertoire approprié (voir *Méthode d'installation préférée pour Linux* et *Manuel d'installation pour Linux*).

2. Changez le propriétaire du répertoire `owncloud` par l'utilisateur HTTP, comme dans cet exemple pour Debian/Ubuntu. Vous devez exécuter la commande `occ` en tant qu'utilisateur HTTP (voir *Lancement d'occ en tant qu'utilisateur HTTP*):

```
$ sudo chown -R www-data:www-data /var/www/owncloud/
```

3. Utilisez la commande `occ` pour terminer l'installation. Ceci remplace l'installation avec l'assistant graphique:

```
$ cd /var/www/owncloud/
$ sudo -u www-data php occ  maintenance:install --database
"mysql" --database-name "owncloud"  --database-user "root" --database-pass
"password" --admin-user "admin" --admin-pass "password"
ownCloud is not installed - only a limited number of commands are available
ownCloud was successfully installed
```

Vous devez vous rendre dans la racine du répertoire ownCloud directory, comme dans l'exemple ci-dessus, pour exécuter `occ maintenance:install`, sans quoi l'installation échouera avec un message d'erreur PHP.

Les bases de données gérées sont:

```
- sqlite (SQLite3 - pas pour l'Édition Entreprise)
- mysql (MySQL/MariaDB)
- pgsql (PostgreSQL)
- oci (Oracle - Édition Entreprise seulement)
```

Voir *Installation en ligne de commande* pour plus d'informations.

Enfin, appliquez les permissions renforcées sur vos fichiers et répertoires ownCloud (voir *Renforcement des permissions de répertoires*). Ceci est une étape très importante. Cela permet de protéger l'installation ownCloud et assure son bon fonctionnement.

BINLOG_FORMAT = STATEMENT

Si votre installation d'ownCloud échou et que vous voyez ceci dans le fichier journal d'ownCloud:

```
An unhandled exception has been thrown: exception ‘PDOException’ with message
'SQLSTATE[HY000]: General error: 1665 Cannot execute statement: impossible to
write to binary log since BINLOG_FORMAT = STATEMENT and at least one table
uses a storage engine limited to row-based logging. InnoDB is limited to
row-logging when transaction isolation level is READ COMMITTED or READ
UNCOMMITTED.'
```

consultez *MySQL / MariaDB avec activation des journaux binaires*.

Changement de la racine Web

Ce manuel suppose que le serveur ownCloud sera accessible à partir de la racine Web `/owncloud` — c'est aussi là que les paquets Linux feront apparaître le serveur. Vous pouvez modifier ceci dans la configuration de votre serveur Web, par exemple de `https://exemple.com/owncloud/` en `https://exemple.com/`.

Des connaissances de base d'administration système et de la configuration d'Apache sont requises. Plusieurs fichiers doivent rester synchrones lors de la modification de la racine Web.

Sur un systèmes Ubuntu-14.04, les fichiers suivants sont d'ordinaire concernés :

- /etc/apache2/conf-enabled/owncloud.conf
- /var/www/owncloud/config/config.php
- /var/www/owncloud/.htaccess

Exemple : déplacement de /owncloud vers /

Modifier le fichier /etc/apache2/conf-enabled/owncloud.conf :

```
Alias / "/var/www/owncloud/"
```

Modifier le fichier /var/www/owncloud/config/config.php :

```
'overwrite.cli.url' => 'http://localhost/',
```

Modifier le fichier /var/www/owncloud/.htaccess :

```
...
####  NE RIEN MODIFIER AU-DESSUS DE CETTE LIGNE ####
...
<IfModule mod_rewrite.c>
  RewriteBase /
...
```

La racine document peut être aussi modifiée — ce n'est généralement ni nécessaire, ni recommandé. Modifier le fichier /etc/apache2/sites-enabled/000-default.conf :

```
DocumentRoot /var/www/owncloud
```

Note :

Depuis la version 9.0.2 d'owncloud, les URL courtes (sans index.php) sont supportées. Le mécanisme de réécriture utilise une règle RewriteBase dans le fichier .htaccess qui est générée automatiquement quand ownCloud est démarré pour la première fois. En fonction de la façon dont ownCloud a été installé (mise à jour ou nouvelle installation, archive .tar ou paquets), la règle RewriteBase pourrait ne pas être présente dans vos fichiers .htaccess. Si elle n'est pas encore présente, revérifier après le démarrage du serveur.

Installation et gestion des applications

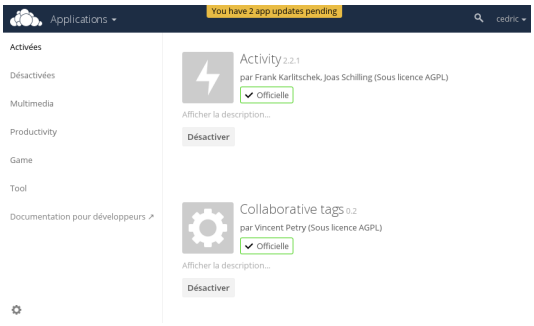
Après avoir installé ownCloud, vous pouvez ajouter des fonctionnalités en isntallant des applications.

Applications gérées

Consulter *Applications gérées dans ownCloud* pour une liste des applications gérées pour l'Édition Entreprise.

Affichage des applications activées

Pendant l'installation d'ownCloud, certaines applications sont activées par défaut. Pour voir lesquelles, rendez-vous sur la page Applications.



Vous verrez les applications activées, désactivées et celles recommandées. Vous verrez également des filtres supplémentaires comme Multimedia, Productivity et Tool pour trouver les applications plus rapidement.

Gestion des applications

Vous pouvez activer ou désactiver des applications sur la page Applications. Certaines ont des options configurables sur la page Applications, comme **Activer uniquement pour certains groupes**, mais principalement, elles sont juste activées ou désactivées ici et configurées dans la page Administration, Dans la page Personnel ou dans le fichier `config.php`.

Ajout d'applications tierces

Certaines application sont développées et gérées directement par ownCloud. Elles ont la marque **Officielle**. Les applications avec la marque **Approuvée** sont développées par la communauté et gérées. Elles sont maintenues par des développeurs de confiance et activement développées. Seules les applications **Officielle** et **Approuvée** sont liées sur cette page par défaut.

Cliquez sur le nom de l'application pour afficher une description de l'application. Cliquer sur le bouton **Activer** pour activer l'application. Si une application ne fait pas partie de l'installation d'ownCloud elle sera téléchargée depuis le magasin d'applications, installée et activée.

Cliquez sur l'icône représentant un engrenage pour voir les applications expérimentales dans le magasin d'application d'ownCloud. Installez les applications expérimentales à vos risques et périls.

Parfois, l'installation d'une application tierce échoue silencieusement probablement parce que `'appcodechecker' => true,` est défini dans le fichier `config.php`. Quand `appcodechecker` est activé, il vérifie si les applications tierces utilise des API privées, plutôt que les API publiques. Si c'est le cas, elles ne seront pas installées.

Note

Si vous voulez créer ou ajouter votre propre application ownCloud, veuillez consulter le manuel du développeur.

Utilisation de répertoires personnalisés pour les applications

Utilisez le tableau **apps_paths** dans dans le fichiers `config.php` pour définir des répertoires personnalisés pour les applications. La clé **path** le chemin absolu du sytsème de fichiers vers le dossier des applications. la clé **url** définit le chemin Web vers ce dossier, débutant à la racine Web d'ownCloud. La clé **writable** indique si l'utilisateur peut installer des applications dans ce dossier.

Note

Pour s'assurer que le dossier par défaut **/apps/** ne contient que des applications fournies par ownCloud, suivez cet exemple pour définir le dossier **/apps2/** qui sera utilisé pour stocker toutes les autres applications.

```
<?php

  "apps_paths" => array (
      0 => array (
              "path"     => OC::$SERVERROOT."/apps",
              "url"      => "/apps",
              "writable" => false,
      ),
      1 => array (
              "path"     => OC::$SERVERROOT."/apps2",
              "url"      => "/apps2",
              "writable" => true,
      ),
  ),
```

Utilisation de son propre magasin d'applications

Vous pouvez activer l'installation d'applications de votre propre magasin. Ceci néessite que vous puissiez écrire dans au moins un des répertoires d'applications configurés.

Pour activer l'installation à partir de votre propre magasin :

1. Définissez le paramètre **appstoreenabled** à `true`.

   Ce paramètre est utilisé pour activer votre propre magasin dans ownCloud.

2. indiquez dans **appstoreurl** l'URL de votre magasin.

   Ce paramètre est utilisé pour définir le chemin HTTP vers votre propre magasin ownCloud. Le serveur du magasin doit utiliser OCS.

```
<?php

  "appstoreenabled" => true,
  "appstoreurl" => "https://api.owncloud.com/v1",
```

Applications gérées dans ownCloud

Applications AGPL

- Activity
- AntiVirus
- Collaborative Tags
- Comments
- Encryption
- External Sites
- External Storage

- ownCloud WebDAV Endpoint (gère les anciens et nouveaux frontaux WebDAV)
- Federated File Sharing (permet le partage de fichiers sur plusieurs instances d'ownCloud)
- Federation (permet l'auto-complétion des noms d'utilisateurs sur plusieurs instances d'ownCloud)
- Files (ne peut être désactivée)
- Files PDF Viewer
- Files Sharing
- Files TextEditor
- Files Trashbin
- Files Versions
- Files VideoPlayer
- First Run Wizard
- Gallery
- Notifications
- Object Storage (Swift)
- Provisioning API
- Template Editor (pour les courriels de notification)
- Update Notifications
- User External
- User LDAP

Applications Édition Entreprise seulement

- Enterprise License Key
- Files Drop
- File Firewall
- LDAP Home Connector
- Log user and Sharing actions (nouvelle application remplaçant les deux précédentes)
- Object Storage (S3)
- SharePoint
- Shibboleth (SAML)
- Windows Network Drives (nécessite External Storage)
- Workflow

Manuel d'installation pour Linux

L'installation d'ownCloud sous Linux à partir des paquets Open Build Service est la méthode préférée (voir *Méthode d'installation préférée pour Linux*). Ces paquets sont maintenus par les ingénieurs ownCloud et vous pouvez utiliser votre gestionnaire de paquets pour garder votre serveur ownCloud à jour.

Note

Les clients de l'Édition Entreprise doivent se référer à *Installation et mise à jour de l'Édition Entreprise*

S'il n'existe pas de paquet pour votre distribution Linux, ou si vous préférez l'installer à partir des sources, vous pouvez configurer ownCloud à partir de zéro en utilisant un ensemble LAMP classique (Linux, Apache,

MySQL/MariaDB, PHP). Ce document décrit les étapes complètes pour l'installation d'ownCloud sur Ubuntu 14.04 LTS Server avec Apache et MariaDB, en utilisant l'archive .tar d'ownCloud.

Note

Les administrateurs de distributions ayant SELinux comme CentOS, Fedora et Red Hat Enterprise Linux peuvent avoir besoin de définir de nouvelles règles pour installer ownCloud. Consulter *Conseils de configuration SELinux* pour une suggestion de configuration.

Prérequis

L'archive .tar d'ownCloud contient tous les modules PHP requis. Cette section liste tous les modules obligatoires et facultatifs. Consulter le manuel PHP pour plus d'informations sur ces modules. Votre distribution Linux devrait contenir tous les modules obligatoires. Vous pouvez vérifier la présence d'un module en saisissant la commande `php -m | grep -i <nom_module>`. Si vous obtenez un résultat, le module est présent.

Obligatoires :

- php5 (>= 5.4) ;
- module PHP ctype ;
- module PHP dom ;
- module PHP GD ;
- module PHP iconv ;
- module PHP JSON ;
- module PHP libxml (>=2.7.0) ;
- module PHP mb multibyte ;
- module PHP posix ;
- module PHP SimpleXML ;
- module PHP XMLWriter ;
- module PHP zip ;
- module PHP zlib.

Connecteurs de base de données (choisissez celui de votre base de données) :

- module PHP sqlite (>= 3, non recommandé pour des raisons de performances) ;
- module PHP pdo_mysql (MySQL/MariaDB) ;
- module PHP pgsql (nécessite PostgreSQL >= 9.0).

Paquets *recommandés* :

- module PHP curl (vivement recommandé, certaines fonctionnalités, comme l'authentification utilisateur HTTP, dépendent de celui-ci) ;
- module PHP fileinfo (vivement recommandé, active les performances d'analyse de fichiers) ;
- module PHP bz2 (recommandé, nécessaire pour l'extraction des applications) ;
- module PHP intl (améliore les performances de traduction des langues et corrige le tri des caractères non ASCII) ;
- module PHP mcrypt (améliore les performances pour le chiffrement des fichiers) ;
- module PHP openssl (nécessaire pour accéder aux ressources HTTPS).

Obligatoires pour des appllications spécifiques :

- module PHP ldap (pour l'intégration LDAP) ;
- module PHP ftp (pour le stockage FTP / l'authentification utilisateur externe) ;
- module PHP imap (l'authentification utilisateur externe).
- module PHP smbclient (pour l'intégration SMB/CIFS)

Note

Les montages de lecteurs réseau SMB/Windows nécessitent le module PHP smbclient. Consulter *SMB/CIFS*).

Recommandés pour des applications spécifiques (*facultatif*) :

- module PHP exif (pour la rotation d'image dans les applications images) ;
- module PHP gmp (pourle stockage SFTP).

Pour améliorer les performances du serveur (*facultatif*) sélectionner un des systèmes de gestion de mémoire cache suivants :

- module PHP apc ;
- module PHP apcu ;
- module PHP memcached ;
- module PHP redis (>= 2.2.6+, nécessaire pour le verrouillage de fichier transactionnel).

Consulter *Configuration de la mémoire cache* pour savoir comment sélectionner et configurer un système de gestion de mémoire cache.

Génération des aperçus (*facultatif*) :

- module PHP imagick ;
- avconv ou ffmpeg ;
- OpenOffice ou LibreOffice.

Traitement en ligne de commande (*facultatif*) :

- module PHP pcntl (active l'interruption de commande en utilisant `ctrl-c`).

Vous n'avez pas besoin du module WebDAV pour votre serveur Web (c'est-à-dire le module `mod_webdav` d'Apache), car ownCloud contient un serveur WebDAV intégré, SabreDAV. Si `mod_webdav` est activé, vous devez le désactiver pour ownCloud (consulter *Configuration du serveur Web Apache* pour un exemple de configuration).

Exemple d'installation pour Ubuntu 14.04 LTS Server

Note

Voir Détails d'installation manuelle pour plusieurs distributions, ownCloud 9.0 et 9.1 pour des astuces d'installation pour Ubuntu 16.04 LTS, RHEL 7.2 et SLES 12.

Sur une machine exécutant un serveur Ubuntu 14.04 LTS fraîchement installé, installer les modules obligatoires et recommandés pour une installation classique d'ownCloud, utilisant Apache et MariaDB, à l'aide des commandes suivantes

```
apt-get install apache2 mariadb-server libapache2-mod-php5
apt-get install php5-gd php5-json php5-mysql php5-curl
apt-get install php5-intl php5-mcrypt php5-imagick
```

- Ceci installe les paquets pour le cœur du système d'ownCloud. `libapache2-mod-php5` fournit les extensions PHP suivantes : `bcmath bz2 calendar Core ctype date dba dom ereg exif fileinfo filter ftp gettext hash iconv libxml mbstring mhash openssl pcre Phar posix Reflection session shmop SimpleXML soap sockets SPL standard sysvmsg sysvsem sysvshm tokenizer wddx xml xmlreader xmlwriter zip zlib`. Si vous projetez d'utiliser des applications supplémetaires, gardez à l'esprit qu'elles peuvent nécessiter des paquets supplémentaires. Consulter *Prérequis* pour des détails.
- Lors de l'installation du serveur MySQL/MariaDB, vous serez invité à créer un mot de passe root. Retenez-le car il sera nécessaire lors de la configuration de la base de données d'ownCloud.

Téléchargez maintenant la dernière version de l'archive d'ownCloud :

- rendez-vous sur la page de téléchargements d'ownCloud ;
- puis dans **Download ownCloud Server > Download > Archive file for server owners** et téléchargez l'archive tar.bz2 ou .zip ;
- ceci télécharge un fichier nommé owncloud-x.y.z.tar.bz2 ou owncloud-x.y.z.zip (où x.y.z est le numéro de version) ;
- téléchargez son fichier de somme de contrôle correspondant : owncloud-x.y.z.tar.bz2.md5 ou owncloud-x.y.z.tar.bz2.sha256 ;
- vérifiez la somme MD5 ou SHA256:

```
md5sum -c owncloud-x.y.z.tar.bz2.md5 < owncloud-x.y.z.tar.bz2
sha256sum -c owncloud-x.y.z.tar.bz2.sha256 < owncloud-x.y.z.tar.bz2
md5sum  -c owncloud-x.y.z.zip.md5 < owncloud-x.y.z.zip
sha256sum  -c owncloud-x.y.z.zip.sha256 < owncloud-x.y.z.zip
```

- vous pouvez aussi vérifier la signature PGP:

```
wget https://download.owncloud.org/community/owncloud-x.y.z.tar.bz2.asc
wget https://owncloud.org/owncloud.asc
gpg --import owncloud.asc
gpg --verify owncloud-x.y.z.tar.bz2.asc owncloud-x.y.z.tar.bz2
```

- vous pouvez maintenant extraire le contenu de l'archive en utilisant la commande appropriée en fonction du type de l'archive:

```
tar -xjf owncloud-x.y.z.tar.bz2
unzip owncloud-x.y.z.zip
```

- le contenu sera décompressé dans un unique répertoire `owncloud`. Copiez le répertoire ownCloud vers sa destination finale. si vous utilisez le serveur HTTP Apache, vous pouvez installer sans risque ownCloud dans le répertoire document racine d'Apache:

```
cp -r owncloud /chemin/vers/serveurweb/racine-document
```

où `/chemin/vers/serveurweb/racine-document` doit être remplacé par la racine document de votre serveur Web:

```
cp -r owncloud /var/www
```

Sur les autres serveurs HTTP, il est recommandé d'installer ownCloud en dehors de la racine document.

BINLOG_FORMAT = STATEMENT

Si votre installation d'ownCloud échoue et que vous voyez ceci dans le fichier journal d'ownCloud:

```
An unhandled exception has been thrown: exception ‘PDOException’ with message
'SQLSTATE[HY000]: General error: 1665 Cannot execute statement: impossible to
write to binary log since BINLOG_FORMAT = STATEMENT and at least one table
uses a storage engine limited to row-based logging. InnoDB is limited to
row-logging when transaction isolation level is READ COMMITTED or READ
UNCOMMITTED.'
```

veuillez consulter *MySQL / MariaDB avec activation des journaux binaires*.

Configuration du serveur Web Apache

Sur Debian, Ubuntu et leurs dérivés, Apache est installé avec une configuration opérationnelle, tout ce que vous avez à faire est de créer le fichier /etc/apache2/sites-available/owncloud.conf contenant ces lignes, en remplaçant l'argument de **Directory** et les autres chemins de fichiers par ceux correspondant à votre configuration:

```
Alias /owncloud "/var/www/owncloud/"

<Directory /var/www/owncloud/>
  Options +FollowSymlinks
  AllowOverride All

 <IfModule mod_dav.c>
  Dav off
 </IfModule>

 SetEnv HOME /var/www/owncloud
 SetEnv HTTP_HOME /var/www/owncloud

</Directory>
```

Créez ensuit un lien symbolique dans /etc/apache2/sites-enabled:

```
ln -s /etc/apache2/sites-available/owncloud.conf /etc/apache2/sites-enabled/owncloud.conf
```

Configurations additionnelles d'Apache

- Pour qu'ownCloud fonctionne correctement, le module `mod_rewrite` est nécessaire. Activez-le avec la commande suivante:

  ```
  a2enmod rewrite
  ```

  Les modules complémentaires suivants sont recommandés : `mod_headers`, `mod_env`, `mod_dir` et `mod_mime`:

  ```
  a2enmod headers
  a2enmod env
  a2enmod dir
  a2enmod mime
  ```

  Si vous utilisez `mod_fcgi` plutôt que `mod_php`, activez aussi:

  ```
  a2enmod setenvif
  ```

- Vous devez sésactiver toute authentification serveur pour ownCloud, car il utilise une authentification basique en interne pour les services DAV. Si vous activé l'authentification sur un dossier parent (avec la directive

`AuthType Basic` par exemple), vous pouvez désactiver l'authentification spécifiquement pour l'entrée ownCloud. En suivant l'exemple de fichier de configuration ci-dessus, ajoutez la ligne suivante dans la section `<Directory`:

```
Satisfy Any
```

- Lors de l'utilisation de SSL, faites attention à la directive `ServerName`. Voius devez l'indiquer dans la configuration du serveur, à l'identique du champ CommonName du certificat.
- Redémarrez maintenant Apache:

```
service apache2 restart
```

- Si vous exécutez ownCloud dans un sous-répertoire et que vous voulez utiliser des clients CalDAV ou CardDAV, assurez-vous d'avoir configuré correctement les URL *Service de découverte*.

Module Multi-Processing (MPM)

Apache prefork doit être utilisé. N'utilisez pas un module `MPM` w threadé » comme `event` ou `worker` avec `mod_php`, car PHP n'est pas encore thread safe.

Activation de SSL

> ***Note***
>
> Vous pouvez utiliser ownCloud avec HTTP, mais nous recommandons vivement d'utiliser SSL/TLS pour chiffrer tout le trafic de votre serveur, et pour protéger les identifiants de connexion de vos utilisateurs et les données en transit.

Apache sous Ubuntu est installé avec un certificat auto-signé. Vous avez juste à activer le module SSL et le site SSL par défaut. Dans un terminal, saisissez les commandes suivantes:

```
a2enmod ssl
a2ensite default-ssl
service apache2 reload
```

> ***Note***
>
> Les certificats auto-signés ont leurs revers - particulièrement si vous comptez rendre votre serveur accessible publiquement. Vous devriez envisager d'acquérir un certificat signé par une autorité de certificationYou. Vérifiez auprès de votre fournisseur d'accès à Internet ou auprès de votre registrar.

Assistant d'installation

Après avoir rdémarré Apache, vos devez terminer votre installation en lançant l'assistant d'installation graphique, ou en ligne de commande en utilisant la commande `occ` . Pour cela, changez temporairement le propriétaire de vos répertoires ownCloud par votre utilisateur HTTP (consulter *Renforcement des permissions de répertoires* pour savoir comment trouver votre utilisateur HTTP):

```
chown -R www-data:www-data /var/www/owncloud/
```

> ***Note***
>
> Les administrateurs des distributions ayant SELinux peuvent avoir besoin d'écrire de nouvells règles pour terminer leur installation d'ownCloud. Voir *Conseils de configuration SELinux*.

Pour utiliser la commande `occ`, consulter *Installation d'ownCloud en ligne de commande*.

Pour utiliser l'assistant d'installation graphique, consulter *Assistant d'installation*.

Renforcement des permissions des répertoires

Après avoir terminé l'installation, vous devez immédiatement définir les permissions de répertoire de votre installation d'ownCloud aussi strictement que possible pour une sécurité renforcéd. Veuillez consulter à cette fin *Renforcement des permissions de répertoires*.

Votre serveur ownCloud est maintenant prêt à être utilisé.

Conseils de configuration SELinux

Consulter *Configuration SELinux* pour des suggestions de configuration pour les distributions ayant SELinux, comme Fedora et CentOS.

Notes de configuration pour php.ini

Gardez à l'esprit que les modifications du fichier `php.ini` peuvent devoir être faites à plusieurs endroits. Ce peut être le cas pour le paramètre `date.timezone` par exemple.

php.ini - utilisé par le serveur Web :

```
  /etc/php5/apache2/php.ini
ou
  /etc/php5/fpm/php.ini
ou …
```

php.ini - utilisé par le client PHP ou par les tâches CRON d'ownCloud :

```
/etc/php5/cli/php.ini
```

Notes de configuration pour php-fpm

Sécurité : Utilisez au moins PHP => 5.5.22 ou >= 5.6.6

En raison d'un bogue ayant des implications de sécurité dans les anciennes versions de PHP dans la gestion des données XML, vous être vivement encouragé à utiliser au moins PHP 5.5.22 ou 5.6.6 dans un environnement threadé.

Variables d'environnement système

Lors de l'utilisation de `php-fpm`, les variables d'environnement système comme PATH, TMP ou d'autres ne sont pas automatiquement renseignées de la même manière qu'en utilisant `php-cli`. Un appel PHP comme `getenv('PATH');` peut par conséquent renvoyer une valeur vide. Vous devez alors configurer manuellement les variables d'environnement dans le fichier ini/config approprié de `php-fpm`.

Voici quelques exemples d'amplacement de ces fichiers :

Ubuntu/Mint	CentOS/Red Hat/Fedora
`/etc/php5/fpm/`	`/etc/php-fpm.d/`

Dans les deux exemples, le fichier ini/config est appelé `www.conf`, et dépend de la version de votre distribution et des personnalisations que vous avez faites. Le fichier peut se trouver dans un sous-répertoire.

Généralement, vous trouverez tout ou partie des variables d'environnement dans ce fichier, commentées comme ceci:

```
;env[HOSTNAME] = $HOSTNAME
;env[PATH] = /usr/local/bin:/usr/bin:/bin
;env[TMP] = /tmp
;env[TMPDIR] = /tmp
;env[TEMP] = /tmp
```

Supprimer les commentaires pour les entrées appropriées. Exécuter ensuite la commande `printenv PATH` pour confirmer vos chemins, par exemple:

```
$ printenv PATH
/home/user/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:
/sbin:/bin:/
```

Si une des variables d'environnements système n'est pas présente dans le fichier, vous devrez alors l'ajouter.

Si vous utilisez un hébergement partagé ou un panneau de contrôle pour gérer votre VM ownCloud ou votre serveur, les fichiers de configuration se trouveront très probablement ailleurs, pour des raisons de sécurité et de flexibilité. Consultez alors votre documentation pour trouver les emplacements corrects.

Gardez à l'esprit qu'il est possible de créer des paramètres différents pour `php-cli` et `php-fpm`, et pour des domaines et des sites Web différents. Le meilleur moyen de vérifier ces paramètres est d'utiliser *Informations et version de PHP*.

Taille maximale de téléversement

Si vous voulez augmenter la taille des fichiers téléversés, vous devez aussi modifier votre configuration `php-fpm` et augmenter les valeurs `upload_max_filesize` et `post_max_size`. Vous devrez redémarrer `php5-fpm` et votre serveur HTTP pour que les changements soient pris en compte.

Notes .htaccess pour Apache

ownCloud fournit son propre fichier `owncloud/.htaccess`. Parce que `php-fpm` ne sait pas lire les paramètres PHP dans `.htaccess`, ces paramètres et permissions doivent être définis dans le fichier `owncloud/.user.ini`.

Autres serveurs Web

Exemples de configurations pour Nginx

Autres serveurs Web

Installation de Univention Corporate Server

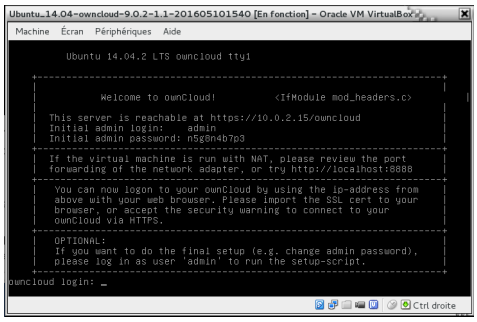
Appliance de la communauté ownCloud

ownCloud a développé une appliance publique sur GitHub. Téléchargez la dernière version sur l'onglet Appliances sur la page d'installation du serveur ownCloud. Le moyen le plus facile est d'utiliser VirtualBox et de télécharger l'image OVA sur la page d'installation.

Instructions pour VirtualBox et OVA

Suivez les étapes suivantes pour faire fonctionner l'appliance :

1. téléchargez le fichier zip de l'image de machine virtuelle et décompressez-la ;
2. démarrez VirtualBox et sélectionnez le menu *Fichier… > Importer une application virtuelle…* et importez votre nouvelle image ownCloud ;
3. cliquez sur la flèche verte. Après une minute vous devriez voir le message de bienvenue de la console ;



4. notez le nom d'utilisateur et le mot de passe ici. C'est un mot de passe aléatoire que nous générons pour vous au premier lancement. Si vous vous connectez à partir de la console, vous serez invité à changer de mot de passe. Ceci est facultatif ;

5. avec votre navigateur Web essayez `http://localhost:8888` ou `http://localhost:80` ou l'adresse indiquée sur la console. L'une d'entre elles devrait fonctionner. Si ce n'est pas le cas, veuillez revoir la configuration réseau de Virtualbox en la passant en `Accès par pont` ;
6. Vous devriez voir une page Web avec les identifiants de connexion (si vous ne les avez pas déjà changés) et une liste d'URL à essayer pour joindre le service Web d'ownCloud. Celle qui fonctionne dépendra de la configuration réseau de votre hyperviseur.

Cliquer pour agrandir

Note

Vous devriez noter votre mot de passe admin et vous assurer que les identifiants ne soient plus affichés. Cliquez sur *[Hide Credentials]*. En utilisant l'application ownCloud Proxy, cette page peut être visible publiquement.

Note

Dans la VM, ownCloud s'exécute avec la taille de disque par défaut de 40 Go et sa propre base de données MySQL. L'utlisateur admin d'ownCloud admin est auusi un compte valide sur le système Ubuntu utilisé par la VM. Vous pouvre administrer la VM via SSH.

Pour VMware

Vous pouvez suivre les mêmes étapes que ci-dessus, cependant, après avoir ouvert le fichier VMX , vous devrez configurer `Accès par pont` pour l'*adaptateur réseau*

Appliances logicielles

Il existe plusieurs machines virtuelles pré-configurées non officielles :

- Tech and Me - VM ownCloud sous Ubuntu 16.04 avec PHP 7, MySQL et Apache ;
- SUSE Studio, ownCloud sous openSuSE, qui se lance directement à partir d'une clé USB ;
- Amahi home server

Installation de PHP 5.4 sous RHEL 6 et CentOS 6

Red Hat Enterprise Linux et CentOS 6 fournissent toujours PHP 5.3. ownCloud nécessite au moins PHP 5.4. Il existe plusieurs dépôt tiers qui fournissent PHP 5.4, mais vous devez utiliser le dépôt Software Collections (SCL) pour être en conformité avec votre contrat de support RHEL.

RHEL 6

Suivez ces étapes pour installer PHP 5.4 à partie de SCL. Vous devez d'abord utiliser votre gestionnaire d'abonnement pour activer SCL:

```
subscription-manager repos --enable rhel-server-rhscl-6-eus-rpms
```

Pour installer PHP 5.4 et ces modules:

```
yum install php54 php54-php php54-php-gd php54-php-mbstring
```

Vous devez aussi installer la mise à jour du module pour votre base de données. Cet exemple installe le nouveau module PHP 5.4 pour MySQL/MariaDB:

```
yum install php54-php-mysqlnd
```

Désactivez le chargement de l'ancien module Apache PHP 5.3:

```
mv /etc/httpd/conf.d/php.conf /etc/httpd/conf.d/php53.off
```

Vous devriez maintenant avoir un fichier /etc/httpd/conf.d/php54-php.conf qui charge l module Apache PHP 5.4.

Ensuite, redémarrer Apache:

```
service httpd restart
```

Vérifiez avec *Informations et version de PHP* que votre serveur Apache utilise PHP 5.4 et charge les modules corrects.

CentOS 6

Installez d'abord le dépôt SCL:

```
yum install centos-release-SCL
```

Puis, installez PHP 5.4 et ces modules:

```
yum install php54 php54-php php54-php-gd php54-php-mbstring
```

Vous devez aussi installer la mise à jour du module pour votre base de données. Ceci installe le nouveau module PHP 5.4 pour MySQL/MariaDB:

```
yum install php54-php-mysqlnd
```

Désactivez le chargement de l'ancien module Apache PHP 5.3:

```
mv /etc/httpd/conf.d/php.conf /etc/httpd/conf.d/php53.off
```

Vous devriez maintenant avoir un fichier /etc/httpd/conf.d/php54-php.conf qui charge l module Apache PHP 5.4.

Ensuite, redémarrer Apache:

```
service httpd restart
```

Vérifiez avec *Informations et version de PHP* que votre serveur Apache utilise PHP 5.4 et charge les modules corrects.

Installation de PHP 5.5 sous RHEL 7 et CentOS 7

PHP 5.4 est en fin de vie depuis September 2015 et n'est plus supporté par l'équipe PHP. RHEL 7 fournit toujours PHP 5.4 et Red Hat le supporte. ownCloud sait aussi gérer PHP 5.4, la mise à jour n'est donc pas nécessaire. Cependant, il est vivement recommandé de procéder à la mise à jour en version PHP 5.5 ou supérieure pour des performances et une sécurité accrues.

Avant de mettre à jour, assurez-vous de la compatibilité de vos applications PHP avec PHP 5.5.

Mise à jour PHP 5.5 pour RHEL 7

Pour mettre à jour en version PHP 5.5, vous devez utiliser le dépôt Software Collections (SCL) pour être en conformité avec votre contrat de support RHEL. Suivez ces étapes pour installer PHP 5.4 à partie de SCL. Vous devez d'abord utiliser votre gestionnaire d'abonnement pour activer SCL:

```
subscription-manager repos --enable rhel-server-rhscl-7-eus-rpms
```

Pour installer PHP 5.5 et ces modules:

```
yum install php55 php55-php php55-php-gd php55-php-mbstring
```

Vous devez aussi installer la mise à jour du module pour votre base de données. Cet exemple installe le nouveau module PHP 5.5 pour MySQL/MariaDB:: :

```
yum install php55-php-mysqlnd
```

Si vous utilisez l'application ownCloud LDAP, vous avez besoin de ce module:

```
yum install php55-php-ldap
```

Désactivez le chargement des anciens modules Apache PHP en changeant leurs noms:

```
mv /etc/httpd/conf.d/php.conf /etc/httpd/conf.d/php54.off
mv /etc/httpd/conf.modules.d/10-php.conf /etc/httpd/conf.modules.d/10-php54.off
```

Copier les modules PHP 5.5 Apache à la place:

```
cp /opt/rh/httpd24/root/etc/httpd/conf.d/php55-php.conf /etc/httpd/conf.d/
cp /opt/rh/httpd24/root/etc/httpd/conf.modules.d/10-php55-php.conf /etc/httpd/conf.modules.d
cp /opt/rh/httpd24/root/etc/httpd/modules/libphp55-php5.so /etc/httpd/modules/
```

Ensuite, redémarrer Apache:

```
service httpd restart
```

Vérifiez avec *Informations et version de PHP* que votre serveur Apache utilise PHP 5.5 et charge les modules corrects.

Mise à jour PHP 5.5 pour CentOS 7

Pour mettre à jour en PHP 5.5, utilisez le dépôt Red Hat Software Collections (SCL).

Avant de mettre à jour, assurez-vous de la compatibilité de vos applications PHP avec PHP 5.5.

Installez d'abord le dépôt SCL:

```
yum install centos-release-scl
```

Puis, installez PHP 5.5 et ces modules:

```
yum install php55 php55-php php55-php-gd php55-php-mbstring
```

Vous devez aussi installer la mise à jour du module pour votre base de données. Ceci installe le nouveau module PHP 5.5 pour MySQL/MariaDB:

```
yum install php55-php-mysqlnd
```

Si vous utilisez l'application ownCloud LDAP, vous avez besoin de ce module:

```
yum install php55-php-ldap
```

Désactivez le chargement des anciens modules Apache PHP en changeant leurs noms:

```
mv /etc/httpd/conf.d/php.conf /etc/httpd/conf.d/php54.off
mv /etc/httpd/conf.modules.d/10-php.conf /etc/httpd/conf.modules.d/10-php54.off
```

Copier les modules PHP 5.5 Apache à la place:

```
cp /opt/rh/httpd24/root/etc/httpd/conf.d/php55-php.conf /etc/httpd/conf.d/
cp /opt/rh/httpd24/root/etc/httpd/conf.modules.d/10-php55-php.conf /etc/httpd/conf.modules.d
cp /opt/rh/httpd24/root/etc/httpd/modules/libphp55-php5.so /etc/httpd/modules/
```

Ensuite, redémarrer Apache:

```
service httpd restart
```

Vérifiez avec *Informations et version de PHP* que votre serveur Apache utilise PHP 5.5 et charge les modules corrects.

Configuration SELinux

Quand SELinux est activé sur votre distribution Linux, vous pouvez rencontrer des problèmes de permissions après une nouvelle installation d'ownCloud et voir des erreurs `permission denied` dans vos journaux ownCloud.

Les paramètres suivants devraient fonctionner pour la plupart des systèmes SELinux utilisant les profils par défaut de la distribution. Exécutez ces commandes en tant que root et ajustez les chemins d'accès des fichiers en fonction de votre installation :

```
semanage fcontext -a -t httpd_sys_rw_content_t '/var/www/html/owncloud/data(/.*)?'
semanage fcontext -a -t httpd_sys_rw_content_t '/var/www/html/owncloud/config(/.*)?'
semanage fcontext -a -t httpd_sys_rw_content_t '/var/www/html/owncloud/apps(/.*)?'
semanage fcontext -a -t httpd_sys_rw_content_t '/var/www/html/owncloud/assets(/.*)?'
semanage fcontext -a -t httpd_sys_rw_content_t '/var/www/html/owncloud/.htaccess'
semanage fcontext -a -t httpd_sys_rw_content_t '/var/www/html/owncloud/.user.ini'

restorecon -Rv '/var/www/html/owncloud/'
```

Si vous désinstallez ownCloud, vous devez supprimer les libellés de répertoires d'ownCloud. Pour cela, exécutez les commandes suivantes en tant que root après avoir désinstallé ownCloud :

```
semanage fcontext -d '/var/www/html/owncloud/data(/.*)?'
semanage fcontext -d '/var/www/html/owncloud/config(/.*)?'
semanage fcontext -d '/var/www/html/owncloud/apps(/.*)?'
semanage fcontext -d '/var/www/html/owncloud/assets(/.*)?'
semanage fcontext -d '/var/www/html/owncloud/.htaccess'
semanage fcontext -d '/var/www/html/owncloud/.user.ini'

restorecon -Rv '/var/www/html/owncloud/'
```

Note : Le dossier « assets » n'est nécessaire que si « JavaScript and CSS Asset Management » est activé. (`'asset-pipeline.enabled' => true`, dans le fichier `config.php`).

Si vous avez des politiques SELinux personnalisées et que ces exemples ne fonctionnent pas, vous devez donner les droits en écriture au serveur HTTP sur ces répertoires :

```
/var/www/html/owncloud/data
/var/www/html/owncloud/config
/var/www/html/owncloud/apps
/var/www/html/owncloud/assets
```

Activation des mises à jour via l'interface Web

Pour activer les mises à jour via l'interface Web d'ownCloud, vous aurez peut-être besoin d'autoriser l'écriture dans les répertoires d'ownCloud

```
setsebool httpd_unified on
```

Quand la mise à jour est terminée, enlever les droits en écriture

```
setsebool -P  httpd_unified  off
```

Empêcher l'accès en écriture sur tout le répertoire Web

Pour des raisons de sécurité, il est suggéré de désactiver l'accès en écriture sur tous les répertoires dans /var/www/ (par défaut)

```
setsebool -P  httpd_unified  off
```

Autorisation d'accès à une base de données distante

Un paramètre supplémentaire est nécessaire si votre installation se connecte sur une base de données distante :

```
setsebool -P httpd_can_network_connect_db on
```

Autorisation d'accès à un serveur LDAP

Utiliser ce paramètre pour autoriser les connexions LDAP :

```
setsebool -P httpd_can_connect_ldap on
```

Autorisation d'accès à un réseau distant

ownCloud nécssite d'accéder à des réseaux distants pour des fonctions comme le partage de serveur à serveur, les stockages externes our le magasin d'applications. Pour permettre ceci, utiliser le paamètre suivant :

```
setsebool -P httpd_can_network_connect on
```

Autorisation d'accès au réseau memcache

Ce paramètre n'est pas nécessaire si `httpd_can_network_connect` est déjà défini à « on » :

setsebool -P httpd_can_network_memcache on

Autorisation d'accès à SMTP/sendmail

Si vous voulez autoriser ownCloud à envoyer des courriels de notification par sendmail, utilisez le paramètre suivant :

```
setsebool -P httpd_can_sendmail on
```

Autorisation d'accès à CIFS/SMB

Si le répertoire de données est situé sur un partage CIFS/SMB, utiliser le paramètre suivant :

```
setsebool -P httpd_use_cifs on
```

Autorisation d'accès à FuseFS

Si votre dossier data d'ownCloud réside sur une système de fichiers Fuse (par ex. : EncFS etc), ce paramètre est également requis

```
setsebool -P httpd_use_fusefs on
```

Autorisation d'accès de GPG pour Rainloop

ISi vous utilisez un client de messagerie Web Rainloop qui gère GPG/PGP, vous pourriez avoir besoin de ceci

```
setsebool -P httpd_use_gpg on
```

Dépannage

Pour le dépannage de SELinux et de ses profils, essayez d'installer `setroubleshoot` et exécutez la commande :

```
sealert -a /var/log/audit/audit.log > /path/to/mylogfile.txt
```

pour obtenir un rapport qui vous aidera à configurer vos profils SELinux.

Un autre outil pour le dépannage est d'activer une règle dans votre répertoire ownCloud

```
semanage fcontext -a -t httpd_sys_rw_content_t '/var/www/html/owncloud(/.*)?'
restorecon -RF /var/www/html/owncloud
```

Il est beaucoup plus sûr d'avoir une granularité des règles comme dans les exemples décrits au début, aussi, n'utilisez ceci que pour les tests et le dépannage. Cela a un effet similaire à la désactivation de SELinux, n'utilisez donc pas ceci sur des systèmes en production.

Consulter cette discussion sur GitHub pour en apprendre plus la configuration correcte de SELinux pour ownCloud.

Exemples de configurations pour Nginx

Cette page couvre des exemples de configurations Nginx à utiliser pour ownCloud. Veuillez noter que Nginx n'est pas officiellement supporté, et cette page est maintenue par la communauté. Merci aux contributeurs !

- Vous devez insérer le code suivant dans **votre fichier de configuration Nginx** ;
- la configuration suppose qu'ownCloud est installé dans `/var/www/owncloud` et qu'elle est accédée via `http(s)://cloud.example.com` ;
- adaptez les paramètres **server_name**, **root**, **ssl_certificate** et **ssl_certificate_key** à vos votre configuration ;
- assurez-vous que vos certificats SSL soient accessibles en lecture par le serveur (conulter la documentation du module SSL HTTP de nginx).
- les instructions `add_header` sont prises du niveau courant et ne sont pas héritées d'un autre niveau ou propagées vers un autre niveau. Toutes les instructions `add_header` doivent être définies dans chaque niveau nécessaire. Pour une meilleure lisibilité, il est possible de déplacer les instructions d'ajout d'en-tête *communes* dans un fichier séparé et d'inclure ce fichier chaque fois que nécessiare. Cependant, chaque instruction `add_header` doit être écrite sur une seule ligne pour éviter les problèmes de connexion avec les clients de synchronisation.

Exemples de configurations

Faites attention aux retours à la ligne si vous copiez les exemples, car les lignes longues peuvent être modifiées par le formatage de la page.

Merci à @josh4trunks pour avoir fourni ces exemples de configuration.

Vous pouvez utiliser ownCloud en HTTP, mais nous vous encourageons vivement à utiliser SSL/TLS pour chiffrer tout votre trafic serveur et pour protéger les identifiants utilisateurs et les données en transit.

- Supprimer le bloc serveur contenant la redirection ;
- changer **listen 443 ssl** pour **listen 80;** ;
- supprimer **ssl_certificate** et **ssl_certificate_key** ;
- suppriimer **fastcgi_params HTTPS on;**.

ownCloud dans la racine Web de nginx

La configuration suivante doit être utilisée quand ownCloud est placé dans la racine Web de votre installation nginx.

```
upstream php-handler {
    server 127.0.0.1:9000;
    #server unix:/var/run/php5-fpm.sock;
```

```
}

server {
    listen 80;
    server_name cloud.example.com;
    # enforce https
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl;
    server_name cloud.example.com;

    ssl_certificate /etc/ssl/nginx/cloud.example.com.crt;
    ssl_certificate_key /etc/ssl/nginx/cloud.example.com.key;

    # Add headers to serve security related headers
    # Before enabling Strict-Transport-Security headers please read into this topic first.
    #add_header Strict-Transport-Security "max-age=15552000; includeSubDomains";
    add_header X-Content-Type-Options nosniff;
    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-XSS-Protection "1; mode=block";
    add_header X-Robots-Tag none;
    add_header X-Download-Options noopen;
    add_header X-Permitted-Cross-Domain-Policies none;

    # Path to the root of your installation
    root /var/www/owncloud/;

    location = /robots.txt {
        allow all;
        log_not_found off;
        access_log off;
    }

    # The following 2 rules are only needed for the user_webfinger app.
    # Uncomment it if you're planning to use this app.
    #rewrite ^/.well-known/host-meta /public.php?service=host-meta last;
    #rewrite ^/.well-known/host-meta.json /public.php?service=host-meta-json last;

    location = /.well-known/carddav {
        return 301 $scheme://$host/remote.php/dav;
    }
    location = /.well-known/caldav {
        return 301 $scheme://$host/remote.php/dav;
    }

    location /.well-known/acme-challenge { }

    # set max upload size
    client_max_body_size 512M;
    fastcgi_buffers 64 4K;

    # Disable gzip to avoid the removal of the ETag header
    gzip off;

    # Uncomment if your server is build with the ngx_pagespeed module
    # This module is currently not supported.
    #pagespeed off;
```

```
    error_page 403 /core/templates/403.php;
    error_page 404 /core/templates/404.php;

    location / {
        rewrite ^ /index.php$uri;
    }

    location ~ ^/(?:build|tests|config|lib|3rdparty|templates|data)/ {
        return 404;
    }
    location ~ ^/(?:\.|autotest|occ|issue|indie|db_|console) {
        return 404;
    }

    location ~ ^/(?:index|remote|public|cron|core/ajax/update|status|ocs/v[12]|updater/.+|oc
        fastcgi_split_path_info ^(.+\.php)(/.*)$;
        include fastcgi_params;
        fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
        fastcgi_param PATH_INFO $fastcgi_path_info;
        fastcgi_param HTTPS on;
        fastcgi_param modHeadersAvailable true; #Avoid sending the security headers twice
        fastcgi_param front_controller_active true;
        fastcgi_pass php-handler;
        fastcgi_intercept_errors on;
        fastcgi_request_buffering off; # Disponible depuis nginx 1.7.11
    }

    location ~ ^/(?:updater|ocs-provider)(?:$|/) {
        try_files $uri $uri/ =404;
        index index.php;
    }

    # Adding the cache control header for js and css files
    # Make sure it is BELOW the PHP block
    location ~* \.(?:css|js)$ {
        try_files $uri /index.php$uri$is_args$args;
        add_header Cache-Control "public, max-age=7200";
        # Add headers to serve security related headers (It is intended to have those duplic
        # Before enabling Strict-Transport-Security headers please read into this topic firs
        #add_header Strict-Transport-Security "max-age=15552000; includeSubDomains";
        add_header X-Content-Type-Options nosniff;
        add_header X-Frame-Options "SAMEORIGIN";
        add_header X-XSS-Protection "1; mode=block";
        add_header X-Robots-Tag none;
        add_header X-Download-Options noopen;
        add_header X-Permitted-Cross-Domain-Policies none;
        # Optional: Don't log access to assets
        access_log off;
    }

    location ~* \.(?:svg|gif|png|html|ttf|woff|ico|jpg|jpeg)$ {
        try_files $uri /index.php$uri$is_args$args;
        # Optional: Don't log access to other assets
        access_log off;
    }
}
```

ownCloud dans un sous-répertoire de nginx

La configuration suivante doit être utilisée quand ownCloud est placé dans un sous-répertoire de votre installation nginx.

```
upstream php-handler {
    server 127.0.0.1:9000;
    #server unix:/var/run/php5-fpm.sock;
}

server {
    listen 80;
    server_name cloud.example.com;
    # enforce https
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl;
    server_name cloud.example.com;

    ssl_certificate /etc/ssl/nginx/cloud.example.com.crt;
    ssl_certificate_key /etc/ssl/nginx/cloud.example.com.key;

    # Add headers to serve security related headers
    # Before enabling Strict-Transport-Security headers please read into this topic first.
    #add_header Strict-Transport-Security "max-age=15552000; includeSubDomains";
    add_header X-Content-Type-Options nosniff;
    add_header X-Frame-Options "SAMEORIGIN";
    add_header X-XSS-Protection "1; mode=block";
    add_header X-Robots-Tag none;
    add_header X-Download-Options noopen;
    add_header X-Permitted-Cross-Domain-Policies none;

    # Path to the root of your installation
    root /var/www/;

    location = /robots.txt {
        allow all;
        log_not_found off;
        access_log off;
    }

    # The following 2 rules are only needed for the user_webfinger app.
    # Uncomment it if you're planning to use this app.
    #rewrite ^/.well-known/host-meta /owncloud/public.php?service=host-meta last;
    #rewrite ^/.well-known/host-meta.json /owncloud/public.php?service=host-meta-json last;

    location = /.well-known/carddav {
        return 301 $scheme://$host/owncloud/remote.php/dav;
    }
    location = /.well-known/caldav {
        return 301 $scheme://$host/owncloud/remote.php/dav;
    }

    location /.well-known/acme-challenge { }

    location ^~ /owncloud {

        # set max upload size
        client_max_body_size 512M;
        fastcgi_buffers 64 4K;
```

```
        # Disable gzip to avoid the removal of the ETag header
        gzip off;

        # Uncomment if your server is build with the ngx_pagespeed module
        # This module is currently not supported.
        #pagespeed off;

        error_page 403 /owncloud/core/templates/403.php;
        error_page 404 /owncloud/core/templates/404.php;

        location /owncloud {
            rewrite ^ /owncloud/index.php$uri;
        }

        location ~ ^/owncloud/(?:build|tests|config|lib|3rdparty|templates|data)/ {
            return 404;
        }
        location ~ ^/owncloud/(?:\.|autotest|occ|issue|indie|db_|console) {
            return 404;
        }

        location ~ ^/owncloud/(?:index|remote|public|cron|core/ajax/update|status|ocs/v[12]|
            fastcgi_split_path_info ^(.+\.php)(/.*)$;
            include fastcgi_params;
            fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
            fastcgi_param PATH_INFO $fastcgi_path_info;
            fastcgi_param HTTPS on;
            fastcgi_param modHeadersAvailable true; #Avoid sending the security headers twic
            fastcgi_param front_controller_active true;
            fastcgi_pass php-handler;
            fastcgi_intercept_errors on;
            fastcgi_request_buffering off; # Disponible depuis nginx 1.7.11
        }

        location ~ ^/owncloud/(?:updater|ocs-provider)(?:$|/) {
            try_files $uri $uri/ =404;
            index index.php;
        }

        # Adding the cache control header for js and css files
        # Make sure it is BELOW the PHP block
        location ~* \.(?:css|js)$ {
            try_files $uri /owncloud/index.php$uri$is_args$args;
            add_header Cache-Control "public, max-age=7200";
            # Add headers to serve security related headers  (It is intended to have those d
            # Before enabling Strict-Transport-Security headers please read into this topic
            #add_header Strict-Transport-Security "max-age=15552000; includeSubDomains";
            add_header X-Content-Type-Options nosniff;
            add_header X-Frame-Options "SAMEORIGIN";
            add_header X-XSS-Protection "1; mode=block";
            add_header X-Robots-Tag none;
            add_header X-Download-Options noopen;
            add_header X-Permitted-Cross-Domain-Policies none;
            # Optional: Don't log access to assets
            access_log off;
        }

        location ~* \.(?:svg|gif|png|html|ttf|woff|ico|jpg|jpeg)$ {
            try_files $uri /owncloud/index.php$uri$is_args$args;
            # Optional: Don't log access to other assets
```

```
            access_log off;
        }
    }
}
```

Suppression des messages de journalisation

Si vous voyez dans votre fichier journal des messages comme client denied by server configuration: /var/www/data/htaccesstest.txt, ajoutez cette section dans votre fichier de configuration nginx pour les supprimer:

```
location = /data/htaccesstest.txt {
  allow all;
  log_not_found off;
  access_log off;
}
```

Les fichiers JavaScript (.js) ou CSS (.css) ne sont pas servis correctement

Un problème courant avec les configurations personnalisées de nginx est que les fichiers JavaScript (.js) ou CSS (.css) ne sont pas servis correctement, conduisant à des erreurs 404 (Fichier non trouvé) et une interface Web non fonctionnelle.

Ceci peut être dû au bloc:

```
location ~* \.(?:css|js)$ {
```

n'étant pas placé **après** le bloc:

```
location ~ \.php(?:$|/) {
```

D'autres configurations personnalisées, comme la mise en cache des fichiers JavaScript (.js) ou CSS (.css) via gzip peuvent aussi provoquer de tels problèmes.

Ajustement des performances

nginx (<1.9.5) <ngx_http_spdy_module nginx (+1.9.5) <ngx_http_http2_module

Pour utiliser http_v2 avec nginx, vous devez vérifier deux choses :

1.) Ce module n'est pas intégré par défaut à cause d'une dépendance à la version openssl utilisée sur votre système. Il sera activé avec le paramètre de configuration `--with-http_v2_module` pendant la compilation. La dépendance devrait être vérifiée automatiquement. Vous pouvez vérifier la présence du module http_v2 avec la commande `nginx -V 2>&1 | grep http_v2 -o`. Un exemple de la manière de compiler nginx est décrit dans la section « Configuration de nginx avec le module `nginx-cache-purge` » ci-dessous.

2.) Si vous avez utilisé SPDY auparavant, la configuration de nginx doit être modifiée de `listen 443 ssl spdy;` pour `listen 443 ssl http2;`.

nginx : mises en cache des vignettes de la galerie d'ownCloud

Une des optimisations possible pour ownCloud avec nginx comme serveur Web est de combiner le cache FastCGI avec « Cache Purge », un module tiers de nginx qui donne la possibilité de purger les contenus de cahces *FastCGI*, *proxy*, *SCGI* et *uWSGI*. Ce mécanisme accélère la présentation des vignettes car il transfère les requêtes à nginx et minimise les invocations PHP qui auraient sans cela été faites pour chaque vignette à chaque fois.

La procédure suivante est basée sur un système Ubuntu 14.04. Vous pourriez avoir besoin de l'adapter en fonction de votre système d'exploitation et de sa version.

Note

Contrairement à Apache, nginx ne charge pas les modules dynamiquement. Tous les modules nécessaires doivent être compilés dans nginx. C'est une des raisons pour les perfomances de nginx. Nous supposeront qu'il existe déjà une configuration opérationnelle définie comme indiqué dans la documentation d'ownCloud.

Vérification du module nginx

Tout d'abord, il est nécessaire de vérifier si votre installation nginx contient le module `nginx cache purge` compilé:

```
nginx -V 2>&1 | grep ngx_cache_purge -o
```

Si la sortie contient `ngx_cache_purge`, vous pouvez continuer la configuration, sinon vous devez compiler manuellement nginx avec le module requis.

Compilation de nginx avec le module `nginx-cache-purge`

1. **Préparation:**

```
cd /opt
wget http://nginx.org/keys/nginx_signing.key
sudo apt-key add nginx_signing.key
sudo vi /etc/apt/sources.list.d/nginx.list
```

Ajoutez les lignes suivantes (si elle est différente, remplacez `{trusty}` par le nom de votre distribution):

```
deb http://nginx.org/packages/mainline/ubuntu/ trusty nginx
deb -src http://nginx.org/packages/mainline/ubuntu/ trusty nginx
```

Puis exécutez la commande `sudo apt-get update`.

> ***Note***
>
> Si vous êtes téméraire et que vous souhaitez installer les tous derniers paquets de nginx contenant les dernières fonctionnalités, vous devrez installer nginx à partir du dépôt principal. Sur la page d'accueil de nginx : « En général, vous devriez toujours déployer nginx à partir de sa branche principale ». Si vous voulez installer le nginx standard nginx à partir de dernière branche principale mais sans compiler de module supplémentaire, exécutez juste la commande `sudo apt-get install nginx`.

2. **Téléchargez les sources nginx à partir du dépôt ppa**

```
cd /opt
sudo apt-get build-dep nginx
sudo apt-get source nginx
```

3. **Téléchargez le ou les modules à compiler et configurer les arguments de compilation**

```
ls -la
```

Veuillez remplacer `{release}` par la version téléchargée:

```
cd /opt/nginx-{release}/debian
```

Si le dossier « modules » n'est pas présent :

```
sudo mkdir modules
cd modules
sudo git clone https://github.com/FRiCKLE/ngx_cache_purge.git
sudo vi /opt/nginx-{release}/debian/rules
```

Si elle n'est pas présente, ajoutez la ligne suivante tout en haut sous:

```
#export DH_VERBOSE=1:
MODULESDIR = $(CURDIR)/debian/modules
```

et à la fin de chaque commande `configure`, ajoutez:

```
--add-module=$(MODULESDIR)/ngx_cache_purge
```

N'oubliez pas d'ajouter une barre de fration inversée `\` pour chaque ligne précédente. Les paramètres devraient maintenant ressembler à ceci:

```
--with-cc-opt="$(CFLAGS)" \
--with-ld-opt="$(LDFLAGS)" \
--with-ipv6 \
--add-module=$(MODULESDIR)/ngx_cache_purge
```

4. **Compilez et installez nginx**

```
cd /opt/nginx-{release}
sudo dpkg-buildpackage -uc -b
ls -la /opt
sudo dpkg --install /opt/nginx_{release}~{distribution}_amd64.deb
```

5. **Vérifiez si la compilation et l'installation du module ngx_cache_purge ont réussi**

```
nginx -V 2>&1 | grep ngx_cache_purge -o
```

Cette commande devrait maintenant afficher : `ngx_cache_purge`

Pour afficher la version de nginx ainsi que toutes les fonctionnalités compilées et installées:

```
nginx -V 2>&1 | sed s/" --"/"\n\t--"/g
```

6. **Bloquez la mise à jour de nginx par apt-get**

```
sudo dpkg --get-selections | grep nginx
```

Pour chaque composant de nginx component listé, lancez `sudo apt-mark hold <composant>`.

7. **Vérifications régulières de mises à jour de nginx**

Visitez régulièrement la page des nouvelles de nginx et reproduisez les étapes 2 à 5 en cas de mise à jour.

Configuration de nginx avec le module `nginx-cache-purge`

1. **Préparation** Créez le répertoire où nginx stockera les vignettes mises en cache. Remplacez `{path}` par le chemin correspondant à votre configuration :

```
sudo mkdir -p /usr/local/tmp/cache
```

2. **Configuration**

```
sudo vi /etc/nginx/sites-enabled/{your-ownCloud-nginx-config-file}
```

Ajoutez au *début* mais *en dehors* du bloc `server{}`:

```
# cache_purge
fastcgi_cache_path {path} levels=1:2 keys_zone=OWNCLOUD:100m inactive=60m;
map $request_uri $skip_cache {
     default 1;
     ~*/thumbnail.php 0;
     ~*/apps/galleryplus/ 0;
     ~*/apps/gallery/ 0;
}
```

Note

Veuillez adopter ou supprimer les expressions rgulières du bloc `map` en fonction de vos besoins et de votre version d'ownCloud. À la place des correspondances (mappings), vous pouvez utiliser autant d'instructions `if` que nécessaire dans votre bloc `server{}`:

```
set $skip_cache 1;
if ($request_uri ~* "thumbnail.php")      { set $skip_cache 0; }
if ($request_uri ~* "/apps/galleryplus/") { set $skip_cache 0; }
if ($request_uri ~* "/apps/gallery/")     { set $skip_cache 0; }
```

Ajoutez à *l'intérieur* du bloc `server{}`, comme dans cet exemple de configuration:

```
# cache_purge (with $http_cookies we have unique keys for the user)
fastcgi_cache_key $http_cookie$request_method$host$request_uri;
fastcgi_cache_use_stale error timeout invalid_header http_500;
fastcgi_ignore_headers Cache-Control Expires Set-Cookie;

location ~ \.php(?:$/) {
      fastcgi_split_path_info ^(.+\.php)(/.+)$;

      include fastcgi_params;
      fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name;
      fastcgi_param PATH_INFO $fastcgi_path_info;
      fastcgi_param HTTPS on;
      fastcgi_pass php-handler;

      # cache_purge
      fastcgi_cache_bypass $skip_cache;
      fastcgi_no_cache $skip_cache;
      fastcgi_cache OWNCLOUD;
      fastcgi_cache_valid  60m;
      fastcgi_cache_methods GET HEAD;
      }
```

Note

Note concernant le paramètre `fastcgi_pass`` : `adaptez à votre configuration. Dans cet exemple, un ``upstream` a été défini dans un fichier de configuration globale de nginx. Cela peut ressembler à ceci:

```
upstream php-handler {
    server unix:/var/run/php5-fpm.sock;
    # or
    # server 127.0.0.1:9000;
  }
```

3. **Test de la configuration**

```
sudo nginx -s reload
```

- Lancez votre navigateur et videz le cache ;
- connectez-vous à votre instance ownCloud, ouvrez l'application Galerie et déplacez-vous dans les dossiers pour observer la génération des vignettes pour la première fois ;
- vous pouvez aussi observer, par exemple avec `htop`, la montée en charge de votre système lors de la génération des vignettes ;
- rendez-vous dans une autre application, déconnectez-vous et reconnectez-vous ;
- ouvrez à nouveau l'application Galrie et naviguez dans les dossiers où vous vous êtes rendus précédemment. Les vignettes devraient apparaître plus ou moins instantanément ;
- `htop` ne montrera pas de charge processeur additionnelle lors du traitement, contrairement à précédemment.

Configuration serveur ownCloud

Avertissements sur la page d'administration

Votre serveur ownCloud dispose d'un vérificateur de configuration intégré qui rapporte ses résultats en haut de la page d'administration. Voici quelques-uns de ces avertissements et ce qui peut être fait pour y remédier.

Avertissements de cache

« Aucun cache mémoire n'est configuré. Si possible, configurez un cache pour augmenter les performances. ». ownCloud peut gérer plusieurs extensions de cache PHP :

- APC (PHP 5.4 seulement)
- APCu (PHP 5.5+, version 4.0.6 de l'extension PHP minimum requise)
- Memcached
- Redis (version minimale requise de l'extension PHP : 2.2.6)

Vous verrez cet avertissement si vous n'avez aucun système de cache installé et activé, ou si le cache n'est pas de la version minimale requise. Les anciennes versions sont désactivées en raison de problèmes de performances.

Si vous voyez « *{Cache}* inférieur à la version *{Version}* est installé. Pour des raisons de stabilité et de performances, nous recommandons de mettre à jour la version de *{Cache}*. », vous devez alors procéder à une mise à jour, ou si vous ne l'utilisez pas, le supprimer.

Il n'est pas obligatoire d'utiliser un cache, mais les caches améliorent les performances du serveur. Consulter *Configuration de la mémoire cache*.

Le verrouillage de fichier transactionnel est désactivé

« Le verrouillage transactionnel de fichiers est désactivé. Cela peut causer des conflits en cas d'accès concurrent. »

Veuillez consulter *Verrouillage de fichier transactionnel* pour voir comment configurer correctement votre environnement pour le verrouillage transactionnel de fichier.

Accès au site en HTTP

« Vous accédez à ce site via HTTP. Nous vous recommandons fortement de configurer votre serveur pour forcer l'utilisation de HTTPS, comme expliqué dans notre Guide pour le renforcement et la sécurité ». Veuillez prendre cet avertissement au sérieux. L'utilisation de HTTPS est une mesure de sécurité fondamentale. Vous devez configurer votre serveur Web pour le gérer et il y a quelques paramètres dans la section **Sécurité** de votre page d'administration d'ownCloud à activer. Les pages suivantes décrivent comment activer HTTPS sur les serveurs Web Apache et Nginx.

Activation de SSL (pour Apache)

Utilisation de HTTPS

Exemples de configurations pour Nginx

Pas de résultat pour le test getenv("PATH")

Certains environnements n'arrivent pas à passer une variable PATH valide à ownCloud. *Notes de configuration pour php-fpm* fournit les informations sur la façon de configurer votre environnement.

L'en-tête HTTP « Strict-Transport-Security » n'est pas configuré

« L'en-tête HTTP « Strict-Transport-Security » n'est pas configuré à au moins « 15552000 » secondes. Pour une sécurité accrue, nous recommandons dactiver HSTS comme indiqué dans nos conseils de sécurité. »

L'en-tête doit être configuré dans votre serveur Web en suivant la documentation *Activation de HTTP Strict Transport Security*.

/dev/urandom n'est pas accessible en lecture par PHP

« /dev/urandom n'est pas lisible par PHP, ce qui est fortement déconseillé pour des raisons de sécurité. Consultez la documentation pour avoir plus d'informations à ce sujet. »

Ce message doit également être pris au sérieux. Veuillez consulter la documentation *Accès en lecture de PHP à /dev/urandom*.

Le serveur Web n'est pas configuré correctement

« Votre serveur web n'est pas correctement configuré pour la synchronisation de fichiers : l'interface WebDAV semble ne pas fonctionner. »

Sur les forums de la communauté ownCloud une FAQ est maintenue contenant diverses informations et astuces de débogage.

Versions NSS et OpenSSL obsolètes

« cURL utilise une version dépassée d'OpenSSL (OpenSSL/$version). Veuillez mettre à jour votre système d'exploitation ou des fonctionnalités comme l'installation et la mise à jour des applications via le magasin d'applications ou le partage fédéré ne fonctionneront pas de manière fiable. »

« cURL utilise une version dépassée de NSS (NSS/$version). Veuillez mettre à jour votre système d'exploitation ou des fonctionnalités comme l'installation et la mise à jour des applications via le magasin d'applications ou le partage fédéré ne fonctionneront pas de manière fiable. »

Il existe des bogues connus dans les anciennes versions d'OpenSSL et de NSS conduisant à un mauvais fonctionnement en combinaison avec les hôtes distants utilisant SNI, une technologie utilisée par la plupart des sites Web HTTPS. Pour s'assurer qu'ownCloud fonctionnera correctement, vous devez mettre à jour OpenSSL au moins en version 1.0.2b ou 1.0.1d. Pour NSS, la version du correctif dépend de votre distribution et une heuristique fonctionne reproduisant ce bogue. Des distributions comme RHEL/CentOS ont ce problème toujours en cours.

Votre serveur Web n'est pas configuré correctement pour résoudre /.well-known/caldav/ ou /.well-known/carddav/

Ces deux URL doivent être correctement redirigée vers le frontal DAV d'ownCloud. Veuillez consulter *Service de découverte* pour plus d'informations.

Certains fichiers n'ont pas passé le contrôle d'intégrité

Veuillez consulter *Correction des messages d'intégrité de code invalide* pour corriger ce problème.

> Please refer to the *Correction des messages d'intégrité de code invalide* documentation how to debug this issue.

Votre base de données n'utilise pas le niveau d'isolation de transaction « READ COMMITED »

« Votre base de données n'utilise pas le niveau d'isolation de transaction « READ COMMITED ». Ceci peut provoquer des problèmes lors de l'exécution de plusieurs actions en parallèle ».

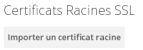
Veuillez consulter *Moteur de stockage MySQL / MariaDB niveau d'isolation de transaction « READ COMMITED »* pour savoir comment configurer votre base de données pour ce prérequis.

Importation de certificats SSL personnel ou global

Les navigateurs modernes essaient de nous protéger et nous bombardent de messages effrayants pour la moindre petite erreur sur un certificat SSL de site Web, ou lorsqu'ils utilisent des certificats auto-signés. Les administrateurs ownCloud font face à cette situation en créant des partages fédérés ou en configurant des stockages externes. Il n'y a as de contre-indication à utiliser des certificats auto-signés sur vos propres réseaux : c'est rapide, facile et gratuit.

Importation de certificat SSL personnel

ownCloud dispose de plusieurs méthodes pour importer des certificats auto-signés et ainsi éviter les avertissements de sécurité des navigateurs. Quand vous autorisez vos utilisateurs à créer leur propre stockage externe ou des partages fédérés, ils peuvent importer des certificats SSL à partir de leur page personnelle.



Cliquez sur **Importer un certificat racine** pour ouvrir un sélecteur de fichiers. Vous pouvez distribuer des copies de vos certificats SSL à vos utilisateurs (avec un partage ownCloud !) ou les utilisateurs peuvent les télécharger à partir de leur navigateur. Cliquez sur l'icône représentant un cadenas jusqu'à voir un bouton **Afficher le certificat**, puis téléchargez-le. Dans Firefox et Chromium, il y a un bouton **Exporter** pour télécharger votre propre copie du certificat SSL d'un site.

Importation de certificat SSL global

Les importations réalisées dans la page personnelle ne fonctionnent que pour l'utilisateur. Vous pouvez activer des certficats SSL pour tous vos utilisateurs dans la page d'administration d'ownCloud. Pour cela, vous devez ajouter cette ligne dans votre fichier `config.php`

```
'enable_certificate_management' => true,
```

Vous aurez alors un bouton **Importer un certificat racine** dans votre page d'administration, tout comme celui de la page personnelle.

Utilisation d'OCC pour importer et gérer les certificats SSL

La commande `occ` dispose d'options pour lister et gérer vos certificats SSL

```
security:certificates         liste les certificats de confiance
security:certificates:import  importe les certificats de confiance
security:certificates:remove  supprime les certificats de confiance
```

Consulter *Utilisation de la commande occ* pour connaître l'utilisation de la commande `occ`.

Utilisation de la commande occ

La commande ownCloud `occ` (ownCloud console) est l'interface de ligne de commande d'ownCloud. Vous pouvez réaliser beaucoup d'opérations sur le serveur avec `occ`, comme l'installation et la mise à jour d'ownCloud, la gestion des utilisateurs, du chiffrement, des mots de passe, des paramètres LDAP et bien plus encore.

`occ` se trouve dans le répertoire owncloud/. Par exemple : /var/www/owncloud sous Ubuntu Linux. `occ` est un script PHP. **Vous devez l'exécuter en tant qu'utilisateur HTTP** pour s'assurer que les permissions correctes sont préservées sur vos fichiers et répertoires d'ownCloud. À compter de la version 8.2 d'ownCloud, il peut être exécuté à partir de n'importe quel répertoire (en spécifiant le chemin d'accès au fichier). Dans les versions précédente d'ownCloud, il devait être lancé à partir du répertoire owncloud/.

Répertoire de la commande occ

- *Conversion de base de données*
- *Chiffrement*
- *Fédération*
- *Opérations sur le fichiers*
- *Fichiers externes*
- *Vérification d'intégrité*
- *l10n, création de fichiers de traduction Javascript pour les applications*
- *Commandes LDAP*
- *Commandes pour les journaux*
- *Commandes de maintenance*
- *Sécurité*
- *Modes Shibboleth (Édition Entreprise seulement)*
- *Corbeille*
- *Commandes utilisateurs*
- *Versions*
- *Installation en ligne de commande*
- *Mise à jour en ligne de commande*
- *Authentification à deux facteurs*
- *Désactivation d'utilisateurs*

Lancement d'occ en tant qu'utilisateur HTTP

Le nom de l'utilisateur HTTP diffère selon les distributions Linux. Consulter *Renforcement des permissions de répertoires* pour savoir comment trouver votre utilisateur HTTP.

- L'utilisateur et le groupe HTTP est « www-data » pour Debian/Ubuntu.
- L'utilisateur et le groupe HTTP est « apache » pour Fedora/CentOS.
- L'utilisateur et le groupe HTTP est « http » pour Arch Linux.
- L'utilisateur HTTP est « wwwrun » et le groupe HTTP « www » pour openSUSE.

Si votre serveur HTTP est configuré pour utiliser une version de PHP différente de celle par défaut (/usr/bin/php), `occ` doit être lancé avec la même version. Par exemple, sous CentOS 6.5 avec SCL-PHP54 installé, la commande ressemble à ceci:

```
sudo -u apache /opt/rh/php54/root/usr/bin/php /var/www/html/owncloud/occ
```

Lancer `occ` sans option énumère toutes les commandes et options, comme dans cet exemple pour Ubuntu:

```
sudo -u www-data php occ
ownCloud version 9.0.0

Usage:
 command [options] [arguments]

Options:
 -h, --help            Display this help message
 -q, --quiet           Do not output any message
 -V, --version         Display this application version
     --ansi            Force ANSI output
     --no-ansi         Disable ANSI output
 -n, --no-interaction  Do not ask any interactive question
     --no-warnings     Skip global warnings, show command output only
```

```
 -v|vv|vvv, --verbose  Increase the verbosity of messages: 1 for normal output,
                       2 for more verbose output and 3 for debug

Available commands:
 check                 check dependencies of the server
                       environment
 help                  Displays help for a command
 list                  Lists commands
 status                show some status information
 upgrade               run upgrade routines after installation of
                       a new release. The release has to be
                       installed before.
```

C'est le même résultat que pour la commande `sudo -u www-data php occ list`.

Lancez-le avec l'option `-h` pour avoir une aide sur la syntaxe:

```
sudo -u www-data php occ -h
```

Affichage de la version d'ownCloud:

```
sudo -u www-data php occ -V
  ownCloud version 9.0.0
```

Affichage de l'état du serveur ownCloud:

```
sudo -u www-data php occ status
  - installed: true
  - version: 9.0.0.19
  - versionstring: 9.0.0
  - edition:
```

`occ` a des options, des commandes et des arguments. Les options et les arguments sont facultatifs, alors que les commandes sont obligatoires. La syntaxe est la suivante:

```
occ [options] commande [arguments]
```

Obtention des informations détaillées des commandes avec la commande `help`, comme dans cet exemple pour la commande `maintenance:mode`:

```
sudo -u www-data php occ help maintenance:mode
Usage:
 maintenance:mode [options]

Options:
     --on              enable maintenance mode
     --off             disable maintenance mode
 -h, --help            Display this help message
 -q, --quiet           Do not output any message
 -V, --version         Display this application version
     --ansi            Force ANSI output
     --no-ansi         Disable ANSI output
 -n, --no-interaction  Do not ask any interactive question
     --no-warnings     Skip global warnings, show command output only
 -v|vv|vvv, --verbose  Increase the verbosity of messages: 1 for normal output,
                       2 for more verbose output and 3 for debug
```

La commande `status` a une option permettant de définir le format de sortie. Par défaut, c'est du texte brut, mais cela peut être aussu du `json`:

```
sudo -u www-data php occ status --output=json
{"installed":true,"version":"9.0.0.19","versionstring":"9.0.0","edition":""}
```

ou du `json_pretty`:

```
sudo -u www-data php occ status --output=json_pretty
{
   "installed": true,
   "version": "9.0.0.19",
   "versionstring": "9.0.0",
   "edition": ""
}
```

Cette option de format de sortie est disponible pour toutes les commandes de type liste : `status`, `check`, `app:list`, `config:list`, `encryption:status` et `encryption:list-modules`.

Commandes pour les applications

La commande `app` liste, active et désactive les applications:

```
app
 app:check-code   vérifie la conformité du code
 app:disable      désactive une application
 app:enable       active une application
 app:getpath      récupère le chemin d'accès absolu d'une application
                  (ajouté dans la version in 9.0)
 app:list         liste toutes les applications disponibles
```

Pour lister toutes les applications installées et afficher si elles sont activées ou désactivées:

```
sudo -u www-data php occ app:list
```

Pour activer une application, par exemple External Storage:

```
sudo -u www-data php occ app:enable files_external
files_external enabled
```

Pour désactiver une application:

```
sudo -u www-data php occ app:disable files_external
files_external disabled
```

La commande `app:check-code` permet d'effectuer plusieurs vérifications : elle vérifie si une application utilise les API publiques d'ownCloud (`OCP`) ou des API privées (`OC_`). Elle vérifie la présence de méthodes obsolètes et la validité du fichier `info.xml`. Par défaut, toutes les vérifications sont activées. L'application Activité est un exemple d'application correctement formatée:

```
sudo -u www-data php occ app:check-code notifications
App is compliant - awesome job!
```

Si votre application a des problèmes, la sortie ressemblera à ceci:

```
sudo -u www-data php occ app:check-code foo_app
Analysing /var/www/owncloud/apps/files/foo_app.php
4 errors
   line   45: OCP\Response - Static method of deprecated class must not be
   called
   line   46: OCP\Response - Static method of deprecated class must not be
   called
   line   47: OCP\Response - Static method of deprecated class must not be
   called
   line   49: OC_Util - Static method of private class must not be called
```

Vous pouvez obtenir le chemin d'accès complet d'une application:

```
sudo -u www-data php occ app:getpath notifications
/var/www/owncloud/apps/notifications
```

Sélecteur des tâches d'arrière-plan

Utilisez la commande `background` pour sélectionner le programmateur à utiliser pour contrôler les tâches d'arrière-plan : Ajax, Webcron ou Cron. C'st la même chose que d'utiliser la section **Cron** sur votre page d'administration:

```
background
 background:ajax       utilise ajax pour exécuter les tâches d'arrière-plan
 background:cron       utilise cron pour exécuter les tâches d'arrière-plan
 background:webcron    utilise webcron pour exécuter les tâches d'arrière-plan
```

Cet exemple sélectionne Ajax:

```
sudo -u www-data php occ background:ajax
  Set mode for background jobs to 'ajax'
```

Les deux autres commandes sont :

- `background:cron`
- `background:webcron`

Consulter *Tâches d'arrière-plan* pour en savoir plus.

Commandes config

Les commandes `config` sont utilisées pour configurer le serveur ownCloud:

```
config
 config:app:delete      supprime une valeur de configuration d'une application
 config:app:get         récupère une valeur de configuration d'une application
 config:app:set         définit une valeur de configuration d'une application
 config:import          importe une liste de configurations
 config:list            Liste toutes les configurations
 config:system:delete   supprime une valeur de configuration système
 config:system:get      récupère une valeur de configuration système
 config:system:set      définit une valeur de configuration système
```

Vous pouvez lister toutes les valeurs de configurations en une seule commande:

```
sudo -u www-data php occ config:list
```

Par défaut, les mots de passe et autres données sensibles sont omis du rapport, ainsi le rapport peut être publié publiquement (par exemple, pour un rapport de bogue). Afin de générer un rapport complet contenant toutes les valeurs de configuration, l'option `--private` doit être ajoutée:

```
sudo -u www-data php occ config:list --private
```

Le contenu exporté peut aussi être importé à nouveau pour permettre la configuration rapide pour des instances similaires. La commande d'import effectue seulement l'ajout ou la mise à jour de valeurs. Les valeurs qui existent dans la configuration courante, mais pas dans la configuration qui est importée, ne sont pas modifiées:

```
sudo -u www-data php occ config:import filename.json
```

Il est aussi possible d'importer des fichiers distants:

```
sudo -u www-data php occ config:import < local-backup.json
```

Note

Bien qu'il soit possible de mettre à jour, définir ou supprimer les versions et états d'installation des applications, et ownCloud lui-même, il n'est **pas** recommandé de faire ceci directement. Utilisez plutôt les commandes `occ app:enable`, `occ app:disable` et `occ update`.

Obtention d'une seule valeur de configuration

Ces commandes récupère une seule valeur de configuration d'une application ou du système:

```
sudo -u www-data php occ config:system:get version
9.0.0.19

sudo -u www-data php occ config:app:get activity installed_version
2.2.1
```

Définition d'une seule valeur de configuration

Ces commandes définissent une seule valeur de configuration d'une application ou du système:

```
sudo -u www-data php occ config:system:set logtimezone
--value="Europe/Berlin"
System config value logtimezone set to Europe/Berlin

sudo -u www-data php occ config:app:set files_sharing
incoming_server2server_share_enabled --value="yes" --type=boolean
Config value incoming_server2server_share_enabled for app files_sharing set to yes
```

La commande `config:system:set` crée la valeur si elle n'existe pas déjà. Pour mettre à jour une valeur, ajouter `--update-only`:

```
sudo -u www-data php occ config:system:set doesnotexist --value="true"
--type=boolean --update-only
Value not updated, as it has not been set before.
```

Veuillez noter que pour définir une valeur booléenne, flottante ou entière dans le fichier de configuration, vous devez spécifier le type de votre commande. Ceci s'applique seulement à la commande `config:system:set`. Voici la liste des types :

- `boolean`
- `integer`
- `float`
- `string` (par défaut)

Si vous voulez par exemple désactiver le mode de maintenance, lancez la commande suivante:

```
sudo -u www-data php occ config:system:set maintenance --value=false
--type=boolean
ownCloud is in maintenance mode - no app have been loaded
System config value maintenance set to boolean false
```

Définition d'un tableau de valeurs de configuration

Certaines configurations (comme les domaines de confiance) sont des tableaux de données. Afin de définir (ou de récupérer) la valeur d'une clé, vous devez spécifier plusieurs noms de `config` séparés par des espaces:

```
sudo -u www-data php occ config:system:get trusted_domains
localhost
owncloud.local
sample.tld
```

Pour remplacer `sample.tld` par `example.com` trusted_domains => 2 doit être défini:

```
sudo -u www-data php occ config:system:set trusted_domains 2
--value=example.com
System config value trusted_domains => 2 set to string example.com

sudo -u www-data php occ config:system:get trusted_domains
localhost
owncloud.local
example.com
```

Suppression d'une seule valeur de configuration

Ces commandes suppriment une seule valeur de configuration d'une application ou du système:

```
sudo -u www-data php occ config:system:delete maintenance:mode
System config value maintenance:mode deleted

sudo -u www-data php occ config:app:delete appname provisioning_api
Config value provisioning_api of app appname deleted
```

Par défaut, la commande delete ne signalera pas si la configuration n'avait pas été définie auparavant. Si vous voulez en être informé, ajouter l'option `--error-if-not-exists`:

```
sudo -u www-data php occ config:system:delete doesnotexist
--error-if-not-exists
Config provisioning_api of app appname could not be deleted because it did not
exist
```

Commandes Dav

Un ensemble de commandes pour créer des carnets d'adresses, des agendas, et pour migrer des carnets d'adresses de la version 8.2 à la version 9.0:

```
dav
 dav:create-addressbook         crée un carnet d'adresses DAV
 dav:create-calendar            crée un agenda DAV
 dav:sync-birthday-calendar     synchronise l'agenda des anniversaires
 dav:sync-system-addressbook    synchronise les utilisateurs vers le carnet d'adresses
                                système
```

La syntaxe pour `dav:create-addressbook` et `dav:create-calendar` est `dav:create-addressbook [utilisateur] [nom]`. Cet exemple crée le carnet d'adresses `mollybook` pour l'utilisateur `molly`:

```
sudo -u www-data php occ dav:create-addressbook molly mollybook
```

Cet exemple crée un nouvel agenda pour `molly`:

```
sudo -u www-data php occ dav:create-calendar molly mollycal
```

Molly les verra immédiatement sur ses pages Agenda et Contacts.

Dans la version 9.0, le serveur CalDAV a été intégré au cœur du système. Vos agendas et carnets d'adresses devraient être migrés automatiquement lors de la mise à jour. Si qulque chose se passe mal, vous pouvez essayer de les migrer manuellement. Supprimez d'abord tout aganda ou carnet d'adresses partiellement migrés. Lancez ensuite cette commande pour migrer les contacts de l'utilisateur:

```
sudo -u www-data php occ dav:migrate-addressbooks [user]
```

Lancez cette commande pour migrer les agendas:

```
sudo -u www-data php occ dav:migrate-calendars [user]
```

Consulter ownCloud 9.0 - calendar migration analysis pour obtenir de l'aide sur le dépannage et rapporter des problèmes.

`dav:sync-birthday-calendar` ajoute tous les anniversaires à votre agenda des carnets d'adresses partagés avec vous. Cet exemple synchronise votre agenda à partir de celui de bernie:

```
sudo -u www-data php occ dav:sync-birthday-calendar bernie
```

`dav:sync-system-addressbook` synchronise tous les utilisateurs vers le carnet d'adresses système:

```
sudo -u www-data php occ dav:sync-system-addressbook
```

Ajouté dans la version 9.0.

Conversion de base de données

Le système de base de données SQLite et très bien pour les tests et pour les serveurs ownCloud avec un seul utilisateur n'utilisant pas de client de synchronisation. Pour les serveurs de production avec plusieurs utilisateurs, vous devez utiliser MariaDB, MySQL ou PostgreSQL. Vous pouvez utiliser `occ` pour convertir votre base SQLite vers l'un de ces système de gestion de base de données.

```
db
 db:convert-type           convertit la base de données ownCloud vers le nouveau
                           type configuré
 db:generate-change-script génère le script de modification à partir de la base de données
                           actuellement connectée vers db_structure.xml
```

Vous avez besoin :

- d'installer le système de gestion de base de données désiré et son connecteur PHP ;
- du nom et du mot de passe d'une administrateur du moteur de base de données ;
- du port du moteur de base de données si celui-ci n'est pas le port standard.

Cet exemple convertit une base SQLite en une base MySQL/MariaDB:

```
sudo -u www-data php occ db:convert-type mysql oc_dbuser 127.0.0.1
oc_database
```

Pour une explication plus détaillée, veuillez consulter *Conversion du moteur de base de données*

Chiffrement

`occ` contient un jeu de commandes complet pour la gestion du chiffrement:

```
encryption
 encryption:change-key-storage-root   change la racine de stockage des clés
 encryption:decrypt-all               désactive le chiffrement côté serveur et
                                      déchiffre tous les fichiers
 encryption:disable                   désactive le chiffrement
 encryption:enable                    active le chiffrement
 encryption:enable-master-key         active la clé maîtresse. Seulement disponible
                                      pour de nouvelles installations sans donnée
                                      chiffrée existante ! Il n'y a pas de moyen de
                                      désactiver ceci après.
 encryption:encrypt-all               chiffre toues les fichiers pour tous les utilisateurs
 encryption:list-modules              liste tous les modules de chiffrement disponibles
 encryption:migrate                   migration initiale en chiffrement 2.0
 encryption:set-default-module        définit le module de chiffrement par défaut
 encryption:show-key-storage-root     affiche la racine de stockage des clés
 encryption:status                    affiche l'état actuel du chiffrment
```

`encryption:status` indique si le chiffrement est activé et affiche le module de chiffrement par défaut. Pour activer le chiffrement, vous devez d'abord activer l'application Encryption, puis exécuter `encryption:enable`:

```
sudo -u www-data php occ app:enable encryption
sudo -u www-data php occ encryption:enable
sudo -u www-data php occ encryption:status
 - enabled: true
 - defaultModule: OC_DEFAULT_MODULE
```

La commande `encryption:change-key-storage-root` déplace vos clés de chiffrement vers un nouveau répertoire. Elle accepte un seul argument, `nouvelleRacine`, qui définit votre nouveau répertoire. Le répertoire doit exister et le chemin doit être relatif à votre répertoire racine d'ownCloud :

```
sudo -u www-data php occ encryption:change-key-storage-root ../../etc/oc-keys
```

Vous pouvez consulter l'emplacement actuel de votre répertoire de clés avec la commande:

```
sudo -u www-data php occ encryption:show-key-storage-root
Current key storage root:  default storage location (data/)
```

`encryption:list-modules` affiche les modules de chiffrement disponibles. Vous ne verrez cette liste que si l'application Encryption est activée. Utilisez `encryption:set-default-module [nom du module]` pour définir le module désiré.

`encryption:encrypt-all` chiffre tous les fichiers de données pour tous les utilisateurs. Vouos devez auparavant passer votre serveur en *mode single-user* pour empêcher toute activité des utilisateurs jusqu'à ce que le chiffrement soit terminé.

`encryption:decrypt-all` déchiffre tous les fichiers de données des utilisateurs. Peut aussi déchiffrer les données d'un seul utilisateur:

```
sudo -u www-data php occ encryption:decrypt freda
```

Les utilisateurs doivent avoir activé leur clé de récupération sur leur page personnelle. Vous devez d'abord passer votre serveur en *mode single-user* pour empêcher toute activité des utilisateurs jusqu'à ce que le déchiffrement soit terminé.

Use `encryption:disable` to disable your encryption module. You must first put your ownCloud server into *single-user mode* to prevent any user activity.

`encryption:enable-master-key` crée une nouvelle clé maîtresse qui est utilisée pour toutes les données utilisateur à la place des clés utilisateur individuelles. Ceci est particulièrement utile pour mettre en place le single-sign on. À n'utiliser que sur de nouvelles installations sans données existantes ou sur les systèmes sur lesquels le chiffrement n'a pas encore été activé. Il n'est pas possible de désactiver ceci.

`encryption:migrate` migre les clés de chiffrement après une mise à jour majeure d'ownCloud. Vous pouvez aussi spécifier des utilisateurs individuels dans une liste séparés par des espaces.

Consulter *Configuration du chiffrement* pour en apprendre plus.

Fédération

> **Note**
>
> Cette commande n'st disponible que si l'application Fédération (`federation`) est activée.

Synchronise les carnets d'adresses de tous les serveurs ownCloud fédérés:

```
federation:sync-addressbooks  synchronise les carnets d'adresses de tous les
                              serveurs ownCloud fédérés
```

À compter de la version 9.0 d'ownCloud, les serveurs connectés avec des partages fédérés peuvent partager les carnets d'adresses de leurs utilisateurs et proposent l'auto-complétion des noms d'utilisateurs dans les dialogues de partage. Utilisez cette commande pour synchroniser les serveurs fédérés:

```
sudo -u www-data php occ federation:sync-addressbooks
```

Ajouté dans la version 9.0.

Opérations sur le fichiers

`occ` dispose de trois commandes pour la gestion de fichiers dans ownCloud:

```
files
 files:cleanup              vide le cache de fichiers
 files:scan                 re-balaye le système de fichiers
 files:transfer-ownership   tous les fichiers et dossiers sont transférés vers
                            un autre utilisateur - les partages sont aussi déplacés. (Ajouté
```

La commande `files:scan` balaye le système de fichiers pour identifier les nouveaux fichiers et met à jour le cache de fichiers. Vous pouvez re-balayer tous les fichiers, par utilisateur, avec une liste d'utilisateurs séparés par des espaces et limiter le chemin de recherche. Si vous n'utilisez pas l'option `--quiet`, des statistiques seront affichées à la fin du balayage:

```
sudo -u www-data php occ files:scan --help
  Usage:
  files:scan [-p|--path="..."] [-q|--quiet] [-v|vv|vvv --verbose] [--all]
  [user_id1] ... [user_idN]

Arguments:
  user_id               balaye tous les fichiers du (des) utilisateur(s) indiqué(s)

Options:
  --path                limite le balayage à l'utilisateur ou au chemin donné
  --all                 balaye tous les fichiers de tous les utilisateurs connus
  --quiet               supprime toute sortie
  --verbose             les fichiers et les répertoires sont affichés pendant
                        le balayage
  --unscanned           analyse seulement les fichiers non analysés précédemment
```

Les niveaux de verbosité `-vv` ou `-vvv` sont automatiquement redéfinis à `-v`.

Note pour l'option --unscanned : En général, il y a une tâche de fond (à l'aide de cron) qui fera ce balayage périodiquement. L'option --unscanned rend possible de déclencher ceci en ligne de commande.

Lors de l'utilisation de l'option `--path`, le chemin doit être constitué des éléments suivants:

```
"id_utilisateur/files/chemin"
  ou
"id_utilisateur/files/nom_du_montage"
  ou
"id_utilisateur/files/nom_du_montage/chemin"
```

où le terme `files` est obligatoire.

Exemple:

```
--path="/alice/files/Musique"
```

Dans l'exemple ci-dessus, l'identifiant d'utilisateur `alice` est déterminé implicitement à partir du chemin donné.

Les paramètres `--path`, `--all` et `[id_utilisateur]` sont exclusifs - un seul peut être spécifié à la fois.

`files:cleanup` nettoie le cache de fichiers en supprimant toutes les entrées de fichiers n'ayant pas d'entrées correspondantes dans la table de stockage.

Vous pouvez transférer tous ls fichiers et partage d'un utilisateur vers un autre. Ceci est utile avant de supprimer un utilisateur:

```
sudo -u www-data php occ files:transfer-ownership <utilisateur-source>
<utilisateur-destination>
```

Fichiers externes

Ces commandes remplacent le fichier de configuration `data/mount.json` utilisé dans les versions ownCloud antérieures à la version 9.0.

> ***Note***
>
> Ces commandes ne sont disponibles que si l'application « External storage support » (`files_external`) est activée.

Les commandes pour la gestion des stockages externes sont:

```
files_external
 files_external:applicable  gère les utilisateurs et groupe d'un montage
 files_external:backends    affiche les méthodes d'authentification et les
                            services de stockage disponibles
```

```
 files_external:config      gère le service de configuration d'un montage
 files_external:create      crée une nouvelle configuration de montage
 files_external:delete      supprime un montage externe
 files_external:export      exporte les configurations de montage
 files_external:import      importe les configurations de montage
 files_external:list        liste les montages configurés
 files_external:option      gère les options de montage d'un point de montage
 files_external:verify      vérifie la configuration du montage
```

Ces commandes répliquent la fonctionnalité de l'interface Web d'ownCloud avec deux fonctionnalités supplémentaires : `files_external:export` et `files_external:import`.

Utilisez `files_external:export` pour exporter tous les points de montage administrateur sur la sortie standard et `files_external:export [id_utilisateur]` pour exporter les points de montage de l'utilisateur ownCloud spécifié.

Utilisez `files_external:import [nom du fichier]` pour importer les anciennes configurations JSON et pour copier les configurations de montages externes sur un autre serveur ownCloud.

Ajouté dans la version 9.0.

Vérification d'intégrité

Les applications identifiées comme officielles DOIVENT être signées numériquement à compter de la version 9.0 d'ownCloud. Les applications officielles non signées ne seront plus installable. La signature du code est optionnelle pour toutes les applications tierces:

```
integrity
 integrity:check-app                vérifie l'intégrité d'une application en utilisant une
 integrity:check-core               vérifié l'intégrité du cœur du système en utilisant une
 integrity:sign-app                 signe une application en utilisant une clé privée
 integrity:sign-core                signe le cœur du système en utilisant une clé privée
```

Après la création de votre clé, signez une application comme dans l'exemple ci-après:

```
sudo -u www-data php occ integrity:sign-app --privateKey=/Users/lukasreschke/contacts.key --
```

Vérifiez votre application:

```
sudo -u www-data php occ integrity:check-app --path=/pathto/app appname
```

Si rien n'est renvoyé, votre application a été signée correctement. Si un message est renvoyé, c'est qu'il y a une erreur. Consultez Code Signing dans le manuel du développeur pour des informations plus détaillées.

`integrity:sign-core` est uniquement destiné aux développeurs d'ownCloud.

Consulter *Signature du code* pour en apprendre plus.

Ajouté dans la version 9.0.

l10n, création de fichiers de traduction Javascript pour les applications

Cette commande sert aux développeurs d'application pour mettre à jour leur mécanisme de traduction à partir des version ownCloud 7, 8 et supérieures.

Commandes LDAP

> **Note**
>
> Ces commandes sont disponibles seulement quand l'application « LDAP user and group backend » (`user_ldap`) est activée.

Vous pouvez lancer les commandes LDAP suivantes avec `occ`:

```
ldap
 ldap:check-user               vérifie si un utilisateur existe dans l'annuaire LDAP.
 ldap:create-empty-config      crée une configuration LDAP vide
 ldap:delete-config            supprime une configuration LDAP existante
 ldap:search                   exécute une recherche sur un utilisateur ou un groupe
 ldap:set-config               modifie une configuration LDAP
 ldap:show-config              affiche la configuration LDAP
 ldap:show-remnants            affiche les utilisateurs qui n'existent plus dans l'annuaire
                               LDAP mais pou lequels il reste des traces dans ownCloud.
 ldap:test-config              teste une configuration LDAP
```

Rechercher un utilisateur LDAPen utilisant cette syntaxe:

```
sudo -u www-data php occ ldap:search [--group] [--offset="..."]
[--limit="..."] search
```

Cet exemple recherche dans l'attribut givenNames les valeurs commençant par « rob »:

```
sudo -u www-data php occ ldap:search "rob"
```

Ceci trouvera robbie, roberta, and robin. Pour élargir la recherche, pour trouver `jeroboam` par exemple, ajouter le méta-caractère astérisque:

```
sudo -u www-data php occ ldap:search "*rob"
```

Les attributs de recherche utilisateur sont définis avec `ldap:set-config` (ci-dessous). Par exemple, si vos attributs de recherche sont `givenName` et `sn`, vous pouvez trouver les utilisateurs avec le nom et le prénom très rapidement. Par exemple, vous trouverez Terri Hanson en recherchant `te ha`. Les espaces finaux sont ignorés.

Vérifier l'existence d'un utilisateur LDAP. Ceci ne fonctionne que si le serveur ownCloud ext connecté à un serveur LDAP:

```
sudo -u www-data php occ ldap:check-user robert
```

`ldap:check-user` ne lancera pas de recherche s'il trouve une connexion LDAP désactivée. Ceci empêche que les utilisateurs existant sur des connexions LDAP désactivées soient marqués comme supprimés. Si vous êtes certain que l'utilisateur que vou cherchez n'est pas dans l'une des connexions désactivées et existe sur une connexion active, utilisez l'option `--force` pour forcer à vérifier toutes les connexions LDAP actives:

```
sudo -u www-data php occ ldap:check-user --force robert
```

`ldap:create-empty-config` crée une configuration LDAP vide. La première que vous créez n'a pas de `configID`, comme dans cet exemple:

```
sudo -u www-data php occ ldap:create-empty-config
  Created new configuration with configID ''
```

C'est un reliquat des premiers jours, quand il n'y avait pas d'option pour créer des configurations supplémentaires. La seconde configuration et toutes les suivantes auront un identifiant:

```
sudo -u www-data php occ ldap:create-empty-config
   Created new configuration with configID 's01'
```

Vous pouvez alors lister et afficher vos configurations:

```
sudo -u www-data php occ ldap:show-config
```

Ou afficher la configuration pour un seul configID:

```
sudo -u www-data php occ ldap:show-config s01
```

`ldap:delete-config [configID]` supprime une configuration LDAP existante:

```
sudo -u www-data php occ ldap:delete  s01
Deleted configuration with configID 's01'
```

La commande `ldap:set-config` sert à manipuler les configurations, comme dans cet exemple qui définit les attributs de recherche:

```
sudo -u www-data php occ ldap:set-config s01 ldapAttributesForUserSearch
"cn;givenname;sn;displayname;mail"
```

`ldap:test-config` teste si la configuration est correcte et si la connexion au serveur se fait:

```
sudo -u www-data php occ ldap:test-config s01
The configuration is valid and the connection could be established!
```

`ldap:show-remnants` sert à nettoyer les tables de correspondance LDAP et est documenté dans *Purge des utilisateurs LDAP*.

Commandes pour les journaux

Ces commandes permettent d'afficher et de configurer les préférences de journaux d'ownCloud:

```
log
 log:manage     gère la configuration des journaux
 log:owncloud   manipule le service de journaux d'ownCloud
```

Exécuter `log:owncloud` pour voir l'état en cours:

```
sudo -u www-data php occ log:owncloud
Log backend ownCloud: enabled
Log file: /opt/owncloud/data/owncloud.log
Rotate at: disabled
```

Utilisez l'option `--enable` pour activer le fichier journal. Utilisez `--file` pour définir un chemin d'accès au fichier différent. Mettez en place la rotation des fichiers en indiquant la taille en octets avec `--rotate-size`. `0` désactive la rotation.

`log:manage` définit le service de journalisation, le niveau de journalisation et le fuseau horaire. Les paramètres par défaut sont : `owncloud`, `Warning` et `UTC`. Les options disponibles sont les suivantes :

- --backend [owncloud, syslog, errorlog]
- --level [debug, info, warning, error]

Commandes de maintenance

Utilisez ces commandes quand vous mettez à jour ownCloud, quand vous gérez le chiffrement, quand vous réalisez des sauvegardes ou des tâches qui nécessitent d'empêcher l'activité des utilisateurs jusqu'à ce que vous ayez terminé:

```
maintenance
 maintenance:mimetype:update-db       met à jour la base de données des types MIME
                                      et le cache de fichiers
 maintenance:mimetype:update-js       met à jour mimetypelist.js
 maintenance:mode                     définit le mode de maintenance
 maintenance:repair                   répare cette installation
 maintenance:singleuser               définit le mode utilisateur unique
```

`maintenance:mode` verrouille les sessions de tous les utilisateurs connectés, y compris les administrateurs, et affiche un message d'avertissement indiquant que le serveur est en mode maintenance. Les utilisateurs qui ne sont pas déjà connectés ne pourront le faire qu'après la fin de la maintenance. Quand vous sortez le serveur du mode maintenance, les utilisateurs connectés doivent recharger leur page en cours pour continuer à travailler:

```
sudo -u www-data php occ maintenance:mode --on
sudo -u www-data php occ maintenance:mode --off
```

Passer le serveur en mode utilisateur unique, permet aux administrateurs de se connecter et de travailler, mais pas les autres utilisateurs. C'est utile pour faire de la maintenance ou du dépannage sur un serveur en cours d'exécution:

```
sudo -u www-data php occ maintenance:singleuser --on
Single user mode enabled
```

Pour revenir en mode normal:

```
sudo -u www-data php occ maintenance:singleuser --off
Single user mode disabled
```

La commande `maintenance:repair` se lance automatiquement pendant les mises à jour pour nettoyer la base de données. Bien que vous puissiez la lancer manuellement, il n'est généralement pas utile de le faire:

```
sudo -u www-data php occ maintenance:repair
```

`maintenance:mimetype:update-db` met à jour la base de données d'ownCloud et le cache de fichiers avec les types MIME modifiés trouvés dans `config/mimetypemapping.json`. Lancez cette commande après avoir modifié `config/mimetypemapping.json`. Si vous modifiez un type MIME, exécutez `maintenance:mimetype:update-db --repair-filecache` pour appliquer les modifications aux fichiers existants.

Sécurité

Utilisez ces commandes pour gérer les certificats SSL du serveur. C'est utile quand vous créez des partages fédérés avec d'autres serveurs ownCloud utilisant des certificats auto-signés:

```
security
 security:certificates         liste les certificats de confiance
 security:certificates:import  importe les certificats de confiance
 security:certificates:remove  supprime les certificats de confiance
```

Cet exemple liste les certificats installés:

```
sudo -u www-data php occ security:certificates
```

Importer un nouveau certificat:

```
sudo -u www-data php occ security:import /path/to/certificate
```

Supprimer un certificat:

```
sudo -u www-data php occ security:remove [certificate name]
```

Modes Shibboleth (Édition Entreprise seulement)

Note

Cette commande n'est disponible que lorsque l'application « Shibboleth user backend » (`user_shibboleth`) est activée.

`shibboleth:mode` définit votre mode Shibboleth à `notactive`, `autoprovision` ou `ssoonly`:

```
shibboleth:mode [mode]
```

Corbeille

Note

Cette commande n'est disponible que lorsque l'application « Deleted files » (`files_trashbin`) est activée.

```
trashbin
 trashbin:cleanup   purge les fichiers supprimés
 trashbin:expire    expiration des corbeilles utilisateurs
```

La commande `trashbin:cleanup` supprime les fichiers supprimés des utilisateurs spécifiés dans une liste séparés par des espaces, ou de tous les utilisateurs si aucun n'est spécifié:

```
sudo -u www-data php occ trashbin:cleanup
Remove all deleted files
Remove deleted files for users on backend Database
 freda
 molly
 stash
 rosa
 edward
```

Cet exemple purge les fichiers supprimés de molly et freda:

```
sudo -u www-data php occ trashbin:cleanup molly freda
Remove deleted files of   molly
Remove deleted files of   freda
```

`trashbin:expire` supprime seulement les fichiers expirés en fonction du paramètre `trashbin_retention_obligation` du fichier `config.php` (voir la section Fichiers supprimés dans *Paramètres de Config.php*). Par défaut, les fichiers expirés sont supprimés pour tous les utilisateurs, ou vous pouvez indiquer une liste d'utilisateurs séparés par des espaces.

Commandes utilisateurs

La commande `user` crée et supprime les utilisateurs, réinitialise les mots de passe, affiche un rapport simple indiquant combien d'utilisateurs vous avez et quand un utilisateur s'est connecté pour la dernière fois:

```
user
 user:add            ajoute un utilisateur
 user:delete         supprime l'utilisateur spécifié
 user:lastseen       affiche quand l'utilisateur s'est connecté pour
                     la dernière fois
 user:report         affiche le nombre d'utilisateurs ayant un accès
 user:resetpassword  réinitialise le mot de passe de l'utilisateur spécifié
```

Vous pouvez créer un utilisateur avec son nom d'affichage, son nom de connexion et l'appartenance à un groupe avec la commande `user:add`. La syntaxe est la suivante:

```
user:add [--password-from-env] [--display-name["..."]] [-g|--group["..."]]
uid
```

`display-name` correspond au **Nom complet** sur la page personnelle de l'utilisateur dans l'interface Web et `uid` correspond au compte de connexion. Cet exemple ajoute le nouvel utilisateur Layla Smith et l'ajoute aux groupes **users** et **db-admins**. Les groupes qui n'existent pas seront créés:

```
sudo -u www-data php occ user:add --display-name="Layla Smith"
  --group="users" --group="db-admins" layla
  Enter password:
  Confirm password:
  The user "layla" was created successfully
  Display name set to "Layla Smith"
  User "layla" added to group "users"
  User "layla" added to group "db-admins"
```

Rendez-vous sur la page Utilisateurs pour voir le nouvel utilisateur.

`password-from-env` permet de définir le mot de pase utilisateur à partir d'une variable d'environnement. Ceci empêche l'exposition du mot de passe à tous les utilisateurs via la liste des processus et ne sera visible que dans l'historique de l'utilisateur (root) exécutant la commande. Cela permet aussi de créer des scripts pour ajouter plusieurs utilisateurs.

Pour utiliser `password-from-env` vous devez être le root « réel » plutôt que `sudo`, car `sudo` ne récupère pas les bonnes variables. Cet exemple ajoute l'utilisateur Fred Jones:

```
export OC_PASS=newpassword
su -s /bin/sh www-data -c 'php occ user:add --password-from-env
  --display-name="Fred Jones" --group="users" fred'
The user "fred" was created successfully
Display name set to "Fred Jones"
User "fred" added to group "users"
```

Vous pouvez réinitialiser le mot de passe de tout utilisateur, y compris ceux des administrateurs (voir *Réinitialisation d'un mot de passe administrateur perdu*):

```
sudo -u www-data php occ user:resetpassword layla
  Enter a new password:
  Confirm the new password:
  Successfully reset password for layla
```

Vous pouvez aussi utiliser `password-from-env` pour réinitialiser les mots de passe:

```
export OC_PASS=newpassword
su -s /bin/sh www-data -c 'php occ user:resetpassword --password-from-env
  layla'
  Successfully reset password for layla
```

Pour supprimer un utilisateur:

```
sudo -u www-data php occ user:delete fred
```

Afficher la dernière connexion d'un utilisateur:

```
sudo -u www-data php occ user:lastseen layla
  layla's last login: 09.01.2015 18:46
```

Générer un rapport succinct indiquant le nombre d'utilisateurs, y compris ceux d'un système d'authentification externe comme LDAP:

```
sudo -u www-data php occ user:report
+------------------+----+
| User Report      |    |
+------------------+----+
| Database         | 12 |
| LDAP             | 86 |
|                  |    |
| total users      | 98 |
|                  |    |
| user directories | 2  |
+------------------+----+
```

Versions

> ***Note***
>
> Cette commande n'est disponible que si l'application « Versions » (`files_versions`) est activée.

Utilisez cette commande pour supprimer des versions de fichiers pour des utilisateurs spécifiques ou pour tous les utilisateurs quand aucun n'est spécifié:

```
versions
 versions:cleanup   supprime les versions
 versions:expire    expiration des versions de fichiers des utilisateurs
```

Utiliser `versions:cleanup` pour supprimer les versions les plus anciennes pour des utilisateurs spécifiques (en conservant les versions les plus récentes), ou pour tous les utilisateurs si aucun n'est spécifié.

Cet exemple supprime toutes les versions pour tous les utilisateurs:

```
sudo -u www-data php occ versions:cleanup
Delete all versions
Delete versions for users on backend Database
  freda
  molly
  stash
  rosa
  edward
```

Vous pouvez supprimer des versions pour des utilisateurs spécifiques dans une liste séparés par des espaces:

```
sudo -u www-data php occ versions:cleanup freda molly
Delete versions of   freda
Delete versions of   molly

``versions:expire`` supprime seulement les versions ayant expiré en fonction du
paramètre ``versions_retention_obligation`` dans le fichier ``config.php`` (voir la
section Versions de fichiers dans :doc:`config_sample_php_parameters`). Par défaut,
les versions de fichiers ayant expiré sont supprimées pour tous les utilisateurs, ou
vous pouvez indiquer une liste d'utilisateurs séparés par des espaces.
```

Installation en ligne de commande

Ces commandes sont disponibles uniquement après avoir téléchargé et décompressé l'archive ownCloud, sans avoir effectué d'étape d'installation.

Vous pouvez installer ownCloud entièrement en ligne de commande. Après avoir téléchargé l'archive et copié ownCloud dans le répertoire désiré, ou après avoir installé le paquet d'ownCloud packages (Consulter *Méthode d'installation préférée pour Linux* et *Manuel d'installation pour Linux*), vous pouvez utiliser les commandes `occ` au lieu de lancer l'assistant d'installation graphique.

Appliquez les permissions correctes sur les répertoires d'ownCloud (voir *Renforcement des permissions de répertoires*). Ensuite, choisissez vos options``occ``. Ceci liste les options disponibles:

```
sudo -u www-data php /var/www/owncloud/occ
ownCloud is not installed - only a limited number of commands are available
ownCloud version 9.0.0

Usage:
 [options] command [arguments]

Options:
 --help (-h)           affiche ce message d'aide
 --quiet (-q)          ne renvoie aucun message de sortie
 --verbose (-v|vv|vvv) augmente la verbosité des messages : 1 normal,
                       2 plus verbeux, 3 débogage
 --version (-V)        affiche la version de cette application
 --ansi                force la sortie en ANSI
 --no-ansi             désactive ANSI pour la sortie
 --no-interaction (-n) ne pose pas de question interactive

Available commands:
 check                 vérifie les dépendances de l'environnement serveur
 help                  affiche l'aide pour une commande
 list                  liste les commandes
 status                affiche des informations d'état
 app
 app:check-code        vérifie la conformité du code
 l10n
 l10n:createjs         crée des fichiers JavaScript de traduction pour une application donné
```

```
 maintenance
 maintenance:install   installe ownCloud
```

Affichage des options de `maintenance:install`:

```
sudo -u www-data php occ help maintenance:install
ownCloud is not installed - only a limited number of commands are available
Usage:
 maintenance:install [--database="..."] [--database-name="..."]
[--database-host="..."] [--database-user="..."] [--database-pass[="..."]]
[--database-table-prefix[="..."]] [--admin-user="..."] [--admin-pass="..."]
[--data-dir="..."]

Options:
 --database               type de la base de données(par défaut : "sqlite")
 --database-name          nom de la base de données
 --database-host          nom d'hôte du serveur de base de données(par défaut : "localhost")
 --database-user          nom d'utilisateur pour se connecter à la base de données
 --database-pass          mot de passe pour l'utilisateur de la base de données
 --database-table-prefix  préfixe pour toutes les tables (par défaut : oc_)
 --admin-user             Nom d'utilisateur de l'administrateur (par défaut : "admin")
 --admin-pass             mot de passe du compte administrateur
 --data-dir               chemin d'accès au répertoire data (par défaut :
                          "/var/www/owncloud/data")
 --help (-h)              affiche ce message d'aide
 --quiet (-q)             ne renvoie aucun message de sortie
 --verbose (-v|vv|vvv)    augmente la verbosité des messages : 1 normal,
                          2 plus verbeux, 3 débogage
 --version (-V)           affiche la version de cette application
 --ansi                   force la sortie en ANSI
 --no-ansi                désactive ANSI pour la sortie
 --no-interaction (-n)    ne pose pas de question interactive
```

Cet exemple termine l'installation:

```
cd /var/www/owncloud/
sudo -u www-data php occ maintenance:install --database
"mysql" --database-name "owncloud"  --database-user "root" --database-pass
"password" --admin-user "admin" --admin-pass "password"
ownCloud is not installed - only a limited number of commands are available
ownCloud was successfully installed
```

Les base de données gérées sont:

```
- sqlite (SQLite3 - pas pour l'Édition Entreprise)
- mysql (MySQL/MariaDB)
- pgsql (PostgreSQL)
- oci (Oracle - Édition Entreprise seulement)
```

Mise à jour en ligne de commande

Ces commandes sont disponibles uniquement après avoir téléchargé et décompressé l'archive ownCloud ou installé les paquets, et avant de commencer la mise à jour.

Liste toutes les options, comme cet exemple sous CentOS Linux:

```
sudo -u apache php occ upgrade -h
Usage:
upgrade [--skip-migration-test] [--dry-run] [--no-app-disable]

Options:
--skip-migration-test  passe outre la simulation de migration de schéma de la base de donnée
                       et fait la mise à jour directement
```

```
--dry-run              ne lance que la simulation de migration de schéma de la base de donné
                       et ne fait pas la mise à jour
--no-app-disable       passe outre la désactivation des applications tierces
--help (-h)            affiche ce message d'aide
--quiet (-q)           ne renvoie aucun message de sortie
--verbose (-v|vv|vvv)  augmente la verbosité des messages : 1 normal,
                       2 plus verbeux, 3 débogage
--version (-V)         affiche la version de cette application
--ansi                 force la sortie en ANSI
--no-ansi              désactive ANSI pour la sortie
--no-interaction (-n)  ne pose pas de question interactive
```

Quand vous réalisez une mise à jour de votre serveur ownCloud (voir la section Maintenance de ce manuel), il est préférable d'utiliser `occ` plutôt que l'interface graphique pour réaliser l'étape de la mise à jour de la base de données, ce afin d'éviter les dépassements de temporisation. Les scripts PHP invoqués à partir de l'interface Web sont limités à 3600 seconds. Sur de très gros environnements, cela peut ne pas être suffisant, laissant le système dans un état incohérent. Après avoir réalisé toutes les étapes préliminaires (voir *Comment mettre à jour votre serveur ownCloud*) utilisez cette commande pour mettre à jour votre base de données, comme dans cet exemple pour CentOS Linux. Veuillez noter la façon dont les étapes sont détaillées:

```
sudo -u www-data php occ upgrade
ownCloud or one of the apps require upgrade - only a limited number of
commands are available
Turned on maintenance mode
Checked database schema update
Checked database schema update for apps
Updated database
Updating <gallery> ...
Updated <gallery> to 0.6.1
Updating <activity> ...
Updated <activity> to 2.1.0
Update successful
Turned off maintenance mode
```

L'activation de la verbosité affiche les horodatages:

```
sudo -u www-data php occ upgrade -v
ownCloud or one of the apps require upgrade - only a limited number of commands are availabl
2015-06-23T09:06:15+0000 Turned on maintenance mode
2015-06-23T09:06:15+0000 Checked database schema update
2015-06-23T09:06:15+0000 Checked database schema update for apps
2015-06-23T09:06:15+0000 Updated database
2015-06-23T09:06:15+0000 Updated <files_sharing> to 0.6.6
2015-06-23T09:06:15+0000 Update successful
2015-06-23T09:06:15+0000 Turned off maintenance mode
```

S'il y a une erreur, une exception est renvoyée qui sera détaillée dans le fichier journal d'ownCloud. Vous pouvez donc utiliser la sortie standard pour savoir ce qui s'est passé ou pour ouvrir un rapport de bogue:

```
Turned on maintenance mode
Checked database schema update
Checked database schema update for apps
Updated database
Updating <files_sharing> ...
Exception
ServerNotAvailableException: LDAP server is not available
Update failed
Turned off maintenance mode
```

Avant de terminer la mise à jour, ownCloud exécute d'abord une simulation en copiant toutes les tables de la base de données vers de nouvelles tables, et réalise la mise à jour sur celles-ci pour s'assurer que la mise à jour se terminera correctement. Les tables copiées sont supprimées après la mise à jour. Ceci prend deux fois plus de

temps, ce qui sur de grosses installations peut prendre plusieurs heures. Vous pouvez donc omettre cette étape en utilisant l'option `--skip-migration-test` option:

```
sudo -u www-data php occ upgrade --skip-migration-test
```

Vous pouvez réaliser cette simulation manuellement avec l'option `--dry-run`:

```
sudo -u www-data php occ upgrade --dry-run
```

Authentification à deux facteurs

Si une application d'authentification à deux facteurs est activée, elle l'est pour tous les utilisateurs par défaut (bien que l'application puisse décider si l'utilisateur doit s'y conformer). Dans le cas où un utilisateur perd son accès au deuxième facteur (par un téléphone perdu avec une vérification SMS à deux facteurs), l'administrateur peut temporairement désactiver cette vérification à deux facteurs pour cet utilisateur avec la commande occ :

```
sudo -u www-data php occ twofactor:disable <utilisateur>
```

Pour réactiver l'authentification à deux facteurs, utiliser la commande suivante :

```
sudo -u www-data php occ twofactor:enable <utilisateur>
```

Désactivation d'utilisateurs

Les administrateurs peuvent désactiver les utilisateurs avec la commande occ :

```
sudo -u www-data php occ user:disable <utilisateur>
```

Et avec cette commande pour les réactiver :

```
sudo -u www-data php occ user:enable <utilisateur>
```

Veuillez noter qu'une fois les utilisateurs désactivés, leurs navigateurs en session seront déconnectés.

Configuration de l'application Activité

Vous pouvez configurer votre serveur ownCloud pou envoyer automatiquement des courriels de notification à vos utilisateurs pour divers événements comme :

- un fichier ou un dossier a été partagé ;
- un nouveau fichier ou dossier a été créé ;
- un fichier ou un dossier a été modifié ;
- un fichier ou un dossier a été supprimé.

Les utilisateurs peuvent voir les actions (supprimer, ajouter, modifier) qui surviennent sur les fichiers auxquels ils ont accès. Les actions de partage ne sont visibles que pour le propriétaire du partage et ses bénéficiaires.

Activation de l'application Activité

L'application Activité est fournie dans le logiciel ownCloud et est activée par défaut. Si ce n'est pas le cas, rendez-vous sur la page Applications et activez-la.

Configuration d'ownCloud pour l'application Activité

Pour configurer votre serveur ownCloud pour envoyer des courriels de notifications, une *Configuration des courriels* fonctionnelle est obligatoire.

De plus, il est recommandé de configurer la tâche de d'arrière-plan `Webcron` ou `Cron` décrite dans *Tâches d'arrière-plan*.

Il existe aussi une option de configuration `activity_expire_days` disponible dans votre fichier `config.php` (consulter *Paramètres de Config.php*) qui permet de purger les activités anciennes de la base de données.

Configuration de l'antivirus ClamAV

Vous pouvez configurer votre serveur ownCloud pour lancer automatiquement une analyse antivirus pour les nouveaux fichiers téléversés avec l'application Antivirus. L'application Antivirus intègre le moteur antivirus libre ClamAV dans ownCloud. ClamAV détecte toute forme de logiciels malveillants, y compris les chevaux de Troie, les virus et les vers, et opère sur tous les types de fichiers majeurs dont les fichiers Windows, Linux et Mac, les fichiers compressés, les exécutables, les fichiers images, Flash, PDF et bien d'autres. Le démon Freshclam de ClanAV met à jour automatiquement sa base de données de logiciels malveillants à intervalles programmés.

ClamAV fonctionne sous Linux et tout système d'exploitation de type Unix et Microsoft Windows. Cependant, il n'a été testé qu'avec ownCloud sous Linux, et les instructions qui suivent s'appliquent uniquement aux système Linux. Vous devez d'abord installer ClamAV, puis installer et configurer l'application Antivirus dans ownCloud.

Installation de ClamAV

Comme toujours, les différentes distributions Linux installent et configurent ClamAV de différentes manières.

Debian, Ubuntu, Linux Mint

Sur les systèmes Debian et Ubuntu, et leurs dérivés, installez ClamAV avec ces commandes:

```
apt-get install clamav clamav-daemon
```

L'installeur crée automatiquement les fichiers de configuration par défaut et lance les démons `clamd` et `freshclam`. Vous n'avez rien d'autre à faire, cependant, il est conseillé de lire la documentation de ClamAV et de vérifier les paramètres dans `/etc/clamav/`. Vous pouvez activer la journalisation détaillée dans `clamd.conf` et `freshclam.conf` pour déboguer les problèmes que vous pourriez rencontrer.

Red Hat 7, CentOS 7

Sous Red Hat 7 et ses dérivés, vous devez installer les paquets supplémentaires (Extra) pour le dépôt Enterprise Linux (EPEL), puis installer ClamAV:

```
yum install epel-release
yum install clamav clamav-scanner clamav-scanner-systemd clamav-server
clamav-server-systemd clamav-update
```

Ceci installe deux fichiers de configuration : `/etc/freshclam.conf` et `/etc/clamd.d/scan.conf`. Vous devez modifier ces deux fichiers avant de pouvoir utiliser ClamAV. Ces deux fichiers sont bien documentés et `man clamd.conf` et `manfreshclam.conf` expliquent toutes les options. Veuillez vous référer aux fichiers `/etc/passwd` et `/etc/group` pour vérifier l'utilisateur et le groupe utilisés pour ClamAV.

Éditez tout d'abord le fichier `/etc/freshclam.conf` et configurez vos options. `freshclam` met à jour la base de données de logiciels malveillants, vous voulez donc qu'il soit lancé régulièrement pour mettre à jour les signatures de ces logiciels. Lancez-le après l'installation pour pour télécharger votre premier jeu de signatures de logiciels malveillants:

```
freshclam
```

Les paquets EPEL ne contiennent pas de fichier d'initialisation pour `freshclam`, aussi, un moyen facile et rapide est de programmer des vérifications régulières à l'aide d'une tâche cron. Cet exemple montre une tâche exécutée à la 47e minute de chaque heure:

```
# m   h  dom mon dow  command
  47  *  *   *    *  /usr/bin/freshclam --quiet
```

Veuillez éviter tout multiple de 10 car c'est à ces moment-là que les serveurs de ClamAV sont les plus fortement sollicités pour les mises à jour.

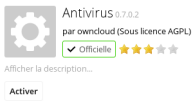
Ensuite, Éditez le fichier `/etc/clamd.d/scan.conf`. quand vous avez terminé, vous devez activer le fichier de service `clamd` et démarrer `clamd`:

```
systemctl enable clamd@scan.service
systemctl start clamd@scan.service
```

Cela devrait suffire. Vous pouvez activer la journalisation détaillée dans `scan.conf` et `freshclam.conf` pour déboguer les problèmes que vous pourriez rencontrer.

Activation de l'application Antivirus

Rendez-vous sur la page Applications d'ownCloud pour l'activer.



Configuring ClamAV on ownCloud

Ensuite, rendez-vous sur votre page d'administration ownCloud et définissez le niveau de journalisationand set your ownCloud logging level to Everything.

Vous pouvez maintenant voir la section de configuration Antivirus sur votre page d'administration.

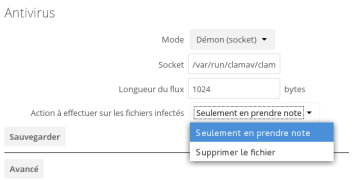
ClamAV peut s'exécuter dans l'un des ces trois modes :

- Démon (Socket) : ClamAV s'exécute sur le même serveur qu'ownCloud. le démon ClamAV, `clamd`, s'exécute en arrière-plan. Quand il n'y a pas d'activité, `clamd` occupe une charge minimale du système. Si vos utilisateurs téléversent beaucoup de fichiers, vous verrez un pic d'utilisation des CPU.
- Démon : ClamAV est exécuté sur un serveur différent. C'est une bonne option pour les serveurs ownCloud gérant de gros volumes de téléversements de fichiers.
- Exécutable : ClamAV est exécuté sur le même serveur qu'ownCloud et la commande `clamscan` est démarrée et arrêtée avec chaque téléversement. `clamscan` est lent et pas toujours fiable pour une utilisation à la demande. Il est préférable d'utiliser un des deux modes démon.

Démon (Socket)

ownCloud devrait détecter votre socket `clamd` et remplir le champ `Socket`. C'est l'option `LocalSocket` du fichier `clamd.conf`. Vous pouvez lancer `netstat` pour le vérifier:

```
netstat -a|grep clam
unix 2 [ ACC ] STREAM LISTENING 15857 /var/run/clamav/clamd.ctl
```

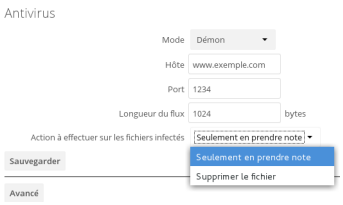


La valeur `Longueur du flux` définit le nombre d'octets lus en une seule passe. 10485760 octets ou dix mégaoctets, est la valeur par défaut. Cette valeur ne peut dépasser la valeur des paramètres PHP `memory_limit` ou de la mémoire physique `memory_limit` si défini à -1 (sans limite).

`Action à effectuer sur les fichiers infectés` donne le choix de noter les alertes sans supprimer les fichiers our de les supprimer les fichiers infectés immédiatement.

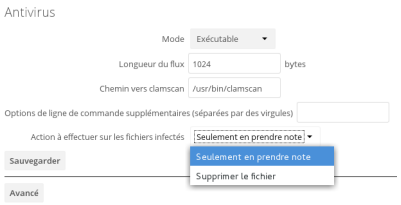
Démon

Pour l'option Démon, vous devez indiquer le nom d'hôte ou l'adresse IP du serveur distant exécutant ClamAV ainsi que le numéro de port.



Exécutable

Pour l'option Exécutable, vous devez indiquer le chemin d'accès complet à `clamscan`, qui est la commande d'analyse interactive de ClamAV. ownCloud devrait le trouver automatiquement.



Quand vous êtes satisfait des optins de ClamAV, vous pouvez changer les paramètres et le niveau de journalisation.

Configuration de la mémoire cache

Vous pouvez améliorer significativement les performances de votre serveur ownCloud en mettant en place de la mémoire cache, où les objets appelés fréquemment sont stockés en mémoire pour une récupération plus rapide. Il existe deux types de cache à utiliser : un cache PHP opcode, communément appelé *opcache*, et le cache de données de votre serveur Web. Si vous n'avez pas installé et activé un système de mémoire cache, vous verrez un avertissement sur votre page d'administration. **Un système de mémoire cache n'est pas obligatoire et vous pouvez ignorer sans risque cet avertissement si vous le voulez**.

Note

Si vous activez un cache distribué dans votre fichier `config.php` (`memcache.distributed`) au lieu d'un cache local (`memcache.local`), vous verrez encore cet avertissement.

Un opcache PHP stocke les scripts PHP compilés de sorte à ce qu'ils ne soient pas recompilés chaque fois qu'ils sont appelés. PHP fournit l'OPcache Zend dans son code de base depuis la version 5.5, de sorte que vous n'aurez pas à installer un opcache pour PHP 5.5 et versions suivantes.

Si vous utiliser PHP 5.4, qui est la plus ancienne version PHP gérée par ownCloud, vous pouvez installer Alternative PHP Cache (APC). C'est à la fois un opcache et un cache de données. APC n'a pas été mis à jour depuis 2012 et n'est plus maintenu, et PHP 5.4 est ancien et se traîne par rapport aux dernières versions. Si cela est possible, il est préférable d'utiliser la dernière version de PHP.

Le cache de données est fourni par Alternative PHP Cache, user (APCu) dans PHP 5.5+, Memcached ou Redis.

ownCloud peut gérer plusieurs services de cache mémoire, vous pouvez donc choisir celui qui convient le mieux à vos besoins. Les services de cache gérés sont les suivants :

- **APC**
  Un cache local pour les systèmes utilisant PHP 5.4.
- **APCu, APCu 4.0.6 ou supérieur est nécessaire.**
  Un cache local pour les systèmes utilisant PHP 5.5 et les versions suivantes.
- **Memcached**
  Un système de cache distribué pour plusieurs installations d'ownCloud.
- **Redis, Le module PHP 2.2.6 ou supérieur est nécessaire.**
  Un système de cache distribué.

Les systèmes de mémoire cache doivent être explicitement configurés à partir de la version 8.1 d'ownCloud. Vous devez installer et activer le système de cache désiré, puis ajouter l'entrée appropriée dans le fichier `config.php` (voir *Paramètres de Config.php* pour un aperçu de tous les paramètres de configuration possibles).

Vous pouvez utiliser à la fois un cache local et un cache distribué. Les systèmes de cache recommandés sont APCu et Redis. Après avoir installé et activé le système de cache désiré, vérifiez qu'il est actif en exécutant *Informations et version de PHP*.

APC

APC is only for systems running PHP 5.4 and older. The oldest supported PHP version in ownCloud is 5.4.

Note

RHEL 6 and CentOS 6 ship with PHP 5.3 and must be upgraded to PHP 5.4 to run ownCloud. See *Installation de PHP 5.4 sous RHEL 6 et CentOS 6*.

On Red Hat/CentOS/Fedora systems running PHP 5.4, install `php-pecl-apc`. On Debian/Ubuntu/Mint systems install `php-apc`. Then restart your Web server.

After restarting your Web server, add this line to your `config.php` file:

```
'memcache.local' => '\OC\Memcache\APC',
```

Refresh your ownCloud admin page, and the cache warning should disappear.

APCu

PHP 5.5 et supérieur contiennent Zend OPcache dans leur code de base, et sur la plupart des distributions Linux, il est activé par défaut. Cependant, il ne fournit pas de cache de données. APCu est un cache de données qui est disponible dans la plupart des distributions Linux. Pour les systèmes Red Hat/CentOS/Fedora utilisant PHP 5.5 et supérieur, installez le paquet `php-pecl-apcu`. Pour les systèmes Debian/Ubuntu/Mint installez `php5-apcu`. Sous Ubuntu 14.04LTS, la version de APCu est 4.0.2, qui est trop ancienne pour être utilisée avec ownCloud. ownCloud nécessite au minimum la version 4.0.6. Vous pouvez installer la version 4.0.7 à partir des rétro-portages d'Ubuntu avec cette commande:

```
apt-get install php5-apcu/trusty-backports
```

Puis, redémarrez votre serveur Web.

Après le redémarrage de votre serveur Web, ajoutez la ligne suivante dans le fichier `config.php`:

```
'memcache.local' => '\OC\Memcache\APCu',
```

Rechergez la page d'administration d'ownCloud : le message d'avertissement a disparu.

Memcached

Memcached est un vénérable système de cache pour les serveurs distribués et et fonctionne bien avec ownCloud à une exception près : il ne peut pas être utilisé avec le *verrouillage de fichier transactionnel* car il ne stocke pas les verrous, et les données peuvent disparaître du cache à tout moment (Redis est le meilleur système de cache pour cela).

Note

Assurez-vous d'installer le module PHP **memcached** et non memcache. ownCloud ne sait gèrer que le module PHP **memcached**.

Le paramétrage de Memcached est facile. Sous Debian/Ubuntu/Mint installez `memcached` et `php5-memcached`. L'installeur démarrera automatiquement `memcached` et le configurera pour être lancé au démarrage.

Sous Red Hat/CentOS/Fedora installez `memcached` et `php-pecl-memcached`. Il ne sera pas démarré automatiquement, vous devrez donc utiliser le gestionnaire de services pour démarrer `memcached` et pour le lancer au démarrage en tant que démon.

Vous pouvez vérifier que Memcached est en cours d'exécution avec la commande `ps ax`:

```
ps ax | grep memcached
19563 ? Sl 0:02 /usr/bin/memcached -m 64 -p 11211 -u memcache -l
127.0.0.1
```

Redémarrez votre serveur Web et ajoutez les entrées appropriées dans le fichier `config.php`, puis recharger la page d'administration d'ownCloud. L'exemple suivant utilise APCu comme cache local, Memcached comme cache distribué et liste tous les serveurs du pool partagé avec leur numéro de port:

```
'memcache.local' => '\OC\Memcache\APCu',
'memcache.distributed' => '\OC\Memcache\Memcached',
'memcached_servers' => array(
     array('localhost', 11211),
     array('server1.example.com', 11211),
     array('server2.example.com', 11211),
     ),
```

Redis

Redis est un excellent système de cache moderne à utiliser pour le cache distribué et le cache local pour le *verrouillage de fichier transactionnel* car il garantit que les objets sont disponibles aussi longtemps que nécessaire.

Le module PHP Redis doit être au minimum en version 2.2.6. Si vous utilisez une distribution Linux qui ne fournit pas de versions gérées de ce module ou qui ne fournit pas du tout Redis, veuillez consulter *Aide supplémentaire pour l'installation de Redis*.

Sous Debian/Ubuntu/Mint installez `redis-server` et `php5-redis`. L'installeur lancera automatiquement `redis-server` et le configurera pour être lancé au démarrage.

Sous CentOS et Fedora installez `redis` et `php-pecl-redis`. Il ne sera pas démarré automatiquement, vous devrez donc utiliser le gestionnaire de services pour démarrer `redis`, et pour le lancer au démarrage en tant que démon.

Vous pouvez vérifier que Memcached est en cours d'exécution avec la commande `ps ax`:

```
ps ax | grep redis
22203 ? Ssl    0:00 /usr/bin/redis-server 127.0.0.1:6379
```

Redémarrez votre serveur Web et ajoutez les entrées appropriées dans le fichier `config.php`, puis recharger la page d'administration d'ownCloud. L'exemple suivant utilise Redis comme cache local:

```
'memcache.local' => '\OC\Memcache\Redis',
'redis' => array(
     'host' => 'localhost',
     'port' => 6379,
      ),
```

Pour de meilleures performances, utilisez Redis pour le verrouillage de fichiers en ejoutant ceci:

```
'memcache.locking' => '\OC\Memcache\Redis',
```

Si vous voulez vvous connecter à Redis configuré pour écouter sur un socket Unix (ce qui est recommandé si Redis fonctionne sur le même serveur qu'ownCloud), utilisez cet exemple de configuration de `config.php`:

```
'memcache.local' => '\OC\Memcache\Redis',
'redis' => array(
     'host' => '/var/run/redis/redis.sock',
     'port' => 0,
      ),
```

Redis est très paramétrable ; consultez la documentation de Redis pour en apprendre plus.

Emplacement du répertoire de cache

Par défaut, le répertoire de cache se situe dans `data/$user/cache` où `$user` est l'utilisateur courant. Vous pouvez utiliser la directive `'cache_path'` dans le fichier `config.php` (voir *Paramètres de Config.php*) pour sélectionner un emplacement différent.

Recommandations en fonction du type de déploiement

Serveur domestique ou de petite taille

Utilisez uniquement APCu:

```
'memcache.local' => '\OC\Memcache\APCu',
```

Petite organisation, un seul serveur

Utilisez APCu pour le cache local et Redis pour le verrouillage de fichiers:

```
'memcache.local' => '\OC\Memcache\APCu',
'memcache.locking' => '\OC\Memcache\Redis',
 'redis' => array(
      'host' => 'localhost',
      'port' => 6379,
       ),
```

Grande organisation, cluster de serveurs

Utilisez Redis pour tout sauf le cache local. Utilisez l'adresse IP ou le nom d'hôte du serveur pour qu'il soit accessible pour les autres hôtes

```
'memcache.distributed' => '\OC\Memcache\Redis',
'memcache.locking' => '\OC\Memcache\Redis',
'memcache.local' => '\OC\Memcache\APCu',
'redis' => array(
     'host' => 'server1',  //exemple de nom d'hôte
     'host' => '12.34.56.78',  //exemple d'adresse IP
     'port' => 6379,
      ),
```

Aide supplémentaire pour l'installation de Redis

Si vous utilisez une version de Mint ou d'Ubuntu qui ne fournit pas la version requise de `php5-redis`, essayez alors this Redis guide on Tech and Me polur une installation complète de Redis sous Ubuntu 14.04 en utilisant PECL. Ces instructions sont adaptables pour toute distribution qui ne fournit pas de paquet ayant la version requise ou qui ne founit pas du tout Redis, comme SUSE Linux Enterprise Server ou Red Hat Enterprise Linux.

Le module PHP Redis doit être au minimum en version 2.2.6.

Consulter https://pecl.php.net/package/redis

Sous Debian/Mint/Ubuntu, utilisez `apt-cache` pour voir la version de `php5-redis` disponible ou la version du paquet installé:

```
apt-cache policy php5-redis
```

Sous CentOS et Fedora, la commande `yum` affiche les informations de version disponible et installée:

```
yum search php-pecl-redis
```

Tâches d'arrière-plan

Un système comme ownCloud nécessite parfois que des tâches soient réalisées régulièrement sans l'intervention de l'utilisateur et sans affecter les performances d'ownCloud. À cette fin, en tant qu'administrateur système, vous

pouvez définir des tâches d'arrière-plan (par exemple le nettoyage de la base de données) qui seront exécutées sans intervention de l'utilisateur.

Ces tâches sont appelées *tâches cron*. Les tâches cron sont des commandes ou des scripts shell programmés pour s'exécuter périodiquement à des heures, dates ou intervalles définis. `cron.php` est un processus interne d'ownCloud qui exécute ce type de tâches d'arrière-plan à la demande.

Les plugins applicatifs d'ownCloud enregistrent des actions dans `cron.php` automatiquement pour gérer les opérations de maintenance classique comme la suppression de fichiers temporaires ou la vérification de nouvelles mises à jour de fichiers en utilisant `filescan()` pour les montages de systèmes de fichiers externes.

Paramètres

Dans la page d'administration, vous pouvez configurer comment ces tâches seront exécutées. Vous avez le choix entre les options suivantes :

- AJAX
- Webcron
- Cron

Tâches Cron

Vous pouvez programmer les tâches cron de trois manières -- en utilisant AJAX, Webcron ou cron. La méthode par défaut est d'utiliser AJAX. Cependant, la méthode recommandée est d'utiliser cron. Les sections suivantes décrivent les différences entre chaque méthode.

AJAX

La méthode de programmation AJAX est l'option par défaut. Malheureusement, c'est aussi la moins fiable. Chaque fois qu'un utilisateur visite la page ownCloud, une tâche d'arrière-plan est exécutée. L'avantage de ce mécanisme est que cela ne nécessite pas d'accéder au système ni de s'enregistrer sur un service tiers. L'inconvénient de ce mécanisme, comparé au service Webcron, est qu'il nécessite de visiter régulièrement la page pour être déclenché.

> **Note**
>
> Particulièrement lors de l'utilisation de l'application Activité ou de stockages externes, où de nouveaux fichiers sont ajoutés, mis à jour ou supprimés, il est préférable d'utiliser l'une des deux méthodes suivantes.

Webcron

En enregistrant l'adresse de votre script `cron.php` owncloud sur un service de webcron externe (par exemple easyCron), vous vous assurez que les tâches d'arrière-plan soient exécutées régulièrement. Pour utiliser ce type de service, votre serveur doit être accessible d'Internet. Par exemple:

```
URL à appeler : http[s]://<domaine-de-votre-serveur>/owncloud/cron.php
```

Cron

L'utilisation de la fonctionnalité cron du système d'exploitation est la méthode préférée pour l'exécution de tâches régulières. Cette méthode active l'exécution des tâches d'arrière-plan sans les limitations inhérentes au serveur Web.

Pour exécuter une tâche cron sur un système de type Unix, toutes les 15 minutes, avec le compte par défaut du serveur Web (souvent `www-data` ou `wwwrun`), vous devez paramétrer la tâche cron suivante pour appeler le script **cron.php**:

```
# crontab -u www-data -e
*/15  *  *  *  * php -f /var/www/owncloud/cron.php
```

Vous pouvez vérifier que la tâche cron a bien été ajoutée et programmée en exécutant la commande suivante:

```
# crontab -u www-data -l
*/15  *  *  *  * php -f /var/www/owncloud/cron.php
```

Note

Vous devez remplacer le chemin `/var/www/owncloud/cron.php` par celui de votre installation ownCloud.

Note

Vous devez vous assurer que `php` soit trouvé par `cron`. La bonne pratique est d'ajouter le chemin absolu, comme par exemple `/usr/bin/php`.

Note

Sur certains systèmes, il pourrait être nécessaire d'appeler **php-cli** au lieu de **php**.

Tâches d'arrière-plan disponibles

Il existe un certain nombre de tâches disponibles pour être exécutées pour des travaux spécifiques.

Note

Ces tâches ne sont généralement nécessaires que sur de grosses instances et peuvent être exécutées en arrière-plan. si le nombre d'utilisateurs de votre installation se situe entre 1000 et 3000, ou si vous utilisez LDAP et que cela devient un goulet d'étranglement, les administrateurs peuvent supprimer plusieurs entrées dans la table *oc_jobs* et les remplacer par les commandes *occ* correspondantes, comme indiqué ci-dessous :

- *OCA\Files_Trashbin\BackgroundJob\ExpireTrash -> occ trashbin:expire*
- *OCA\Files_Versions\BackgroundJob\ExpireVersions -> occ versions:expire*
- *OCA\DAVCardDAV\SyncJob -> occ dav:sync-system-addressbook*
- *OCA\Federation\SyncJob -> occ federation:sync-addressbooks*

Si ceci est utilisé, cela doit l'être sur une base quotidienne.

La liste qui suit n'est pas exhaustive :

ExpireTrash

La tâche ExpireTrash, contenue dans `OCA\Files_Trashbin\BackgroundJob\ExpireTrash`, supprimera tout fichier de la corbeille d'ownCloud plus âgé que le temps maximal de rétention spécifié. Elle peut être lancée en utilisant la commande OCC

```
occ trashbin:expire
```

ExpireVersions

La tâche ExpireVersions, contenue dans `OCA\Files_Versions\BackgroundJob\ExpireVersions`, fera expirer les versions de fichiers plus âgées que le temps maximal de rétention spécifié. Elle peut être lancée en utilisant la commande OCC

```
occ versions:expire
```

SyncJob (CardDAV)

La tâche SyncJob, contenue dans `OCA\DAV\CardDAV\SyncJob`, synchronise le carnet d'adresses local en mettant à jour les contacts existants et en supprimant ceux périmés. Elle peut être lancée en utilisant la commande OCC

```
occ dav:sync-system-addressbook
```

SyncJob (Fédération)

OCAFederationSyncJob

Elle peut être lancée en utilisant la commande OCC

```
occ federation:sync-addressbooks
```

Paramètres de Config.php

ownCloud utilise le fichier `config/config.php` pour contrôler les opérations du serveur. `config/config.sample.php` recense tous les paramètres de configuration d'ownCloud, avec un exemple ou les valeurs par défaut. Ce document fournit une référence plus détaillée. La plupart des options sont configurables sur la page d'administation. Il n'est donc souvent pas nécessaire de modifier le fichier `config/config.php`.

Note

L'installeur crée une configuration contenant les paramètres essentiels. N'ajoutez manuellement au fichier `config/config.php` que les paramètres pour lesquels vous avez besoin d'une valeur spéciale. **Ne copier pas tout le contenu de** `config/config.sample.php`. **Ne copier que les paramètres que vous souhaitez modifier !**

ownCloud gère le chargement des paramètres de configuration à partir de plusieurs fichiers. Vous pouvez ajouter des fichiers dont le nom se termine par .config.php dans le répertoire config/. Par exemple, vous pouvez placer votre configuration du serveur de messagerie dans un fichier email.config.php. Ceci permet de gérer plus facilement les configurations personnalisées ou de diviser un long fichier de configuration en un jeu de fichiers plus petits. Ces fichiers personnalisés ne sont pas écrasés par ownCloud et les valeurs dans ces fichiers ont la priorité sur ceux de config.php.

Paramètres par défaut

Ces paramètres sont configurés par l'installeur d'ownCloud et sont nécessaires pour que le serveur ownCloud fonctionne.

```
'instanceid' => '',
```

C'est un identifiant unique pour votre installation d'ownCloud, créé automatiquement par l'installeur. Cet exemple est seulement destiné à illustrer la documentation, et vous ne devez pas l'utiliser car il ne fonctionnera pas. Une valeur d'`instanceid` valide est créée quand vous installez ownCloud.

'instanceid' => 'd3c944a9a',

```
'passwordsalt' => '',
```

Le « sel » utilisé pour créer tous les mots de passe, généré automatiquement par l'installeur d'ownCloud. (Il existe aussi des « sels » par utilisateur). Si vous le perdez, vous perdez tous les mots de passe. Cet exemple est seulement destiné à illustrer la documentation et vous ne devez pas l'utiliser.

'passwordsalt' => 'YjA1Sfvjc3+ZDrVTL/upqDAuZIcohN',

```
'trusted_domains' =>
  array (
    'demo.exemple.org',
    'autredomaine.exemple.org',
  ),
```

La liste des domaines de confiance sur lesquels vos utilisateurs peuvent se connecter. Spécifier des domaines de confiance empêche l'empoisonnement des en-têtes d'hôte. Ne supprimer pas ceci car il réalise des vérifications de sécurité obligatoires.

```
'datadirectory' => '/var/www/owncloud/data',
```

Le répertoire où sont stockés les fichiers des utilisateurs. Par défaut, c'est le répertoire `data/` qui se situe dans le répertoire d'ownCloud. La base de données SQLite est également stockée ici quand vous utilisez SQLite.

(SQLite n'est pas disponible pour l'Édition Entreprise)

```
'version' => '',
```

Le numéro de version actuel de votre installation ownCloud. Ceci est défini pendant et la mise à jour. Vous ne devez jamaiis le modifier.

```
'dbtype' => 'sqlite',
```

Identifie le type de base de données utilisée pour cette installation. Consulter également l'option de configuration `supportedDatabases`.

Disponible :

- sqlite (SQLite3 - pas dans l'Édition Entreprise)
- mysql (MySQL/MariaDB)
- pgsql (PostgreSQL)
- oci (Oracle - Seulement dans l'Édition Entreprise)

```
'dbhost' => '',
```

Le nom d'hôte du serveur, par exemple : `localhost`, `nom_hote`, `nom_hote.exemple.com` ou l'adresse IP. Pour spécifier un port à utiliser : `nom_hote:####`. Pour spécifier un socket Unix : `localhost:/chemin/vers/socket`.

```
'dbname' => 'owncloud',
```

Le nom de la base de données d'ownCloud qui est définie pendant l'installation. Vous ne devez pas le modifier.

```
'dbuser' => '',
```

Le nom d'utilisateur qu'utilisera ownCloud pour écrire dans la base de données. Il doit être unique pour toutes les instances utilisant la même base de données. Ceci est défini pendant l'installation. Vous ne devez pas le modifier.

```
'dbpassword' => '',
```

Le mot de passe pour l'utilisateur de la base de données. Ceci est défini pendant l'installation. Vous ne devez pas le modifier.

```
'dbtableprefix' => '',
```

Le préfixe utilisé pour les tables ownCloud dans la base de données.

```
'installed' => false,
```

Indique si l'installation de l'instance ownCloud s'est bien terminée. `true` indique une installation réussie et `false` une installation qui a échoué.

Exemples de config.php

Quand vous utilisez SQLite comme base de données, votre fichier `config.php` ressemble à ceci après l'installation. La base de données SQLite est stockée dans le répertoire `data/` dans votre répertoire ownCloud. SQLite est un système de gestion de base de donnée léger pratique pour les tests et les très petites installations. Pour les installations de production, vous devez utiliser MySQL, MariaDB ou PosgreSQL.

```
<?php
$CONFIG = array (
  'instanceid' => 'occ6f7365735',
  'passwordsalt' => '2c57784763467863063',
```

```
  'trusted_domains' =>
  array (
    0 => 'localhost',
    1 => 'studio',
  ),
  'datadirectory' => '/var/www/owncloud/data',
  'dbtype' => 'sqlite3',
  'version' => '7.0.2.1',
  'installed' => true,
);
```

Voici un exemple pour une nouvelle installation d'ownCloud utilisant MariaDB:

```
<?php
$CONFIG = array (
  'instanceid' => 'oc8c0fd71e03',
  'passwordsalt' => '515a13302a6b3950a9d0fdb970191a',
  'trusted_domains' =>
  array (
    0 => 'localhost',
    1 => 'studio',
    2 => '192.168.10.155'
  ),
  'datadirectory' => '/var/www/owncloud/data',
  'dbtype' => 'mysql',
   'version' => '7.0.2.1',
  'dbname' => 'owncloud',
  'dbhost' => 'localhost',
  'dbtableprefix' => 'oc_',
  'dbuser' => 'oc_carla',
  'dbpassword' => '67336bcdf7630dd80b2b81a413d07',
  'installed' => true,
);
```

Expérience utilisateur

Ces paramètres facultatifs contrôlent certains aspects de l'interface utilisateur. Les valeurs par défaut, quand il en existe, sont indiquées.

```
'default_language' => 'en',
```

Ceci définit la langue par défaut de votre serveur ownCloud, utilisant les codes de langues ISO_639-1, comme en pour l'anglais, de pour l'allemand et `fr` pour le français. Il a priorité sur la détection de langue automatique sur les pages publiques comme celle de connexion et les éléments partagés. La préférence de langue utilisateur configurée dans la page personnelle a la priorité sur ce paramètre après la connexion de l'utilisateur.

```
'defaultapp' => 'files',
```

Définit l'application ouverte par défaut après la connexion. Utilisez le nom de l'application tel qu'il apparaît dans l'URL après avoir cliqué sur celle-ci dans le menu Applications, comme « documents », « calendar » et « gallery » par exemple. Vous pouvez utiliser une liste de noms d'applications séparés par des virgules, de sorte que si la première application n'est pas activée, ownCloud essaiera la seconde, et aisni de suite. Si aucune application n'est trouvée, l'application Fichiers sera affichée par défaut.

```
'knowledgebaseenabled' => true,
```

`true` active l'élément de menu Aide dans le menu utilisateur (dans le coin supérieur droit de l'interface Web d'ownCloud). `false` retire l'élément Aide.

```
'enable_avatars' => true,
```

`true` active les avatars ou les photos des profils utilisateurs. Ils apparaissent dans la page Utilisateurs, sur la page personnelle et sont utilisés par certaines applications (Contacts, Mail, etc.). `false` les désactive.

```
'allow_user_to_change_display_name' => true,
```

`true` permet à l'utilisateur de modifier son nom d'affichage (dans la page personnelle), et `false` empêche de modifier le nom d'affichage.

```
'remember_login_cookie_lifetime' => 60*60*24*15,
```

Durée de vie du cookie de connexion, qui est défini lorsque l'utilisateur coche la case « Rester connecté » sur l'écran de connexion. Elle est par défaut de 15 jours, exprimée en secondes.

```
'session_lifetime' => 60 * 60 * 24,
```

Durée de vie d'une session après inactivité. Elle est par défaut de 24 heures, exprimée en secondes.

```
'session_keepalive' => true,
```

Active ou désactive le maitien d'établissement de session quand un utilisateur se connecte sur l'interface Web.

Son activation envoir une « pulsation » au serveur pour empêcher la fin de session.

```
'token_auth_enforced' => false,
```

Met en œuvre des jetons d'authentification pour les clients, qui bloquent les requêtes utilisant le mot de passe utilisateur, pour une sécurité renforcée. Les utilisateurs doivent générer les jetons sur leur page personnelle qui peuvent être utilisés comme mot de passe sur leur client.

```
'token_auth_enforced' => false,
```

Met en œuvre l'authentication par jeton pour les clients, ce qui bloque les requêtes utilisant le mot de passe utilisateur, pour une sécurité renforcée. Les utilisateurs Les utilisateurs doivent générer les jetons dans la page des paramètres personnels qui seront alors utilisés comme mot de passe pour leurs clients.

```
'skeletondirectory' => '/path/to/owncloud/core/skeleton',
```

Le répertoire où sont situés les fichiers « squelette ». Ces fichiers seront copiés dans le répertoire data des nouveaux utilisateurs. Laisser la valeur de ce paramètre vide pour ne copier aucun fichier.

```
'user_backends' => array(
        array(
                'class' => 'OC_User_IMAP',
                'arguments' => array('{imap.gmail.com:993/imap/ssl}INBOX')
        )
),
```

L'application `user_backends` (qui doit être activée en premier) permet de configurer les services alternatifs d'authentification. Les services gérés sont : IMAP (OC_User_IMAP), SMB (OC_User_SMB) et FTP (OC_User_FTP).

```
'lost_password_link' => 'https://example.org/link/to/password/reset',
```

Quand le service ne permet pas de réinitialiser le mot de passe (par exemple quand il s'agit d'un service en lecture seule comme LDAP), vous pouvez spécifier un lien personnalisé sur lequel sera redirigé l'utilisateur en cliquant sur le lien « Mot de passe incorrect. Réinitialiser ? » après un échec de connexion.

Paramètre de messagerie

Ceci configure les paramètres de messagerie pour les notifications d'ownCloud et les réinitialisations de mot de passe.

```
'mail_domain' => 'exemple.com',
```

L'adresse du domaine de retour que vous voulez voir apparaître sur les courriels envoyés par le serveur ownCloud, par exemple : `admin-owncloud@exemple.com`, et qui se substitue à votre nom de domaine.

```
'mail_from_address' => 'owncloud',
```

L'adresse d'expéditeur qui remplace les adresses par défaut `sharing-noreply` et `lostpassword-noreply`.

```
'mail_smtpdebug' => false,
```

Active la classe de débogage de SMTP.

```
'mail_smtpmode' => 'sendmail',
```

Le mose à utiliser pour envoyer des courriels : `sendmail`, `smtp`, `qmail` ou `php`.

Pour l'utilisation d'un serveur SMTP local ou distant, définir ceci à `smtp`.

Pour l'utilisation de PHP mail, un serveur de messagerie doit être installé et opérationnel sur le serveur. Le programme utilisé pour envoyer les courriels est défini dans le fichier `php.ini`.

Pour l'option `sendmail`, un serveur de messagerie doit être installé et opérationnel sur le serveur, avec `/usr/sbin/sendmail` installé sur le serveur de type Unix.

Pour `qmail`, le binaire est `/var/qmail/bin/sendmail` et doit être installé sur le serveur de type Unix.

```
'mail_smtphost' => '127.0.0.1',
```

Ceci dépend de `mail_smtpmode`. Spécifier l'adresse IP du serveur de messagerie. Cette valeur peut être composée de plusieurs hôtes séprarés par des point-virgules. S'il est nécessaire d'indiquer le numéro de port, il faut l'ajouter après l'adresse IP en le séprarant par un double-point, comme ceci : `127.0.0.1:24`.

```
'mail_smtpport' => 25,
```

Ceci dépend de `mail_smtpmode`. Spécifier le port pour l'envoi des courriels.

```
'mail_smtptimeout' => 10,
```

Ceci dépend de `mail_smtpmode`. Il définit la temporisation du serveur SMTP, en secondes. Il peut être nécessaire d'augmenter cette valeur si un programme d'analyse de logiciels malveillants ou de courriers indésirables est exécuté sur le serveur.

```
'mail_smtpsecure' => '',
```

Ceci dépend de `mail_smtpmode`. Définit l'utilisation du chiffrement `ssl` ou `tls`. Laisser vide si aucun chiffrement n'est utilisé.

```
'mail_smtpauth' => false,
```

Ceci dépend de `mail_smtpmode`. Changer ceci pour `true` si le serveur de messagerie nécessite une authentification.

```
'mail_smtpauthtype' => 'LOGIN',
```

Dépend de `mail_smtpmode`. Si l'authentification SMTP est requise, choisir le type d'authentification `LOGIN` (par défaut) ou `PLAIN`.

```
'mail_smtpname' => '',
```

Ceci dépend de `mail_smtpauth`. Indiquer le nom d'utilisateur pour l'authentification sur le serveur SMTP.

```
'mail_smtppassword' => '',
```

Ceci dépend de `mail_smtpauth`. Indiquer le mot de passe pour l'authentification sur le serveur SMTP.

Configurations de proxy

```
'overwritehost' => '',
```

La détection automatique de nom d'hôte d'ownCloud peut échouer dans certaines situations. Cette option permet de passer outre la détection automatique. Par exemple : `www.exemple.com` ou avec un numéro de port : `www.exemple.com:8080`.

```
'overwriteprotocol' => '',
```

Lors de la génération des URL, ownCloud essaie de déterminer si le serveur est accédé via `https` ou `http`. Cependant, si ownCloud se trouve derrière un proxy et que le proxy gère les appels `https`, ownCloud ne saura pas si `ssl` est utilisé t les URL générées seront incorrectes.

Les valeurs valides sont `http` et `https`.

```
'overwritewebroot' => '',
```

ownCloud essaie de déterminer la racine Web pour la génération automatique des URL.

Par exemple, si `www.exemple.com/owncloud` est l'URL pointant sur l'instance ownCloud, la racine Web est `/owncloud`. Quand des proxies sont utilisés, il peut être difficile pour ownCloud de détecter ce paramètre et les URL générées sont alors incorrectes.

```
'overwritecondaddr' => '',
```

Cette option permet de définir manuellement une condition de remplacement de l'adresse IP distante à l'aide d'une expression régulière. Par exemple, la définition d'une étendue d'adresses IP débutant par `10.0.0.` et se terminant par 1 ou 3 : `^10\.0\.0\.[1-3]$`

```
'overwrite.cli.url' => '',
```

Utiliser ce paramètre de configuration pour spécifier l'URL de base pour toutes les URL générées dans ownCloud utilisant toutes sortes d'outils de ligne de commande (cron ou occ). La valeur doit contenir l'URL de base complète : `https://www.exemple.com/owncloud`

```
'htaccess.RewriteBase' => '/',
```

Pour avoir des URL propres, sans */index.php*, ce paramètre doit être configuré.

Ce paramètre sera écrit comme « RewriteBase » lors des mises à jour ou de l'installation d'ownCloud dans le fichier *.htaccess*. Bien que souvent cette valeur soit simplement l'URL de l'installation d'ownCloud, elle ne peut être définie automatiquement sans erreur dans tous les scénarios possibles, et doit donc être définie manuellement.

Dans une installation Apache standard, c'est généralement le dossier dans lequel est accessible ownCloud. Docn, si ownCloud est accessible via « https://moncloud.org/owncloud », la valeur correcte sera très vraisemblablement « /owncloud ». Si ownCloud est accessible via « https://moncloud.org/ » alors ce sera « / ».

Veuillez noter que la règle ci-dessus n'est pas valide dans tous les cas. Dans certains rares cas, cette règle ne peut pas s'appliquer. Cependant, pour éviter tout problème lors des mises à jour, cette valeur doit être explicitement modifiée.

Après avoir modifié cette valeur, il faut exécuter la commande *occ maintenance:update:htaccess* et quand les deux conditions suivantes sont remplies, ownCloud utilise les URL sans index.php :

- *mod_rewrite* est installé
- *mod_env* est installé

```
'proxy' => '',
```

L'URL du serveur proxy, par exemple : `proxy.exemple.com:8081`.

```
'proxyuserpwd' => '',
```

Si c'est nécessaire, l'authentification à utiliser pour le proxy pour se connecter à internet.

Le format est le suivant : `utilisateur:mot_de_passe`.

Éléments supprimés (corbeille)

Ces paramètres contrôlent l'application Deleted Files.

```
'trashbin_retention_obligation' => 'auto',
```

Si la corbeille est activée (par défaut), ce paramètre définit la politique de suppression défnitive des fichiers et des dossiers dans la corbeille.

L'application autorise deux réglages : les délais minimum et maximum de rétention dans la corbeille. Le délai minimum est le nombre de jours pendant lequel un fichier est conservé, après quoi il peut être supprimé. Le délai maximum est le nombre de jours après lesquels il est garanti que l'élément est supprimé. Ces deux délais peuvent être définis ensemble pour définir explicitement la suppression des fichiers et dossiers. Pour des raisons de migration, ce paramètre est initialement défini à « auto », ce qui est le paramètre par défaut dans ownCloud 8.1 et les versions précédentes.

Valeurs disponibles :

- `auto`

  (par défaut) Conserve les fichiers et dossiers dans la corbeille pendant 30 jours et les supprime automatiquement après ce délai si de l'espace disque est nécessaire (note : les fichiers peuvent ne pas être supprimés s'il n'y a pas besoin d'espace).

- `D, auto`

  Conserve les fichiers et dossiers dans la corbeille pendant D jours et les supprime ensuite si de l'espace disque est nécessaire (note : les fichiers peuvent ne pas être supprimés s'il n'y a pas besoin d'espace).

- `auto, D`

  Supprime tous les fichiers de la corbeille âgés de plus de D jours automatiquement, supprime les autres fichiers si de l'espace disque est nécessaire.

- `D1, D2`

  Conserve les fichiers dans la corbeille pendant au moins D1 jours et les supprime après D2 jours.

- `disabled`

  Les fichiers sont conservés pour une durée illimitée.

Versions de fichiers

Ces paramètres contrôlent l'application Versions.

```
'versions_retention_obligation' => 'auto',
```

Si l'application Versions est activée (par défaut), ce paramètre définit la politique de conservation des versions de fichiers.

L'application autorise deux réglages : les délais minimum et maximum de rétention. Le délai minimum est le nombre de jours pendant lequel un fichier est conservé, après quoi il peut être supprimé. Le délai maximum est le nombre de jours après lesquels il est garanti que l'élément est supprimé. Ces deux délais peuvent être définis ensemble pour définir explicitement la suppression des fichiers. Pour des raisons de migration, ce paramètre est initialement défini à « auto », ce qui est le paramètre par défaut dans ownCloud 8.1 et les versions précédentes.

Valeurs disponibles :

- `auto`

  (par défaut) Supprime automatiquement les versions en fonction des règles d'expiration. Veuillez consulter *Contrôle de version et rétention de fichiers* pour plus d'informations.

- `D, auto`

  Conserve les versions de fichiers pendant D jours et les supprime ensuite si de l'espace disque est nécessaire (note : les versions peuvent ne pas être supprimées s'il n'y a pas besoin d'espace).

- `auto, D`

  Supprime toutes les versions de fichiers âgées de plus de D jours automatiquement, supprime les autres versions si de l'espace disque est nécessaire.

- `D1, D2`

  Conserve les versions au moins D1 jours et les supprime après D2 jours.

- `disabled`

  Les versions sont conservées pour une durée illimitée.

Vérifications ownCloud

ownCloud réalise plusieurs vérifications. Il existe deux options, `true` et `false`.

```
'appcodechecker' => true,
```

Vérifie avant son installation si une application utilise une API privée au lieu des API publiques. Si ceci est défini à « true », il n'autorisera l'installation ou l'activation des applications que si elles ont passé avec succès cette vérification.

```
'updatechecker' => true,
```

Vérifie si ownCloud est à jour et affiche une notification si une nouvelle version est disponible.

```
'updater.server.url' => 'https://updates.owncloud.com/server/',
```

URL que doit utiliser ownCloud pour rechercher les mises à jour.

```
'has_internet_connection' => true,
```

Indique si ownCloud est connecté à Internet ou limité au réseau local.

```
'check_for_working_webdav' => true,
```

Permet à ownCloud de vérifier si une connexion WebDAV est opérationnelle. Ceci réalisé en essayant de faire une requête WebDAV avec PHP.

```
'check_for_working_wellknown_setup' => true,
```

Permet à ownCloud de vérifier si les URL de redirection « .well-known » sont opérationnelles. Ceci est réalisé en essayant de faire une requête JavaScript sur https://votre-domaine.com/.well-known/caldav/

```
'check_for_working_htaccess' => true,
```

Ceci est un point de sécurité crucial pour les serveurs Apache et qui doit toujours être défini à `true`. Ceci vérifie que le fichier `.htaccess` est accessible en écriture et fonctionne.

Si ce n'est pas le cas, alors toutes les options contrôlées par le fichier `.htaccess`, tel que le téléversement de gros fichiers, ne fonctionneront pas. Il vérifie également si le répertoire `data/` peut être accédé directement par le serveur Web.

```
'config_is_read_only' => false,
```

Dans certains environnement, il peut être souhaité d'avoir le fichier de configuration accessible en lecture seulement.

Si ce paramètre est défini à``true``, ownCloud ne vérifiera pas si le fichier de configuration est accessible en écriture. Cependant, il ne sera pas possible de configurer toutes les options par l'interface Web. D'autre part, lors de la mise à jour d'ownCloud, il est nécessaire de rendre le fichier de configuration accessible en écriture pour que le processus de mise à jour puisse aboutir.

Journaux

```
'log_type' => 'owncloud',
```

Par défaut, les journaux d'ownCloud sont envoyés dans le fichier `owncloud.log` dans le répertoire `data/` par défaut d'ownCloud.

Si le fichier syslog du système est préféré, définir ce paramètre à `syslog`. Définir ce paramètre à `errorlog` utilisera la fonction PHP error_log pour les journaux.

```
'logfile' => '/var/log/owncloud.log',
```

Chemin d'accès au fichier journal d'ownCloud.

Par défaut : `[répertoire_data]/owncloud.log`

```
'loglevel' => 2,
```

Niveau de journalisation. Les valeurs valides sont : 0 = Débogage, 1 = Information, 2 = Avertissement, 3 = Erreur et 4 = Erreur fatale. La valeur par défaut est 2.

```
'syslog_tag' => 'ownCloud',
```

Si vous maintenez plusieurs instances et que vous agrégez les journaux, vous pourriez vouloir les distinguer. `syslog_tag` peut être défini pour chaque instance avec un identifiant unique. Seulement disponible si `log_type` est défini à `syslog`.

La valeur par défaut est `ownCloud`.

```
'log.condition' => [
        'shared_secret' => '57b58edb6637fe3059b3595cf9c41b9',
```

```
        'users' => ['exemple-user'],
        'apps' => ['files'],
],
```

Des conditions peuvent être ajoutées. Si l'une des conditions est remplie, le niveau de journalisation passe en débogage. Ceci permet de déboguer des requêtes, des utilisateurs ou des applications spécifiques.

Conditions gérées :

- `shared_secret`**: Si un paramètre de requête avec le nom** *log_secret* **est** défini, la condition est remplie.
- `users`**: Si la requête est faite par l'un des utilisateurs spécifiés, la** condition est remplie.
- `apps`**: Si le message provient des applications spécifiées, la condition** est remplie.

La valeur par défaut est un tableau vide.

```
'logdateformat' => 'F d, Y H:i:s',
```

Ceci utilise le format PHP.date. Consulter : http://php.net/manual/en/function.date.php

```
'logtimezone' => 'Europe/Berlin',
```

Le fuseau horaire par défaut pour les journaux est UTC. Vous pouvez modifier ceci. Consulter : http://php.net/manual/en/timezones.php

```
'log_query' => false,
```

Ajoute toutes les requêtes de base de données et les paramètres au journal. À n'utiliser que pour faire du débogage car le journal deviendra énorme.

```
'cron_log' => true,
```

Consigne les tâches cron qui se sont exécutées avec succès.

```
'log_rotate_size' => false,
```

Active la rotation des fichiers journaux et limite la taille totale de ceux-ci. La valeur par défaut est `0`, soit aucune rotation. Spécifier une taille en octets, par exemple : 104857600 (100 méga-octets = 100 * 1024 * 1024 octets). Un nouveau fichier avec un nouveau nom est créé lorsque le fichier journal atteint la limite indiquée.

Emplacements alternatifs des binaires

Ceci permet de définir de nouveaux emplacements pour télécharger les clients.

```
'customclient_desktop' =>
        'https://owncloud.org/install/#install-clients',
'customclient_android' =>
        'https://play.google.com/store/apps/details?id=com.owncloud.android',
'customclient_ios' =>
        'https://itunes.apple.com/us/app/owncloud/id543672169?mt=8',
```

Cette section permet de configurer les liens de téléchargement pour les clients ownCloud, tels qu'ils apparaissent sur l'écran de l'assistant de premier lancement sur la page personnelle.

Applications

Options pour le dossier des applications, le magasin des applications et me vérificateur de code des applications.

```
'appstoreenabled' => true,
```

Quand ceci est activé, les administrateurs peuvent installer les applications à partir du magasin d'applications d'ownCloud.

```
'appstoreurl' => 'https://api.owncloud.com/v1',
```

L'URL du magasin d'applications à utiliser.

```
'appstore.experimental.enabled' => false,
```

Indique s'il faut afficher les applications expérimentales dans l'interface du magasin d'appliations.

Les applications expérimentales ne pas reçu de contrôle de sécurité, ne sont pas stables ou sont encore en cours de développement. Les installer peut provoquer des pertes de données ou des failles de sécurité.

```
'apps_paths' => array(
        array(
                'path'=> '/var/www/owncloud/apps',
                'url' => '/apps',
                'writable' => true,
        ),
),
```

Utiliser le paramètre `apps_paths` pour définir l'emplacement du répertoire des applications, qui doit être accédé pour vérifier la disponibilité de nouvelles applications, et où les applications spécifiques aux utilisateurs doivent être installées à parti du magasin d'applications. La valeur de `path` est le chemin d'accès absolu du système de fichiers pour accéder au répertoire. La valeur de `url` définit le chemin d'accès HTTP pour ce répertoire, à partir de la racine Web d'ownCloud. La valeur de `writable` indique si le serveur Web peut écrire dans ce répertoire.

```
'appcodechecker' => true,
```

Contrôle une application avant de l'installer pour savoir si elle utilise des API privées au lieu des API publiques qui doivent être utilisées. Si défini à `true`, il n'autorisera l'installation ou l'activation que pour les applications ayant passé ce contrôle avec succès.

Aperçus

ownCloud gère la génération d'aperçus d'image, de couvertures de fichiers MP3 et de fichiers texte. Ces options contrôlent l'activation et la désactivation des aperçus, et la taille des vignettes.

```
'enable_previews' => true,
```

Par défaut, ownCloud peut générer des aperçus pour les types de fichiers suivants :

- les fichiers images ;
- les couvertures de fichiers MP3 ;
- les documents texte.

Les valeurs valides sont `true` pour activer les aperçus ou `false` pour les désactiver.

```
'preview_max_x' => 2048,
```

La largeur maximale en pixels d'un aperçu. La valeur `null` signifie qu'il n'y a pas de limite.

```
'preview_max_y' => 2048,
```

La hauteur maximale en pixels d'un aperçu. La valeur `null` signifie qu'il n'y a pas de limite.

```
'preview_max_scale_factor' => 10,
```

S'il y a beaucoup de petites images stockées dans ownCloud et que le système d'aperçus génère des vignettes floues, vous pourriez envisager de définir un facteur de mise à l'échelle maximal. Par défaut, un facteur 10 est appliqué sur les images. Les valeurs `1` ou `null` désactive la mise à l'échelle.

```
'preview_max_filesize_image' => 50,
```

Taille maximale du fichier pour générer les vignettes avec imagegd (comportement par défaut). Si l'image est plus grande, il essaiera un autre générateur d'aperçus, mais affichera très probablement l'icône par défaut.

La valeur représente la taille maximale en méga-octets. Par défaut, cette valeur est `50`. Définir cette valeur à `-1` pour ne pas établir de limite.

```
'preview_libreoffice_path' => '/usr/bin/libreoffice',
```

Chemin d'accès personnalisé pour accéder au binaire de LibreOffice ou OpenOffice.

```
'preview_office_cl_parameters' =>
        ' --headless --nologo --nofirststartwizard --invisible --norestore '.
        '--convert-to pdf --outdir ',
```

Utiliser ceci si LibreOffice ou OpenOffice nécessite des arguments supplémentaires.

```
'enabledPreviewProviders' => array(
        'OC\Preview\PNG',
        'OC\Preview\JPEG',
        'OC\Preview\GIF',
        'OC\Preview\BMP',
        'OC\Preview\XBitmap',
        'OC\Preview\MP3',
        'OC\Preview\TXT',
        'OC\Preview\MarkDown'
),
```

N'enregitre que les foounrisseurs qui ont été explicitement activés.

Les fournisseurs suivants sont activés par défaut :

- OC\Preview\PNG
- OC\Preview\JPEG
- OC\Preview\GIF
- OC\Preview\BMP
- OC\Preview\XBitmap
- OC\Preview\MarkDown
- OC\Preview\MP3
- OC\Preview\TXT

Les fournisseurs suivants sont désactivés par défaut pour des raisons de performances ou de confidentialité :

- OC\Preview\Illustrator
- OC\Preview\Movie
- OC\Preview\MSOffice2003
- OC\Preview\MSOffice2007
- OC\Preview\MSOfficeDoc
- OC\Preview\OpenDocument
- OC\Preview\PDF
- OC\Preview\Photoshop
- OC\Preview\Postscript
- OC\Preview\StarOffice
- OC\Preview\SVG
- OC\Preview\TIFF
- OC\Preview\Font

Note

Des étapes de débogage pour les aperçus MS Word sont disponibles dans la section *Configuration de l'application collaborative Documents* du manuel administrateur.

Les fournisseurs suivants ne sont pas disponibles sous Microsoft Windows :

- OC\Preview\Movie
- OC\Preview\MSOfficeDoc
- OC\Preview\MSOffice2003
- OC\Preview\MSOffice2007
- OC\Preview\OpenDocument
- OC\Preview\StarOffice

LDAP

Paramètres globaux utilisés par le service LDAP User and Group.

```
'ldapUserCleanupInterval' => 51,
```

Définit l'intervalle en minutes pour la tâche d'arrière-plan qui vérifie l'existence d'un utilisateur et le marque prêt à être supprimé. Ce nombre est tjours exprimé en minutes. Définir ceci à `0` désactive la fonctionnalité.

Voir les méthodes en ligne de commande (occ) ldap:show-remnants et user:delete

Commentaires

Paramètres globaux pour l'infrastructure de commentaires.

```
'comments.managerFactory' => '\OC\Comments\ManagerFactory',
```

Remplace le Comments Manager Factory par défaut. Ceci peut être utilisé si un autre gestionnaire de commentaires doit être utilisé – par exemple – s'il utilise le système de fichiers plutôt que la base de données pour conserver les commentaires.

```
'systemtags.managerFactory' => '\OC\SystemTag\ManagerFactory',
```

Remplace le System Tags Manager Factory par défaut. Ceci peut être utilisé si un autre gestionnaire de marqueurs doit être utilisé – par exemple – s'il utilise le système de fichiers plutôt que la base de données pour conserver les marqueurs.

Maintenance

Ces options servent à stopper l'activité des utilisateurs quand vous réalisez des opérations de maintenance.

```
'maintenance' => false,
```

Active le mode de maintenance pour désactiver ownCloud.

Si vous voulez empêcher les utilisateurs de se connecter à ownCloud avant de commencer une opération de maintenance, vous devez définir cette valeur à `true`. Veuillez garder à l'esprit que les utilisateurs céjà connectés sont déconnectés instantanément d'ownCloud.

```
'singleuser' => false,
```

Si défini à `true`, l'instance ownCloud sera indisponible pour tous les utilisateurs qui ne sont pas dans le groupe `admin`.

SSL

```
'openssl' => array(
        'config' => '/absolute/location/of/openssl.cnf',
),
```

Options SSL supplémentaires utilisées pour la configuration.

```
'enable_certificate_management' => false,
```

Autorise la configuration de certificats de confiance à l'échelle du système.

Services de configuration de mémoire cache

Services disponibles :

- `\OC\Memcache\APC` Service Alternative PHP Cache
- `\OC\Memcache\APCu` Service APC user
- `\OC\Memcache\ArrayCache` Service In-memory array-based (non recommandé)
- `\OC\Memcache\Memcached` Service Memcached
- `\OC\Memcache\Redis` Service Redis
- `\OC\Memcache\XCache` Service XCache

Conseil sur le choix du service :

- APCu est le plus facile à installer. La plupart des distributions ont le paquet. À utiliser pour un environnement mono-utilisateur pour tous les caches.
- Utiliser Redis ou Memcached pour les environnements distribués. Pour les cache local (vous pouvez en configurer deux), prenez APCu.

```
'memcache.local' => '\OC\Memcache\APCu',
```

Service de cache mémoire pour les données stockées localement.

- Utilisé pour les données spécifiques à un hôte, par exemple les chemins d'accès aux fichiers.

```
'memcache.distributed' => '\OC\Memcache\Memcached',
```

Service de mémoire cache pour les données distribuées.

- Utilisé pourles données d'installation spécifique, par exemple le cache de base de données
- Si non défini, prend par défaut la valeur de memcache.local

```
'redis' => array(
        'host' => 'localhost', // peut être aussi un socket de domaine Unix : '/tmp/redis.so
        'port' => 6379,
        'timeout' => 0.0,
        'password' => '', // facultatif, si non défini, aucun mot de passe se**ne sera utili
        'dbindex' => 0, // facultatif, si non défini, SELECT ne fonctionnera pas et utiliser
                   l'index de base de données par défaut du serveur Redis.
),
```

Détails de connexion de Redis pour utiliser la mémoire cache.

Pour une sécurité renforcée, il est recommandé de configurer Redis pour qu'il demande un mot de passe. Consulter http://redis.io/topics/security pour plus d'informations.

```
'memcached_servers' => array(
        // nom d'hôte, port et poids (facultatif). Voir aussi :
        // http://www.php.net/manual/en/memcached.addservers.php
        // http://www.php.net/manual/en/memcached.addserver.php
        array('localhost', 11211),
        //array('autre.hote.local', 11211),
),
```

Détails du serveur pour un ou plusieurs serveurs de mémoire cache.

```
'memcached_options' => array(
        // Définit les temporisations à 50ms
        \Memcached::OPT_CONNECT_TIMEOUT => 50,
        \Memcached::OPT_RETRY_TIMEOUT =>   50,
        \Memcached::OPT_SEND_TIMEOUT =>    50,
        \Memcached::OPT_RECV_TIMEOUT =>    50,
```

```
        \Memcached::OPT_POLL_TIMEOUT =>    50,

        // Active la compression
        \Memcached::OPT_COMPRESSION =>          true,

        // Active le « consistent hashing »
        \Memcached::OPT_LIBKETAMA_COMPATIBLE => true,

        // Active le protocole binaire
        \Memcached::OPT_BINARY_PROTOCOL =>      true,

        // Sérialiseur binaire qui sera activé si le module PECL igbinary est disponible
        //\Memcached::OPT_SERIALIZER => \Memcached::SERIALIZER_IGBINARY,
),
```

Options de connexion pour memcached, consulter http://apprize.info/php/scaling/15.html

```
'cache_path' => '',
```

Emplacement du dossier de cache, par défaut `data/$user/cache` où `$user` est l'utilisateur courant. Quand c'est spécifié, le format change pour `$cache_path/$user` où `$cache_path` est le répertoire de cache configuré et `$user` est l'utilisateur.

```
'cache_chunk_gc_ttl' => 86400, // 60*60*24 = 1 jour
```

TTL des objets stockés dans le dossier de cache avant qu'ils soient supprimés par le ramasse-miettes (en secondes). Augmenter cette valeur si les utilisateurs rencontrent des problèmes téléversant de gros fichiers à l'aide du client ownCloud.

Utilisation d'Object Store avec ownCloud

```
'objectstore' => array(
        'class' => 'OC\\Files\\ObjectStore\\Swift',
        'arguments' => array(
                // trystack utilisera votre identifiant Facebook comme nom d'utilisateur
                'username' => 'facebook100000123456789',
                // dans le tableau de bord de trystack, aller dans utilisateur -> paramètres
                // pour générer un mot de passe
                'password' => 'Secr3tPaSSWoRdt7',
                // doit déjà exister dans objectstore, le nom peut être différent
                'container' => 'owncloud',
                // créer le container s'il n'existe pas. par défaut à false
                'autocreate' => true,
                // obligatoire, dev-/trystack par défaut 'RegionOne'
                'region' => 'RegionOne',
                // Le frontal Identity / Keystone
                'url' => 'http://8.21.28.222:5000/v2.0',
                // obligatoire pour dev-/trystack
                'tenantName' => 'facebook100000123456789',
                // dev-/trystack utilise swift par défaut, par défaut 'cloudFiles'
                // si omis
                'serviceName' => 'swift',
                // Le type d'URL, facultatif
                'urlType' => 'internal'
        ),
),
```

Cet exemple montre comment configurer ownCloud pour stocker tous les fichiers dans un stockage d'objets Swift.

Il est important de noter qu'ownCloud en mode Object Store s'attend à avoir l'accès exclusif au container Object Store car il stocke seulement les données binaires pour chaque fichier. Les métadonnées sont actuellement conservées dans une base de données locale pour des raisons de performances.

ATTENTION : La mise en œuvre actuelle est incompatible avec les applications utilisant l'accès direct aux fichiers. Ceci inclut les application Encryption et Galerie. Galerie stockera les vignettes directement sur le système de fichiers et le chiffrement provoquera une grosse augmentation du temps système car les clés des fichiers doivent être récupérées en plus des fichiers.

Un moyen de tester est de demander un compte trystack sur http://trystack.org/

Partage

Paramètres globaux de partage.

```
'sharing.managerFactory' => '\OC\Share20\ProviderFactory',
```

Remplace le Share Provider Factory par défaut. Ceci peut être utilisé si un autre fournisseur de partage peut être utilisé qui – par exemple – utilise le système de fichiers au lieu de la base de données pour conserver les informations de partage.

Toutes les autres options de configuration

```
'dbdriveroptions' => array(
        PDO::MYSQL_ATTR_SSL_CA => '/file/path/to/ca_cert.pem',
        PDO::MYSQL_ATTR_INIT_COMMAND => 'SET wait_timeout = 28800'
),
```

Options de pilotes supplémentaires pour la connexion à la base de données, par exemple pour activer le chiffrement SSL dans MySQL ou spécifier une temporisation différente sur un hôte peu performant.

```
'sqlite.journal_mode' => 'DELETE',
```

Le mode journal de sqlite3 peut être spécifié en utilisant ce paramètre de configuration. La valeur peut être `WAL` ou `DELETE`. Consulter https://www.sqlite.org/wal.html pour plus de détails.

```
'supportedDatabases' => array(
        'sqlite',
        'mysql',
        'pgsql',
        'oci',
),
```

Type de bases de données gérées pour l'installation.

Disponible :

- sqlite (SQLite3 - pas dans l'Édition Entreprise)
- mysql (MySQL)
- pgsql (PostgreSQL)
- oci (Oracle - Édition Entreprise seulement)

```
'tempdirectory' => '/tmp/owncloudtemp',
```

Remplace la valeur du répertoire où ownCloud stocke ses fichiers temporaires. C'est utile dans les situations où le répertoire temporaire du système a une taille limitée ou si des stockages externes qui ne gèrent pas le streaming sont utilisés.

The Web server user must have write access to this directory.

```
'hashingCost' => 10,
```

Coût du « hash » utilisé pour la génération des « hashes » par ownCloud. Utiliser une valeur plus grande nécessite plus de puissance processeur pour calculer les « hashes ».

```
'blacklisted_files' => array('.htaccess'),
```

Met en liste noire un ou des fichiers spécifiques et empêche le téléversement de fichier de même nom. `.htaccess` est bloqué par défaut.

ATTENTION : N'UTILISEZ CECI QUE SI VOUS SAVEZ CE QUE VOUS FAITES.

```
'share_folder' => '/',
```

Définit un dossier par défaut pour les fichiers et dossiers partagés autre que la racine.

```
'theme' => '',
```

Si vous mettez en place un thème pour ownCloud, saisissez son nom ici.

L'emplacement par défaut pour les thèmes est `owncloud/themes/`.

```
'cipher' => 'AES-256-CFB',
```

Le type de chiffrement par défaut. Actuellement seuls AES-128-CFB et AES-256-CFB sont gérés.

```
'minimum.supported.desktop.version' => '1.7.0',
```

La version minimum du client ownCloud pour ordinateur autorisé à faire des synchronisations avec cette instance d'ownCloud. Toutes les connexions effectuées par ds clients de version plus ancienne seront rejetées par le serveur. Par défaut, la version minimale d'ownCloud officiellement supportée au moment de la publication de cette version de serveur.

Si vous modifiez cela, veuillez noter que les anciennes versions non supportées des clients ownCloud pour ordinateur peuvent ne pas fonctionner comme attendu et conduire à des pertes de données ou d'autres résultats inattendus.

```
'quota_include_external_storage' => false,
```

EXPÉRIMENTAL : Option pour inclure les stockages externes dans le calcul du quota. Par défaut à `false`.

```
'filesystem_check_changes' => 0,
```

Spécifie la fréquence de vérification des changements effectués sur le système de fichiers local en dehors d'ownCloud (le répertoire `data/` d'ownCloud et les montages NFS dans `data/`). Ceci ne s'applique pas aux stockages externes.

0 -> Ne jamais vérifier le système de fichiers pour des modifications externes. Ceci permet d'améliorer les performances quand il est certain qu'aucune modification n'est faite directement sur le système de fichiers.

1 -> Vérifier chaque fichier ou dossier au plus une fois par requête. Recommandé pour une utilisation générale si des modifications externes peuvent survenir.

```
'part_file_in_storage' => true,
```

Par défaut, ownCloud stockera les fichiers partiels créés pendant les téléversements sur le même stockage que la cible du téléversement. Définir ceci à `false` stockera les fichiers partiels dans la racine des dossiers utilisateurs, ce qui pourrait être nécessaire avec certains stockages externes ayant des capacités de renommage limitée.

```
'asset-pipeline.enabled' => false,
```

Tous les fichiers CSS et JS seront servis par le serveur Web statiquement en un seul fichier JS et un seul fichier CSS si ce paramètre est défini à `true`. Ceci améliore les performances.

```
'assetdirectory' => '/var/www/owncloud',
```

Le répertoire parent où seront stockés les jeux de fichiers CSS et JS si le pipelining est activé. Par défaut, c'est le répertoire ownCloud. Les jeux de fichiers seront stockés dans un sous-répertoire nommé `assets`. Le serveur *doit* être configuré pour servir ce répertoire comme $WEBROOT/assets.

Vous n'avez besoin de changer ce paramètre que si le répertoire principal d'ownCloud n'est pas accessible en écriture par le serveur Web dans votre configuration.

```
'mount_file' => '/var/www/owncloud/data/mount.json',
```

Le répertoire où le fichier `mount.json` doit être stocké. Par défaut, c'est le répertorie `data/mount.json` dans le répertoire ownCloud.

```
'filesystem_cache_readonly' => false,
```

Si défini à `true`, empêche ownCloud de modifier le cache en raison de changements survenus sur le système de fichiers pour tout le stockage.

```
'secret' => '',
```

Clé utilisée par ownCloud pour divers usages, comme le chiffrement de données par exemple. Si vous perdez cette clé, il y aura corruption de données.

```
'trusted_proxies' => array('203.0.113.45', '198.51.100.128'),
```

Liste des serveurs proxy de confiance.

Si vous configurez cela, envisagez aussi de définir le paramètre *forwarded_for_headers* qui est par défaut *HTTP_X_FORWARDED_FOR* (l'en-tête *X-Forwarded-For*).

```
'forwarded_for_headers' => array('HTTP_X_FORWARDED', 'HTTP_FORWARDED_FOR'),
```

Les en-têtes à qui il peut être fait confiance comme adresses IP des clients en combinaison avec *trusted_proxies*. Si l'en-tête HTTP ressemble à `X-Forwarded-For`, utilisez alors `HTTP_X_FORWARDED_FOR` ici.

Si ce paramètre est défini de façon incorrecte, un client peut déguiser son adresse IP pour la présenter à ownCloud, et passer alors outre les contrôles d'accès en rendant les journaux inutiles !

Par défaut 'HTTP_X_FORWARED_FOR' si non défini.

```
'max_filesize_animated_gifs_public_sharing' => 10,
```

Taille maximale pour les fichiers gif animés sur les sites de partage publics.

Si le fichier gif est trop gros, un aperçu statique sera affiché.

La valeur représente la taille maximale du fichier en méga-octets. Par défaut à `10`. Définir à `-1` pour ne pas imposer de limite.

```
'filelocking.enabled' => true,
```

Enables transactional file locking.

This is enabled by default.

Prevents concurrent processes from accessing the same files at the same time. Can help prevent side effects that would be caused by concurrent operations. Mainly relevant for very large installations with many users working with shared files.

```
'filelocking.ttl' => 3600,
```

Définit la durée de vie (TTL) des verrous en secondes.

Tout verrou plus âgé sera automatiquement purgé.

Si ce paramètre n'est pas défini, la valeur par défaut sera d'une heure ou la valeur du paramètre PHP max_execution_time, qui est encore plus élevée.

```
'memcache.locking' => '\\OC\\Memcache\\Redis',
```

Service de mémoire cache pour le verrouillage de fichier.

Parce que la plupart des services de mémoire cache peuvent purger des valeurs sans prévenir, il est vivement recommandé d'utiliser Redis pour *éviter les pertes de données*.

```
'upgrade.disable-web' => false,
```

Désactive la mise à jour basée sur le Web.

```
'debug' => false,
```

Permet de passer cette instance ownCloud en mode débogage.

N'activez ceci que pour le développement local et pas sur les environnements de production. Ceci désactive le « minifier » et ajoute des informations de débogage additionnelles.

```
'data-fingerprint' => '',
```

Définit l'empreinte de données sur les données actuellement servies.

C'est une propriété utilisée par les clients pour trouver si une sauvegarde a été restaurée sur le serveur. Une fois qu'une sauvegarde a été restaurée lancez la commande `./occ maintenance:data-fingerprint` pour définir une nouvelle valeur.

Mettre à jour ou supprimer cette valeur peut immobiliser les clients connectés jusqu'à ce que l'utilisateur ait résolu les conflits.

```
'copied_sample_config' => true,
```

Cette entrée sert juste à afficher un avertissement pour le cas où quelqu'un aurait copié le fichier d'exemple de configuration. N'AJOUTEZ PAS CE PARAMÈTRE À VOTRE CONFIGURATION !

Si vous, courageuse personne, avez lu ceci jusqu'ici, vous savez que vous ne devez modifier *AUCUN* paramètre dans ce fichier sans avoir lu la documentation auparavant.

Options de configuration des applications

Rétention des activités pour l'application Activité :

```
'activity_expire_days' => 365,
```

Chaque jour une tâche cron est exécutée, qui supprime, pour tous les utilisateurs, les activités dont l'âge est supérieur au nombre de jours défini dans `activity_expire_days`.

```
'wnd.logging.enable' => true,
```

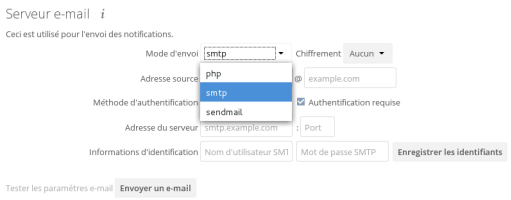
Ceci active le mode de débogage pour l'application windows_network_drive.

Configuration des courriels

ownCloud peut envoyer des courriels de réinitialisation de mot de passe, notifier aux utilisateurs de nouveaux partages de fichiers ou des modifications de fichiers, et des notifications d'activité. Vos utilisateurs peuvent configurer les notifications qu'ils veulent recevoir sur leur page personnelle.

ownCloud n'intègre pas de serveur de messagerie complet mais se connecte au serveur de messagerie existant. Vous devez avoir un serveur de messagerie fonctionnel pour qu'ownCloud puisse envoyer des courriels. Le serveur de messagerie peut être installé sur le même serveur qu'ownCloud ou sur un serveur distant.

La version 7 d'ownCloud introduit une nouvelle fonctionnalité : un assistant de configuration graphique pour la messagerie.



Avec ce nouvel assistant, connecter ownCloud à votre serveur de messagerie est rapide et facile. L'assistant remplit automatiquement les valeurs dans le fichier `config/config.php`.

L'assistant de configuration gère trois types de connexion : SMTP, PHP et Sendmail. Utilisez le configurateur SMTP pour un serveur distant et PHP ou Sendmail quand le serveur de messagerie est sur le même serveur qu'ownCloud.

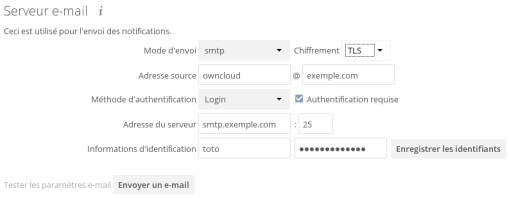
Note

L'option Sendmail se réfèère au serveur SMTP Sendmail et toute autre alternatives telles que Postfix, Exim ou Courier. Celles-ci contiennent toutes un binaire de `sendmail` et sont donc librement interchangeables.

Configuration du serveur SMTP

Vous avez besoin des informations suivantes de la part de votre administrateur de messagerie pour connecter ownCloud à un serveur distant SMTP :

- type de chiffrement : Aucun, SSL/TLS ou STARTTLS ;
- l'adresse d'expéditeur qui sera visible dans les courriels envoyés par ownCloud ;
- si l'authentification est nécessaire ;
- la méthode d'authentification : Aucune, Login, En clair ou Gestionnaire du réseau NT ;
  l'adresse IP du serveur ou son nom complet (FQDN) ;
- les identifiants, si nécessaire.



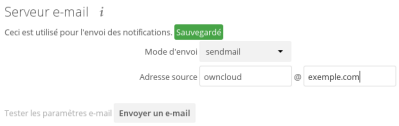
Vos modifications sont enregistrées immédiatement et vous pouvez cliquer sur le bouton « Envoyer un e-mail » pour tester votre configuration. Ceci envoie un message de test à l'adresse électronique configurée sur votre page personnelle. Le message indique:

```
If you received this email, the settings seem to be correct.

--
ownCloud - services web sous votre contrôle
https://owncloud.org
```

Configuration de PHP et Sendmail

La configuration de PHP ou Sendmail nécessite que vous sélectionniez l'un d'eux et que vous saisissiez l'adresse de retour désirée.



Comment décider lequel utiliser ? Le mode PHP utilise votre bianire local `sendmail`. Utilisez ceci si vous voulez utilisez `php.ini` pour contrôler certaines fonctions de votre serveur de messagerie, comme les chemins, les en-têtes ou le passage d'options de commandes supplémentaires à votre binaire `sendmail`. Ceci varie en fonction du serveur que vous utilisez, aussi, veuillez consulter la documentation de votre serveur pour connaître les options.

Dans la plupart des cas, l'option `smtp` est la meilleure car elle évite les étapes supplémentaires de la configuration de PHP et vous pouvez contrôler toutes les options de votre serveur de messagerie en un seul endroit, dans la configuration de votre serveur de messagerie.

Utilisation de modèles de courriel

Une autre fonctionnalité utile est la possibilité de modifier des modèles de courriel. Vous pouvez maintenant modifier les modèles de courriels d'ownCloud sur votre page d'administration. Voici les modèles disponibles :

- Email de partage (HTML) -- Version HTML des courriels informant les utilisateurs d'un nouveau partage de fichiers
- Email de partage (version texte simple alternative) -- Courriels en texte brut informant les utilisateurs d'un nouveau partage de fichiers
- Mesage envoyé en cas de perte de mot de passe -- Courriels de réinitialisation de mot de passe pour les utilisateurs ayant perdu leur mot de passe
- Email de notification d'activité -- Notification d'activité que les utilisateurs ont activé sur leur page parsonnelle dans la section Notifications

En plus de fournir des modèles de courriels, cette fonctionnalité permet d'appliquer un thème pré-configuré aux courriels.

Pour modifier un modèle de courriel :

1. rendez-vous sur la page d'administration ;
2. sélectionnez la section « Modèles de mail » ;
3. choisissez un modèle dans la liste déroulante ;
4. appliquez les modifications désirées dans le modèle.

Les modèles sont écrits en PHP et en HTML et sont déjà chargés avec les variables appropriées telles que le nom d'utilisateur, les liens partagés et les noms de fichiers. Vous pouvez en faisant attention les modifier sans connaître PHP ou HTML ; ne touchez pas au code, mais vous pouvez modifier les portions de texte des messages. Par exemple, voici le modèle de courriel pour la perte du mot de passe :

```
<?php

   echo str_replace('{link}', $_['link'], $l->t('Use the following link to
   reset your password: {link}'));
```

Vous pouvez modifier la portion de texte du modèle : `Use the following link to reset your password:` pour `Veuillez utiliser le lien suivant pour réinitialiser votre mot de passe. Si vous n'avez pas demander de réinitialisation de mot de passe, veuillez ignorer ce message.`

À nouveau, soyez très prudent à ne rien modifier d'autre que le texte car la plus petite erreur dans le code rendrait le modèle inopérant.

> ***Note***
>
> Vous pouvez modifier les modèles directement dans la boîte de texte ou copier le contenu dans un éditeur de texte puis le recopier dans la boîte de texte après avoir effectué les modifications.

Définitionb des paramètres sdu serveur de messagerie dans config.php

Si vous préférez, vous pouvez définir les paramètres du serveur de messagerie dans le fichier `config/config.php`. Les exemples suivants sont pour SMTP, PHP, Sendmail, and Qmail.

SMTP

Si vous voulez envoyer des courriels en utilisant un serveur SMTP local ou distant, il est nécessaire de saisir le nom ou l'adresse IP du serveur, et, de manière facultative, le faire suivre d'un double point (« : ») et du numéro du port, par ex. **:425**. Si cette valeur n'est pas donnée, le port utilisé par défaut sera le port 25/tcp à moins que vous ne modifiiez le paramètre **mail_smtpport**. Plusieurs serveurs peuvent être indiqués, séparés par des point-virgules :

```
<?php

  "mail_smtpmode"     => "smtp",
  "mail_smtphost"     => "smtp-1.serveur.dom;smtp-2.serveur.dom:425",
  "mail_smtpport"     => 25,
```

ou

```
<?php

  "mail_smtpmode"     => "smtp",
  "mail_smtphost"     => "smtp.serveur.dom",
  "mail_smtpport"     => 425,
```

Si un scanner de logiciels malveillants ou de courriels indésirables s'exécute sur le serveur SMTP, il pourrait être nécessaire d'augmenter la temporisation du serveur SMTP par exemple à 30 secondes :

```
<?php

  "mail_smtptimeout"  => 30,
```

Si le serveur accepte les connexions non sécurisées, le paramètre par défaut peut être utilisé :

```
<?php

"mail_smtpsecure"   => '',
```

Si le serveur n'accepte que les connexions sécurisées, vous pouvez choisir entre deux variantes :

SSL/TLS

Une connexion sécurisée sera initiée en utilisant SSL/TLS via SMTPS sur le port par défaut 465/tcp :

```
<?php

  "mail_smtphost"     => "smtp.serveur.dom:465",
  "mail_smtpsecure"   => 'ssl',
```

STARTTLS

Une connexion sécurisée sera initiée en utilisant le protocole STARTTLS via SMTP sur le port par défaut 25/tcp :

```
<?php

  "mail_smtphost"     => "smtp.serveur.dom",
  "mail_smtpsecure"   => 'tls',
```

Une alternative est le port 587/tcp (recommndé) :

```
<?php

  "mail_smtphost"     => "smtp.server.dom:587",
  "mail_smtpsecure"   => 'tls',
```

Authentification

Et enfin, il est nécessaire de configurer le serveur SMTP pour indiquer s'il utilise ou non l'authentification. Si ce n'est pas le cas, les valeurs par défaut peuvent être conservées telles quelles.

```
<?php

  "mail_smtpauth"     => false,
  "mail_smtpname"     => "",
  "mail_smtppassword" => "",
```

Dans le cas contraire, vous devez définir l'utilisateur et le mot de passe et de manière facultative, choisir le type d'authentification **LOGIN** (par défaut) ou **PLAIN**.

```
<?php

  "mail_smtpauth"     => true,
  "mail_smtpauthtype" => "LOGIN",
  "mail_smtpname"     => "utilisateur",
  "mail_smtppassword" => "mot_de_passe",
```

PHP mail

Si vous voulez utiliser PHP mail, il est nécessaire d'avoir un serveur de messagerie déjà installé et opérationnel sur votre serveur. Le programme utilisé pour envoyer les courriels est défini par les paramètres de configuration du fichier **php.ini**. (Sur les systèmes de type Unix, ce sera très vraisemblablement Sendmail). ownCloud devrait être capable d'envoyer des courriels sans autre paramétrage.

```
<?php

  "mail_smtpmode"     => "php",
  "mail_smtphost"     => "127.0.0.1",
  "mail_smtpport"     => 25,
  "mail_smtptimeout"  => 10,
  "mail_smtpsecure"   => "",
  "mail_smtpauth"     => false,
  "mail_smtpauthtype" => "LOGIN",
  "mail_smtpname"     => "",
  "mail_smtppassword" => "",
```

Sendmail

Si vous voulez utiliser le programme bien connu Sendmail pour envoyer des courriels, il est nécessaire d'avoir un serveur de messagerie installé et opérationnel sur votre serveur de type Unix. Le binaire (**/usr/sbin/sendmail**) fait généralement partie du système d'exploitation. ownCloud devrait pouvoir envoyer des courriels sans autre configuration.

```
<?php

  "mail_smtpmode"     => "sendmail",
  "mail_smtphost"     => "127.0.0.1",
  "mail_smtpport"     => 25,
  "mail_smtptimeout"  => 10,
  "mail_smtpsecure"   => "",
  "mail_smtpauth"     => false,
  "mail_smtpauthtype" => "LOGIN",
  "mail_smtpname"     => "",
  "mail_smtppassword" => "",
```

qmail

Si vous voulez utiliser le programme qmail pour envoyer des courriels, il est nécessaire d'avoir un système qmail installé et opérationnel sur votre serveur. Le binaire sendmail (**/var/qmail/bin/sendmail**) sera alors utilisé pour envoyer des courriels. ownCloud devrait pouvoir envoyer des courriels sans autre configuration.

```
<?php

  "mail_smtpmode"     => "qmail",
  "mail_smtphost"     => "127.0.0.1",
  "mail_smtpport"     => 25,
  "mail_smtptimeout"  => 10,
  "mail_smtpsecure"   => "",
  "mail_smtpauth"     => false,
  "mail_smtpauthtype" => "LOGIN",
  "mail_smtpname"     => "",
  "mail_smtppassword" => "",
```

Envoi d'un courriel de test

Pour tester votre configuration de messagerie, enregistrez votre adresse électronique sur votre page personnelle et utilisez le bouton **Envoyer un e-mail** dans la section *Serveur e-mail* de votre page d'administration.

Utilisation de certificats auto-signés

Lors de l'utilisation de certifcats auto-signés sur le serveur distant SMTP, le certificat doit être importé dans ownCloud. Veuillez consulter *Importation de certificats SSL personnel ou global* pour plus d'informations.

Dépannage

Si vous n'arrivez pas à envoyer de courriels, essayez d'activer le mode débogage. Pour cela, activez le paramètre `mail_smtpdebug` dans `config/config.php`.

```
<?php

  "mail_smtpdebug" => true;
```

Note

Immédiatement après avoir cliqué sur le bouton **Envoyer un e-mail**, comme décrit précédemment, plusieurs messages **SMTP -> get_lines(): ...** apparaissent à l'écran. Ceci est le comportement attendu et peut être ignoré.

Question : Pourquoi mon domaine Web est-il différent de mon domaine de messagerie ?

Réponse : Le nom de domaine par défaut utilisé pour l'adresse de l'expéditeur est celui du nom d'hôte où est installé ownCloud. Si vous avez un nom de domaine de messagerie différent, vous pouvez passer outre ce comportement en définissant le paramètre de configuration suivant :

```
<?php

  "mail_domain" => "exemple.com",
```

Ce paramètre s'appliquera à tous les courriels envoyé par ownCloud (par exemple, pour le courriel de réinitialisation de mot de passe). L'adresse de l'expéditeur apparaîtra ainsi:

```
no-reply@exemple.com
```

Question : Comment savoir si le serveur SMTP est accessible ?

Réponse : Utilisez la commande `ping` pour vérifier l'accessibilité du sevreur:

```
ping smtp.serveur.dom
```

```
PING smtp.server.dom (ip-address) 56(84) bytes of data.
64 bytes from your-server.local.lan (192.168.1.10): icmp_req=1 ttl=64
time=3.64ms
```

Question : Comment savoir sur le serveur SMTP écoute sur un port TCP spécifique ?

Réponse : Le meilleur moyen d'obtenir les informations de configuration du serveur de messagerie est de le demander à l'administrateur du serveur de messagerie. Si vous êtes l'administrateur du serveur de messagerie ou que vous avez besoin de cette information en urgence, vous pouvez utiliser la commande `netstat`. Cet exemple montre tous les serveurs actifs sur votre système et les ports sur lesquels ils écoutent. Le serveur SMTP écoute sur le port 25 de l'hôte local.

```
# netstat -pant
```

```
Active Internet connections (servers and established)
Proto Recv-Q Send-Q Local Address   Foreign Address  State  ID/Program name
tcp    0      0    0.0.0.0:631     0.0.0.0:*        LISTEN   4418/cupsd
tcp    0      0    127.0.0.1:25    0.0.0.0:*        LISTEN   2245/exim4
tcp    0      0    127.0.0.1:3306  0.0.0.0:*        LISTEN   1524/mysqld
```

- 25/tcp est le protocole non chiffré smtp
- 110/tcp/udp est le protocole non chiffré pop3
- 143/tcp/udp est le protocole non chiffré imap4
- 465/tcp est le protocole chiffré smtps
- 993/tcp/udp est le protocole chiffré imaps
- 995/tcp/udp est le protocole chiffré pop3s

Question : Comment déterminer si le serveur SMTP gère le protocole SMTPS ?

Réponse : Une bonne indication que le serveur SMTP gère le protocole SMTPS est qu'il écoute sur le port **465**.

Question : Comment déterminer le type d'authentification et les protocoles de chiffrement que sait gérer le serveur ?

Réponse : Les serveurs SMTP annoncent généralement la disponibilité de STARTTLS immédiatement après l'établissement d'une connexion. Vous pouvez facilement vérifier cela en utilisant la commande `telnet`.

Note

Vous devez saisir les lignes indiquées pour obtenir les informations affichées.

```
telnet smtp.domaine.dom 25
```

```
Trying 192.168.1.10...
Connected to smtp.domaine.dom.
Escape character is '^]'.
220 smtp.domaine.dom ESMTP Exim 4.80.1 Tue, 22 Jan 2013 22:39:55 +0100
EHLO votre-serveur.local.lan                    # <<< Saisissez cette commande
250-smtp.domaine.dom Hello votre-serveur.local.lan [adresse-ip]
250-SIZE 52428800
250-8BITMIME
250-PIPELINING
250-AUTH PLAIN LOGIN CRAM-MD5                 # <<< Protocoles d'authentification gérés
250-STARTTLS                                  # <<< Le chiffrement est géré
250 HELP
QUIT                                          # <<< Saisissez cette commande
221 smtp.domaine.dom closing connection
Connection closed by foreign host.
```

Activation du mode de débogage

Si vous n'arrivez pas à envoyer de courriels, essayez d'activer le mode débogage. Pour cela, activez le paramètre `mail_smtpdebug` dans `config/config.php`.

```
<?php

  "mail_smtpdebug" => true,
```

Note

Immédiatement après avoir cliqué sur le bouton **Envoyer un e-mail**, comme décrit précédemment, plusieurs messages **SMTP -> get_lines(): ...** apparaissent à l'écran. Ceci est le comportement attendu et peut être ignoré.

Liens vers des sites externes

Vous pouvez intégrer des sites Web externes dans vos pages ownCloud avec l'application External Sites, comme le montre cette copie d'écran.

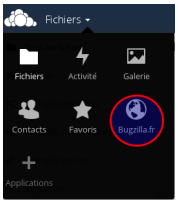
Cliquer pour agrandir

C'est utile pour accéder rapidement à des pages Web importantes comme les manuels et les pages d'information d'ownCloud pour votre société et pour présenter des pages externes intégrées à votre ownCloud personnalisé, si vous utilisez votre propre thème.

L'application External Sites est incluse dans toutes les versions d'ownCloud. Rendez-vous sur la pages des applications, puis dans la section « Désactivées » pour l'activer. Rendez-vous ensuite sur votre page d'administration pour créer vos liens qui seront enregistrés automatiquement. Il y a un menu déroulant pour sélectionner une icône, mais il y a une seule icône par défaut, il n'est donc pas nécessaire d'en sélectionner une. Passer la souris à droite du lien pour découvrir une icône représentant une corbeille permettant de supprimer le lien.

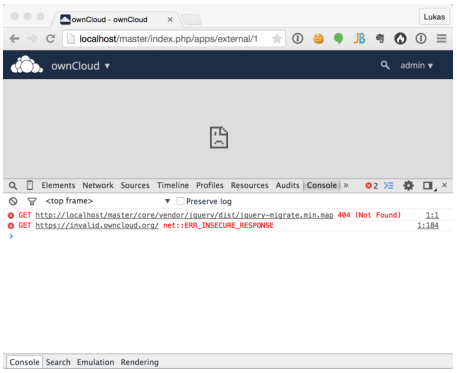
Cliquer pour agrandir

Les liens apparaissent dans le menu déroulant d'ownCloud sur le coin supérieur gauche de l'écran, après avoir rechargé la page et sont matérialisés par des icônes représentant un globe terrestre.

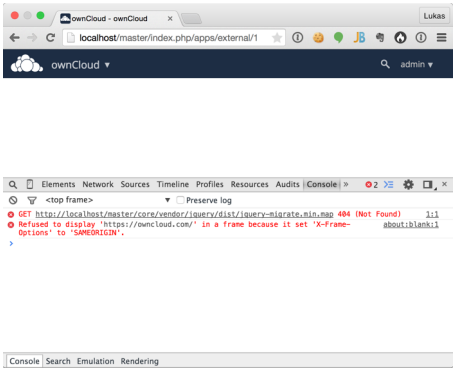


Vos liens peuvent fonctionner ou pas en fonction de la façon dont les navigateurs gèrent les URL HTTP et HTTPS, et car l'application External Sites intègre les liens externes dans des iFrames. Les navigateurs modernes essaient de protéger leurs utilisateurs de liens dangereux, et des applications telles que Privacy Badger et les bloqueurs de publicité peuvent empêcher les pages de s'afficher. Il est vivement recommandé de forcer l'utilisation de HTTPS sur votre serveur ownCloud. Ne baissez pas le niveau de sécurité juste pour faire fonctionner ces pages intégrées, car après tout, vous pouvez librement y accéder en dehors d'ownCloud.

La plupart des sites Web proposent des fonctionnalités de connexion en utilisant les en-têtes HTTP `X-Frame-Options` ou `Content-Security-Policy` qui indiquent aux navigateurs de ne pas intégrer leurs pages pour des raisons de sécurité. (par ex. du « Clickjacking »). Vous pouvez normalement vérifier pourquoi l'intégrations du site Web n'est pas possible en utilisant l'outil de console de votre navigateur. Par exemple, ce site a un certificat SSL invalide.



Sur cette page, X-Frame-Options empêche d'intégrer la page.



Vous ne pouvez pas faire grand chose contre cela, mais si vous êtes curieux, vous pouvez voir ce qu'il se passe.

Dépôts de téléchargement personnalisés

Vous pouvez configurer les URL vers vos propres dépôts pour le téléchargement des clients pour ordinateurs et des applications pour mobiles dans le fichier config/config.php. Cet exemple montre les adresses de téléchargement par défaut :

```
<?php

  "customclient_desktop" => "https://owncloud.org/sync-clients/",
  "customclient_android" => "https://play.google.com/store/apps/details?id=com.owncloud.andr
  "customclient_ios"     => "https://itunes.apple.com/us/app/owncloud/id543672169?mt=8",
```

Il suffit de remplacer les URL par les vôtres.

Vous pouvez tester des URL alternatives sans modifier le fichier config/config.php en définissant une URL de test en variable d'environnement:

```
export OCC_UPDATE_URL=https://test.exemple.com
```

Quand vous avez terminé, vous pouvez désactiver la variable d'environnement:

```
unset OCC_UPDATE_URL
```

Configuration de la base de connaissances

L'utilisation d'ownCloud s'explique plus ou moins d'elle-même, cependant, un utilisateur pourrait rencontrer un problème et avoir besoin de consulter la documentation ou la base de connaissances. Pour faciliter l'accès à la documentation et à la base de connaissances d'ownCloud, un menu d'aide est affiché dans le menu des paramètres par défaut.

Paramètres

Si vous voulez désactiver le menu d'aide d'ownCloud, vous pouvez utiliser le paramètre **knowledgebaseenabled** dans le fichier config/config.php.

```
<?php

  "knowledgebaseenabled" => true,
```

Note

Désactiver le menu d'aide pourrait augmenter le nombre de requêtes de support de vos utilisateurs.

Configuration de la langue

Dans une situation normale, ownCloud détectera automatiquement la langue de l'interface Web. Si cela ne fonctionne pas correctement ou si vous voulez être sûr qu'ownCloud s'affiche toujours dans une langue donnée, vous pouvez utiliser le paramètre **default_language**.

Veuillez garder à l'esprit que cela n'affectera pas les préférences de langue de l'utilisateur, qui ont été paramétrées dans sa page personnelle, dans la section « Langue » quand il s'est connecté.

Veuillez consulter settings/languageCodes.php pour connaître la liste des codes de langue gérés par ownCloud.

Paramètres

```
<?php

  "default_language" => "en",
```

Ce paramètre peut être défini dans le fichier config/config.php.

Configuration des fichiers journaux

Utilisez les journaux d'ownCloud pour voir l'état du système ou pour aider à la résolution de problèmes. Vos pouvez ajuster le niveau de journalisation et choisir entre le fichier journal d'ownCloud ou le fichier syslog du système.

Paramètres

Les niveaux de journalisation vont de **DEBUG**, qui consigne toute activité, à **FATAL**, qui ne consigne que les erreurs fatales.

- **0** : DEBUG : Toute activité. C'est le mode le plus détaillé.
- **1** : INFO : L'activité telles que les connexions utilisateurs et l'activité sur les fichiers, plus les avertissements, les erreurs et les erreurs fatales.
- **2** : WARN : Les opérations réussies, mais avec les avertissements de problèmes potentiels, plus les erreurs et les erreurs fatales.
- **3** : ERROR : Une opération échoue, mais d'autres services et opérations continuent, plus les erreurs fatales.
- **4** : FATAL : Le serveur s'arrête.

Par défaut, le niveau est défini à **2** (WARN). Utilisez **DEBUG** seulement quand vous devez diagnostiquer un problème, puis remettez le niveau dans un mode moins verbeux. **DEBUG** consigne beaucoup d'informations ce qui peut altérer les performances du serveur.

Les niveaux de journalisation sont définis dans le fichier config/config.php, ou sur la page d'administration.

ownCloud

Toutes les informations seront écrites dans un fichier journal séparé qui peut être lu en utilisant la visionneuse dans la page d'administration. Par défaut, un fichier nommé **owncloud.log** sera créé dans le répertoire configuré par le paramètre **datadirectory** dans le fichier config/config.php.

Le format de date peut être modifié en utilisant le paramètre **logdateformat** du fichier config/config.php. Par défaut, le paramètre PHP date function « *c* » est utilisé, et par conséquent l'heure et la date seront écrites dans le format « *2013-01-10T15:20:25+02:00* ». En utilisant le format de date de l'exemple ci-dessous, la date et l'heure seront écrites sous cette forme : « *January 10, 2013 15:20:25* ».

```
"log_type" => "owncloud",
"logfile" => "owncloud.log",
"loglevel" => "3",
"logdateformat" => "F d, Y H:i:s",
```

syslog

Toutes les informations seront envoyées vers le démon syslog par défaut.

```
"log_type" => "syslog",
"logfile" => "",
"loglevel" => "3",
```

Élévation conditionnelle du niveau de journalisation

Vous pouvez configurer le niveau de journalisation pour passer automatiquement à `debug` quand l'une des trois conditions est rempplie :

`shared_secret` : si un paramètre de requête avec le nom `log_secret` est défini à cette valeur ;

`users` : si la requête est faite par l'un des utilisateurs spécifiés ;

`apps`: si le message de journalisation est invoqué par l'une des applications spécifiées.

L'exemple suivant montre à quoi ressemble ces trois conditions

```
'log.condition' => [
   'shared_secret' => '57b58edb6637fe3059b3595cf9c41b9',
   'users' => ['utilisateur'],
   'apps' => ['files'],
],
```

Guide de durcissement et de sécurité

ownCloud est livré avec des paramètres de sécurité par défaut qui n'ont pas besoin d'être modifiés par les administrateurs. Cependant, dans certains cas, un durcissement de la sécurité peut être appliqué dans des scénarios où les administrateurs ont le contrôle total sur leur instance ownCloud. Cette page suppose que vous exécutiez un serveur ownCloud sur Apache2 dans un environnement Linux.

> ***Note***
>
> ownCloud vous avertira sur la page d'administration si des options de sécurité critiques sont manquantes. Cependant, il appartient toujours à l'administrateur du serveur de revoir et de maintenir la sécurité du système.

Limite de la longueur du mot de passe

ownCloud utilise l'algorithme bcrypt, et par conséquent, pour des raisons de sécurité et de performances, par exemple un déni de service, car la demande en CPU augmente exponentiellement, il ne vérifie que les 72 premiers caractères des mots de passe. Ceci s'applique à tous les mots de passe utilisés dans ownCloud : les mots de passe utilisateur, ceux des partages de liens et ceux des partages externes.

Système d'exploitation

Accès en lecture de PHP à `/dev/urandom`

ownCloud utilise un mixeur conforme à la RFC 4086 ("Randomness Requirements for Security") pour générer des nombres pseudo-aléatoires sécurisés cryptographiquement. Ceci signifie que lors de la génération d'un nombre aléatoire, ownCloud demandera de multiples nombres aléatoires à différentes sources et en dérivera le nombre aléatoire final.

La génération de nombres aléatoires essaiera aussi d'obtenir des nombres aléatoires de `/dev/urandom`. Par conséquent, il est vivement recommandé de configurer PHP pour qu'il puisse lire des données aléatoires sur celui-ci.

> ***Note***
>
> Quand `open_basedir` est configuré dans le fichier `php.ini`, assurez-vous d'inclure `/dev/urandom`.

Activation des modules de durcissement tel que SELinux

Il est vivement recommandé d'activer les modules de durcissement tel que SELinux quand c'est possible. Consulter *Configuration SELinux* pour en apprendre plus sur SELinux.

Déploiement

Placement du répertoire data en dehors de la racine Web

Il est vivement recommandé de placer le répertoire data en dehors de la racine Web (c-à-d. en dehors de `/var/www`). Il est plus facile de le faire sur une nouvelle installation.

Désactivation de la génération d'aperçus

ownCloud peut générer des aperçus de type de fichiers courants tels que les images ou les fichiers texte. Par défaut, la génération d'aperçus pour quelques types de fichiers que nous considérons comme suffisamment sûrs pour le déploiement est activée par défaut. Cependant, les administrateurs doivent garder à l'esprit que ces aperçus sont générés en utilisant des bibliothèques PHP écrites en C et qui peuvent être vulnérables aux attaques par vecteurs.

Pour des déploiements très sécurisés, nous recommandons de désactiver la génération d'aperçus en définissant `enable_previews à false` dans le fichier `config.php`. En tant qu'administrateur, vous pouvez aussi gérer quels fournisseurs d'aperçus sont activés en utilisant l'option `enabledPreviewProviders`.

Utilisation de HTTPS

L'utilisation d'ownCloud sans connexion chiffrée HTTPS rend votre serveur vulnérable aux attaques de type man-in-the-middle (MITM), et permet potentiellement l'interception de données utilisateur et de mots de passe. La bonne pratique, que nous recommandons vivement, est de toujorus utilsier HTTPS pour les serveurs de production et de ne jamais autoriser les connexions non chiffrées HTTP.

La configuration HTTPS de votre serveur Web est fonction du serveur utilisé. Veuillez consulter la documentation de votre serveur HTTP. Les exemples suivants s'appliquent à un serveur Apache.

Redirection de tout trafic non chiffré en HTTPS

Pour rediriger tout le trafic HTTP en HTTPS, les administrateurs sont encouragés à définir une redirection permanente en utilisant le code d'état 301. Avec Apache, ceci peut être accompli comme suit dans la configuration des hôtes virtuels d'Apache contenant l'entrée `<VirtualHost *:80>` :

```
Redirect permanent / https://exemple.com/
```

Activation de HTTP Strict Transport Security

Rediriger tout le trafic vers HTTPS est bien mais ne protège pas complètement des attaques man-in-the-middle. Par conséquent, les administrateurs sont encouragés à définir l'en-tête HTTP Strict Transport Security, qui indique aux

navigateurs de ne pas autoriser de connexion à l'instance ownCloud en utilisant HTTP, et essaient d'empêcher les visiteurs du site de passer outre les avertissements de certificats invalides.

Ceci peut être accompli en définissant les paramètres suivants dans la configuration de l'hôte virtuel d'Apache contenant l"entrée `<VirtualHost *:443>`

```
<IfModule mod_headers.c>
  Header always set Strict-Transport-Security "max-age=15768000; includeSubDomains"
</IfModule>
```

Si vous n'avez pas accès à votre fichier de configuration Apache, il est aussi possible d'ajouter ceci dans le fichier `.htaccess` principal livré avec ownCloud. Assurez-vous de l'ajouter sous la ligne suivante

```
#### DO NOT CHANGE ANYTHING ABOVE THIS LINE ####
```

Cet exemple de configuration rendra également les sous-domaines uniquement accessibles par HTTPS. Si vous avez des sous-domaines non accessibles par HTTPS, supprimez `includeSubdomains;`.

Note

Ceci nécessite l'activation du module Apache `mod_headers`.

Si vous utilisez nginx comme serveur Web, un exemple est déjà inclus dans *Exemples de configurations pour Nginx*

```
#add_header Strict-Transport-Security "max-age=15768000; includeSubDomains";
```

Vous devez retirer le caractère # et recharger nginx pour activer cette modification.

Configuration SSL correcte

Les configurations SSL par défaut des serveurs Web ne sont souvent pas optimales et nécessitent d'être ajustée pour des performances et une sécurité optimales. Les chiffrements SSL et les options dépendent compmlètement de votre environnement et par conséquent, une recommandation générique n'est pas possible.

Nous recommandons l'utilisation de Mozilla SSL Configuration Generator pour générer une configuration sécurisée adaptée à vos besoins et votre environnement, et l'outil gratuit Qualys SSL Labs Tests qui donne de bonnes indications sur la configuration de votre serveur SSL.

Assurez-vous également que la compression HTTP soint désactivée pour réduire le risque d'attaque BREACH.

Utilisation d'un domaine dédié pour ownCloud

Les administrateurs sont encouragés à installer ownCloud sur un domaine dédié comme par exemple « cloud.domaine.tld » au lieu de « domaine.tld » pour obtenir tous les bénéfices offerts par la Same-Origin-Policy (politique de même origine).

Installation de l'instance ownCloud dans une DMZ

Comme ownCloud gère des fonctionnalités tel que le partage de fichiers fédéré, nous ne considérons pas que Server Side Request Forgery (SSRF) fasse partie de notre modèle de menace. En fait, étant donné tous les connecteurs de stockage externes, cela peut être considéré comme une fonctionnalité et non une vulnérabilité.

Cela signifie qu'un utilisateur de votre instance ownCloud pourrait explorer si d'autres hôtes sont accessibles depuis le réseau ownCloud. Si vous ne voulez pas ceci, vous devez vous assurer que votre ownCloud est correctement installé dans un réseau distinct et avec des règles de pare-feu correctes.

Service d'en-têtes relatifs à la sécurité par le serveur Web

Les en-têtes de sécurité de base sont déjà servis par ownCloud dans un environnement par défaut. Ce qui inclut :

- `X-Content-Type-Options: nosniff`
    - Indique à certains navigateurs de ne pas renifler le type MIME des fichiers. Ceci est utilisé pour empêcher les navigateurs d'interprêter des fichiers texte comme du JavaScript.

- `X-XSS-Protection: 1; mode=block`
    - Indique aux navigateurs d'activer leurs propres filtres de Cross-Site-Scripting.
- `X-Robots-Tag: none`
    - Indique aux robots de recherche de ne pas indexer ces pages.
- `X-Frame-Options: SAMEORIGIN`
    - Empêche l'intégration de l'instance ownCloud dans un iframe d'un autre domaine pour empêcher le Clickjacking ou d'autres attaques similaires.

Ces en-têtes sont codés en dur dans le serveur ownCloud et ne nécessitent aucune intervention de la part des administrateurs.

Pour une sécurité optimale, les administrateurs sont encouragés à faire servir ces en-têtes de sécurité de base par leur serveur Web. Pour faire cela, Apache doit être configuré pour utiliser le fichier `.htaccess` et les modules Apache suivantes doivent être activés :

- mod_headers
- mod_env

Les administrateurs peuvent vérifier si ce changement de sécurité est actif en accédant à une ressource statique servie par le serveur Web et en vérifiant que les en-têtes de sécurité mentionnés plus haut soient bien présents.

Configuration de reverse proxy

ownCloud fonctionner derrière un reverse proxy, qui peut mettre en cache les éléments statiques comme les images et les fichiers JS ou CSS, déporter la charge du traitement HTTPS sur un serveur différent ou faire de l'équilibrage de sur plusieurs serveurs.

Définition de proxy de confiance

Pour des raisons de sécurité, vous devez définir explicitement les serveurs proxy auxquels ownCloud peut faire confiance. Les connexions aux serveurs proxy de confiance seront traitées de façon spéciale pour obtenir l'information réelle du client, utilisée pour le contrôle d'accès et la journalisation. Les paramètres sont configurés dans le fichier config/config.php.

Définissez le paramètre **trusted_proxies** comme un tableau d'adresses IP pour définir les serveurs proxy auxquels le serveur ownCloud peut faire confiance. Ce paramètre fournit une protection contre le spoofing client, et vous devez sécuriser ce serveurs comme vous le faites pour le serveur ownCloud.

Un reverse proxy peut définir des en-têtes HTTP avec l'adresse IP originale du client, et ownCloud peut utiliser ces en-têtes pour récupérer l'adresse IP. ownCloud utilise le standard de facto d'en-tête « X-Forwarded-For » par défaut, mais cela peut être configuré avec le paramètre **forwarded_for_headers**. Ce paramètre est un tableau de chaînes de recherches PHP, par exemple « X-Forwarded-For » devient « HTTP_X_FORWARDED_FOR ». Si ce paramètre est mal défini, il peut permettre à des clients de spoofer leur adresse IP telle qu'elle est vue par ownCloud, même s'ils passent par des proxy de confiance ! La valeur correcte pour ce paramètre dépend de votre logiciel de proxy.

Paramètres d'écrasement

La détection automatique du nom d'hôte, du protocole ou de la racine Web d'ownCloud peut échouer dans certaines situations. Cette configuration permet de passer outre la détection automatique pour être écrasée manuellement.

Si ownCloud échoue à détecter automatiquement le nom d'hôte, le protocole ou la racine Web, vous pouvez utiliser les paramètres **overwrite** du fichier config/config.php. Le paramètre **overwritehost** est utilisé pour définir le nom d'hôte du proxy. Vous pouvez également spécifier un port. Le paramètre **overwriteprotocol** est utilisé pour définir le protocole du proxy. Vous avez le choix entre deux options : **http** et **https**. Le paramètre **overwritewebroot** est utilisé pour définir le chemin absolu de la racine Web du proxy au dossier d'ownCloud. Si vous voulez conserver la détection automatique de l'un de ces trois paramètres,vous pouvez laisser la valeur vide. Le paramètre **overwritecondaddr** est utilisé pour écraser les valeurs dépendant de l'adresse distante. La valeur La valeur doit être une **expression régulière** des adresses IP du proxy. C'est utile quand vous utilisez un reverse proxy SSL uniquement pour les accès https et que vous voulez utiliser la détection automatique pour les accès http.

Exemple

Reverse proxy SSL multi-domaine

Si vous voulez accéder à votre installation ownCloud **http://domaine.tld/owncloud** par l'intermédiaire de reverse proxy SSL multi-domaine **https://ssl-proxy.tld/domaine.tld/owncloud** avec l'adresse IP **10.0.0.1** vous pouvez définir les paramètres suivants dans le fichier config/config.php.

```
<?php
$CONFIG = array (
  "trusted_proxies"   => ['10.0.0.1'],
  "overwritehost"     => "ssl-proxy.tld",
  "overwriteprotocol" => "https",
  "overwritewebroot"  => "/domaine.tld/owncloud",
  "overwritecondaddr" => "^10\.0\.0\.1$",
);
```

> ***Note***
>
> Si vous voulez utiliser le proxy SSL pendant l'installation, vous devez créer auparavant le fichier config/config.php, sans quoi, vous devrez étendre le tableau **$CONFIG** existant.

Utilisation de composants PHP tiers

ownCloud utilise quelques composants PHP tiers ppour proposer certaines de ses fonctionnalités. Ces composants font partie du logiciel et sont situés dans le dossier **/3rdparty**.

Gestion des paramètres tiers

Lors de l'utilisation de composants tiers, gardez les paramètres suivants à l'esprit :

- **3rdpartyroot** -- Spécifie l'emplacement du dossier 3rd-party. Pour modifier l'emplacement par défaut de ce dossier, vous pouvez utiliser ce paramètre pour définir le chemin absolu pour l'emplacement de ce dossier.
- **3rdpartyurl** -- Spécifie le chemin web HTTP du dossier 3rdpartyroot, débutant à la racine Web d'ownCloud.

Ci-après, un exemple de configuration de ces paramètres :

```
<?php

"3rdpartyroot" => OC::$SERVERROOT."/3rdparty",
"3rdpartyurl"  => "/3rdparty",
```

JavaScript et gestion de CSS

Dans les environnements de production, les fichiers JavaScript et CSS doivent être délivrés sous forme concaténée et compressée.

ownCloud peut automatiquement collecter tous les fichiers JavaScript et CSS, les agréger et les compresser pour enregistrer le résultats dans un dossier nommé « assets » qui se trouve dans le dossier où a été installé ownCloud.

Si votre serveur Web a des droits en écriture sur le répertoire d'installation d'ownCloud, alors le dossiers « assets » sera automatiquement créé pour vous. Sinon, vous devrez le crtéer vous-même avant d'activer cette option et vous devrez donner les droits en écriture sur ce répertoire au serveur Web.

Les fichiers dans ce dossier seront dès lors servis comme des fichiers statiques par votre serveur Web et seront automatiquement réactualisés quand ownCloud ou une de ses applications est mise à jour. Il est important de noter que les applications installées via git pourraient ne pas toujours mettre à jour leur numéro de version avec chaque commit et cela pourrait conduire à un répertoire asset désynchronisé. Il n'est pas recommandé d'activer « asset-pipeline » lors de l'utilisation d'applications mises à jour via git.

Paramètres

```
<?php
  $CONFIG = array (
    ...
    'asset-pipeline.enabled' => true,
    ...
  );
```

Vous pouvez définir ce paramètre dans le fichier config/config.php

Configuration d'installation automatique

Si vous avez besoin d'installer ownCloud sur plusieurs serveurs, vous ne voudrez sûrement pas paramétrer chaque instance séparément comme décrit dans *Configuration de la base de données*. Pour cette raison, ownCloud propose une fonctionnalité de configuration automatique.

Pour tirer avantage de cette fonctionnalité, vous devez créer un fichier de configuration, nommé ../owncloud/config/autoconfig.php et définir les paramètres du fichier. Vous pouvez spécifier tous les paramètres nécessaires dans ce fichier. Tous les paramètres non définis apparaîtront dans l'écran « Terminer l'installation » au premier lancement d'ownCloud.

Le fichier ../owncloud/config/autoconfig.php est automatiquement supprimé après l'application de la configuration initiale.

Paramètres

Lors de la configuration des paramètres, vous devez comprendre que deux paramètres sont nommés différemment dans ce fichier de configuration comparé au fichier standard config.php.

autoconfig.php	config.php
directory	datadirectory
dbpass	dbpassword

Exemples de configurations automatiques

Les sections suivantes proposent des exemples de configurations automatiques et indiquent quelles informations seront demandées à la fin de la configuration.

Répertoire data

En utilisant les paramètres suivants, l'écran « Terminer l'installation » demandera le nom de la base de données et les identifiants de l'administrateur.

```
<?php
$AUTOCONFIG = array(
  "directory"     => "/www/htdocs/owncloud/data",
);
```

Base de données SQLite

En utilisant les paramètres suivants, l'écran « Terminer l'installation » demandera le chemin du répertoire data et les identifiants de l'administrateur.

```
<?php
$AUTOCONFIG = array(
  "dbtype"        => "sqlite",
  "dbname"        => "owncloud",
  "dbtableprefix" => "",
);
```

Base de données MySQL

En utilisant les paramètres suivants, l'écran « Terminer l'installation » demandera le chemin du répertoire data et les identifiants de l'administrateur.

```
<?php
$AUTOCONFIG = array(
  "dbtype"        => "mysql",
  "dbname"        => "owncloud",
  "dbuser"        => "utilisateur",
  "dbpass"        => "mot_de_passe",
  "dbhost"        => "nom_hote_local",
  "dbtableprefix" => "",
);
```

Note

Gardez à l'esprit que la configuration automatique ne dispense pas de créer la base de données et l'utilisateur de la base de données au préalable comme indiqué dans *Configuration de la base de données*.

Base de données PostgreSQL

En utilisant les paramètres suivants, l'écran « Terminer l'installation » demandera le chemin du répertoire data et les identifiants de l'administrateur.

```
<?php
$AUTOCONFIG = array(
  "dbtype"        => "pgsql",
  "dbname"        => "owncloud",
  "dbuser"        => "utilisateur",
  "dbpass"        => "mot_de_passe",
  "dbhost"        => "nom_hote_local",
  "dbtableprefix" => "",
);
```

Note

Gardez à l'esprit que la configuration automatique ne dispense pas de créer la base de données et l'utilisateur de la base de données au préalable comme indiqué dans *Configuration de la base de données*.

Tous les paramètres

En utilisant les paramètres suivants, car tous les paramètres sont déjà configurés dans le fichier, l'installation d'ownCloud ne proposera pas l'écran « Terminer l'installation » en fin d'installation

```
<?php
$AUTOCONFIG = array(
  "dbtype"        => "mysql",
  "dbname"        => "owncloud",
  "dbuser"        => "utilisateur",
  "dbpass"        => "mot_de_passe",
  "dbhost"        => "nom_hote_local",
  "dbtableprefix" => "",
  "adminlogin"    => "root",
  "adminpass"     => "mot_de_passe_root",
  "directory"     => "/www/htdocs/owncloud/data",
);
```

> **Note**
>
> Gardez à l'esprit que la configuration automatique ne dispense pas de créer la base de données et l'utilisateur de la base de données au préalable comme indiqué dans *Configuration de la base de données*.

Mise au point du serveur ownCloud

Utilisation de cron pour réaliser des tâches d'arrière-plan

Consulter *Tâches d'arrière-plan* pour une description et ses bénéfices.

Activation de la gestion de JavaScript et CSS

Consulter See *JavaScript et gestion de CSS* pour une description et ses bénéfices.

Cache

La mise en cache améliore les performances en stockant les données, le code et d'autres objets en mémoire. La configuration de la mémoire cache pour le serveur ownCloud n'est plus automatique depuis la version 8.1 d'ownCloud, mais doit être installée et configurée. Consulter *Configuration de la mémoire cache*.

Utilisation de MariaDB/MySQL au lieu de SQLite

MySQL ou MariaDB sont préférés en raison des limitations de performances de SQLite avec les applications ayant beaucoup d'accès concurrents, comme ownCloud.

Voir la section *Configuration de la base de données* pour savoir comment configurer ownCloud pour MySQL ou MariaDB. Si votre installation exécute déjà SQLite, il est alors possible de faire la conversion vers MySQL ou MariaDB en suivant les étapes indiquées dans *Conversion du moteur de base de données*.

Utilisation du verrouillage de fichier transactionnel basé sur Redis

Le verrouillage de fichiers est activé par défaut, en utilisant le service de verrouillage de la base de données. Ceci provoque une charge significative sur la base de données. Consulter la section *Verrouillage de fichier transactionnel* pour savoir comment configurer ownCloud pour utiliser le verrouillage de fichier transactionnel basé sur Redis.

SSL et application Encryption

SSL (HTTPS) et le chiffrement/déchiffrement de fichiers peuvent être déportés sur l'extension AES-NI d'un processeur. Ceci peut à la fois accélérer ces opérations tout en limitant la charge de traitement. Ceci nécessite un processeur avec le jeu d'instructions AES-NI.

Voici quelques exemples pour vérifier si votre processeur/environnement gère l'extension AES-NI :

- Pour chaque cœur du processeur : `grep flags /proc/cpuinfo` ou pour un résumé de tous les cœurs : `grep -m 1 ^flags /proc/cpuinfo`. Si le résultat contient `aes`, l'extension est présente.
- Recherchez par exemple sur le site Web d'Intel utilisé gère l'extension Intel Processor Feature Filter Vous pouvez définir un filtre avec `"AES New Instructions"` pour limiter le nombre de résultats.
- Pour les versions d'openssl >= 1.0.1, AES-NI ne fonctionne pas via un moteur et n'apparaîtra pas avec la commande `openssl engine`. Il est actif par sur le matériel pouvant le gérer. Vous pouvez vérifier la version d'openssl avec la commande``openssl version -a``
- Si votre processeur gère AES-NI mais qu'il n'apparaît pas par exemple avec grep ou coreinfo, il est peut-être désactivé dans le BIOS.
- Si votre environnement est virtualisé, vérifiez avec le support du vendeur.

Activer les URL sans index.php

Depuis ownCloud 9.0.3, vous devez explicitement configurer et activer les URL sans index.php (par exemple https://exemple.com/apps/files/ au lieu de https://exemple.com/index.php/apps/files/). La documentation suivantes fournit les indications pour configurer ceci pour le serveur Web `Apache`. Ces étapes ne sont pas nécessaires si vous utilisez le serveur Web nginx, car ceci est déjà activé dans la *Exemples de configurations pour Nginx*.

Prérequis

Avant de pouvoir utiliser les URL sans index.php, vous devez activer les modules Apache `mod_rewrite` et `mod_env`. De plus, la directive `AllowOverride All` dans le vhost de votre serveur Web est nécessaire. Veuillez consulter le manuel `Apache` pour savoir comment activer et configurer cela.

De plus, ces instructions ne fonctionnent qu'en utilisant Apache avec le module Apache pour PHP `mod_php`. D'autres modules comme `php-fpm` ou `mod_fastcgi` ne sont pas supportés.

Enfin, l'utilisateur exécutant votre serveur Web (par ex. : `www-data`) doit pouvoir écrire dans le fichier `.htaccess` fourni à la racine du répertoire ownCloud (par ex. : `/var/www/owncloud/.htaccess`). Si vous avez appliqué *Renforcement des permissions de répertoires* l'utilisateur pourrait ne pas pouvoir écrire dans ce fichier et la mise à jour nécessaire échouera. Vous devez enlever temporairement ces permissions en suivant les instructions décrites dans *Permissions pour la mise à jour*.

Étapes de configuration

La première étape est de configurer les options `overwrite.cli.url` et `htaccess.RewriteBase` dans le fichier config.php (voir *Paramètres de Config.php*). Si vous accéder à votre instance ownCloud via `https://exemple.com/`, les deux options suivantes doivent être configurées :

```
'overwrite.cli.url' => 'https://exemple.com',
'htaccess.RewriteBase' => '/',
```

Si l'instance est accédée via `https://exemple.com/owncloud`, l'option de configuration suivante est nécessaire :

```
'overwrite.cli.url' => 'https://exemple.com/owncloud',
'htaccess.RewriteBase' => '/owncloud',
```

La seconde étape est d'activer dans ownCloud les URL sans index.php. Ceci est réalisé :

- lors de la prochaine mise à jour de votre instance ownCloud ;
- en lançant manuellement la commande occ `occ maintenance:update:htaccess` (voir *Utilisation de la commande occ*).

Après cela, votre instance ne devrait plus avoir d'URL avec index.php.

Dépannage

Si l'accès à votre installation d'ownCloud échoue après avoir suivi ces instructions et que vous voyez des messages comme ceci dans votre fichier journal d'ownCloud :

```
The requested uri(\\/login) cannot be processed by the script '\\/owncloud\\/index.php'
```

assurez-vous d'avoir configuré correctement les deux options dans votre fichier `config.php` comme indiqué précédemment.

Gestion des utilisateurs

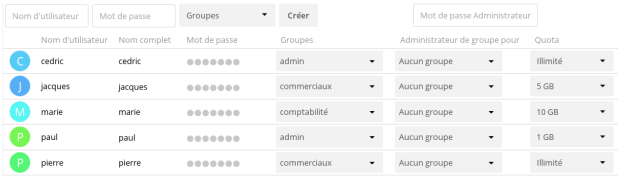
Gestion des utilisateurs

Sur la page ge gestion des utlisateurs d'ownCloud, vous pouvez :

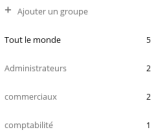
- créer de nouveaux utilisateurs ;
- voir tous les utilisateurs dans une fenêtre unique munie d'un ascenseur ;

- filtrer les utilisateurs par groupe ;
- voir à quels groupe ils appartiennent ;
- modifier leurs noms complets et leurs mots de passe ;
- voir l'emplacement du stockage de leurs données ;
- voir et définir les quotas ;
- créer et modifier leurs adresses électroniques ;
- envoyer un courriel de notification automatique aux nouveaux utilisateurs ;
- les supprimer d'un simpmle clic.

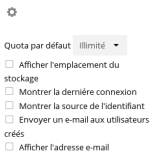
La vue par défaut affiche les informations de base de vos utilisateurs.



Le filtres de groupes sur la barre latérale de gauche permet de filtrer rapidement les utilisateurs en fonction de leur appartenance à un groupe et de créer de nouveaux groupes.



Cliquez sur l'icône représentant un engrenage dans le coin inférieur gauche de l'écran pour définir un quota de stockage par défaut et pour afficher des champs supplémentaires : **Afficher l'emplacement du stockage**, **Montrer la dernière connexion**, **Montrer la source de l'identifiant**, **Envoyer un e-mail aux utilisateurs créés** et **Afficher l'adresse e-mail**.



Les comptes utilisateurs ont les propriétés suivantes :

Nom d'utilisateur

L'identifiant unique d'un utilisateur ownCloud. Il ne peut être modifié.

Nom complet

Le nom utilisateur qui sera affiché sur les partages de fichiers, l'interface Web d'ownCloud et les courriels. Les administrateurs et les utilisateurs peuvent changer à tout moment leur *Nom complet*. Si celui-ci n'est pas défini, ce sera par défaut le *Nom d'utilisateur*.

Mot de passe

L'administrateur définit le premier mot de passe du nouvel utilisateur. L'administrateur et l'utilisateur peuvent à tout moment changer le mot de passe de l'utilisateur.

Groupes

Vous pouvez créer des groupes et y associer des utilisateurs. Par défaut, les nouveaux utilisateurs ne sont affectés à aucun groupe.

Administrateur de groupe pour

Les administrateurs de groupe ont des privilèges administratifs sur des groupes spécifiques et peuvent ajouter ou supprimer des utilisateurs de leurs groupes.

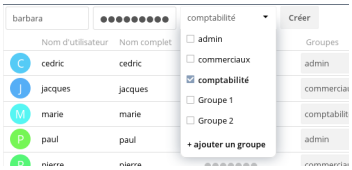
Quota

La quantité maximale de disque alloué à chaque utilisateur. Tout utilisateur dépassant le quota ne pourra plus téléverser de fichier ou synchroniser ses données. Vous pouvez ajouter les stockages externes dans le calcul du quota.

Création d'un nouvel utilisateur

Pour créer un nouveau compte :

- saisir le **Nom d'utilisateur** du nouvel utilisateur et son **Mot de passe** initial ;
- l'affecter à des **Groupes** (facultatif) ;
- cliquer sur le bouton **Créer**.



Les **Nom d'utilisateur** peuvent contenir des lettres (a-z, A-Z), des chiffres (0-9), des tirets (-), des traits de soulignement (_), des points (.) et le symbole (@). Après la création de l'utilisateur, vous pouvez renseigner son **Nom complet** s'il est différent de son nom d'utilisateur ou laisser le choix à l'utilisateur de le saisir.

Si vous avez coché la case **Envoyer un e-mail aux utilisateurs créés** dans le panneau de contrôle danss le coin inférieur gauche de l'écran, vous pouvez aussi saisir l'adresse électronique de l'utilisateur et ownCloud lui enverra automatiquement une notification avec les informations de connexion. Vous pouvez modifier ce courriel en utilisant l'éditeur de modèle de courriel sur votre page d'administration (voir *Configuration des courriels*).

Réinitialisation du mot de passe d'un utilisateur

Vous ne pouvez récupérer le mot de passe d'un utilisateur, mais vous pouvez en définir un nouveau :

- passez le curseur de la souris au-dessus du champ **Mot de passe** de l'utilisateur ;
- cliquez sur l'icône représentant un crayon ;
- saisissez le nouveau mot de passe de l'utilisateur et transmettez le nouveau mot de passe à l'utilisateur.

Si le chiffrement est activé, il faut prendre en comte d'autres considérations pour la réinitialisation du mot de passe. Veuillez consulter *Configuration du chiffrement*.

Renommage d'un utilisateur

Chaque utilisateur ownCloud a deux noms : le **Nom d'utilisateur** (unique) qui est utilisé pour l'authentification, et le **Nom complet**, qui est le nom d'affichage. Vous pouvez modifier le nom d'affichage d'un utilisateur mais pas son **Nom d'utilisateur**.

Pour définir ou modifier le nom d'affichage :

- passez le curseur de la souris au-dessus du champ **Nom complet** de l'utilisateur ;
- cliquez sur l'icône représentant un crayon ;
- saisissez le nouveau nom d'affichage.

Privilèges administratifs pour un utilisateur

ownCloud a deux types d'administrateurs : les **Super administrateurs** et les **Administrateurs de groupe**. Les administrateurs de groupe ont le droit de créer, modifier et supprimer les utilisateurs de leurs groupes. Ils ne peuvent pas accéder aux paramètres système, ni ajouter ou modifier des utilisateurs dans les groupes pour lesquels ils ne sont pas **Administrateurs de groupe**. Utilisez les menus déroulants dans la colonne **Administrateur de groupe pour** donner des privilèges d'administration de groupe.

Les **Super administrateurs** ont tous les droits sur le serveur ownCloud et peuvent modifier tous les paramètres. Pour affecter le rôle **Super administrateur** à un utilisateur, il suffit de l'ajouter au groupe `admin`.

Gestion des groupes

Vous pouvez affecter des utilisateurs à des groupes lors de leur création et créer de nouveaux groupes quand vous créez de nouveaux utilisateurs. Vous pouvez aussi utiliser le bouton **Ajouter un groupe** dans le coin supérieur gauche de l'écran pour créer de nouveaux groupes. Les nouveaux membres de groupes auront immédiatement accès aux partages de leurs groupes.

Définition des quotas de stockage

Cliquez sur l'icône représentant un engrenage dans le coin inférieur gauche de l'écran pour définir le quota de stockage par défaut. Ceci s'applique immédiatement pour les nouveaux utilisateurs. Vous pouvez affecter un quota différent pour un utilisateur spécifique en sélectionnant la liste déroulante **Quota**, et en sélectionnant soit une valeur pré-enregistrée soit en saisissant une valeur personnalisée. Lors de la création de quotas personnalisés, utiliser les abréviations normales comme 500 MB, 5 GB, 5 TB, etc.

Vous disposez maintenant d'une option dans le fichier `config.php` qui permet d'indiquer si le stockage externe doit être pris en compte dans le calcul du quota utilisateur. Ceci est expérimental et pourrait ne pas fonctionner comme prévu. Par défaut, le stockage externe n'est âs pris en compte. Si vous voulez l'inclure, changez la valeur `false` pour `true`.:

```
'quota_include_external_storage' => false,
```

Les méta-données (comme les vignettes, les fichiers temporaires et les clés de chiffrement) prennent environ 10 % de l'espace disque et ne sont pas prises en compte dans le calcul du quota. Les utilisateurs peuvent vérifier leur utilisation et l'espace disponible sur leur page personnelle. Les fichiers partagés par d'autres utilisateurs ne sont pas pris en compte dans le quota. Par exemple, si vous téléversez un fichier sur le partage d'un autre utilisateur, ce fichier ne sera pas pris en compte dans le calcul de votre quota. Si vous re-partagez un fichier qui a été partagé avec vous, ce fichier n'est pas pris en compte dans le calcul de votre quota mais dans celui de l'utilisateur qui a initialement partagé le fichier.

Les fichiers chiffrés sont un peu plus gros que les fichiers non chiffrés. C'est la taille du fichier non chiffré qui est prise en compte dans le calcul du quota.

Les fichiers supprimés qui se trouvent dans la corbeille ne sont pas pris en compte dans le calcul du quota. La corbeille est définie à 50 % du quota. La rétention des fichiers supprimés est définie à 30 jours. Quand la taille des fichiers supprimés excédent 50 % du quota, les fichiers les plus anciens sont supprimés de la corbeille jusqu'à ce que la taille totale repasse en dessous de 50 %.

Quand le contrôle de version est activé, les plus anciennes versions de fichiers ne sont pas prises en compte dans le calcul du quota.

Quand un utilisateur crée un partage public via une URL et autorise les téléversements, les fichiers téléversés sont pris en compte dans le caclcul du quota.

Suppression d'utilisateurs

La suppression d'un utilisateur est simple : passez le curseur de la souris au-dessus du nom de l'utilisateur sur la page **Utilisateurs** jusqu'à ce qu'apparaisse à l'extrême droite une icône représentant une corbeille. Cliquez sur l'icône et c'est fini. Un bouton d'annulation apparaît en haut de la page. Celui-ci reste présent jusqu'à l'actualisation de la page. Quand le bouton d'annulation a disparu, il n'est plus possible de restaurer l'utilisateur.

Tous les fichiers détenus par l'utilisateur sont également supprimés, y compris les fichiers qui ont été partagés. Si vous devez conserver les fichiers et les partages de l'utilisateur, vous devez d'abord les télécharger à partir de la page Fichiers downCloud, qui les compresse dans un fichier zip, ou utiliser un client de synchronisation pour les copier sur votre ordinateur local. Consultez *Partage de fichiers* pour en apprendre plus sur la création de partage de fichiers persistants qui survivent à la suppression d'un utilisateur.

Réinitialisation d'un mot de passe administrateur perdu

Les deux moyens normaux de réinitialiser un mot de passe sont :

1. Cliquer sur le lien de réinitialisation de mot de passe sur l'écran de connexion. Celui-ci apparaît après un échec de connexion. Ceci ne fonctionne que si vous avez saisi votre adresse électronique sur votre page personnelle dans l'interface Web, pour qu'ownCloud puisse vous envoyer un courriel avec le lien de réinitialisation.
2. Demander à un autre administrateur de réinitialiser le mot de passe pour vous.

Si aucune de ces deux méthodes n'est envisageable, il existe une troisième option qui est d'utiliser la commande `occ`. `occ` se situe dans le répertoire `owncloud`, par exemple `/var/www/owncloud/occ`. `occ` a une commande permettant de réintialiser les mots de passe de tous les utilisateurs, `user:resetpassword`. Il est préférable de lancer `occ` en tant qu'utilisateur HTTP, comme dans cet exemple pour Ubuntu Linux:

```
$ sudo -u www-data php /var/www/owncloud/occ user:resetpassword admin
Enter a new password:
Confirm the new password:
Successfully reset password for admin
```

Si votre nom d'utilisateur ownCloud n'est pas `admin`, substituez-le par votre nom utilisateur dans cette commande.

Vous pouvez trouver votre utilisateur HTTP dans le fichier de configuration HTTP. Voici les valeurs par défaut pour l'utilisateur et le groupe HTTP pour les distributions Linux :

- Centos, Red Hat, Fedora : apache:apache
- Debian, Ubuntu, Linux Mint : www-data:www-data
- openSUSE : wwwrun:www

Consulter *Utilisation de la commande occ* pour en apprendre plus sur l'utilisation de la commande `occ`.

Réinitialisation d'un mot de passe utilisateur

L'écran de connexion d'ownCloud affiche un message **Mot de passe incorrect. Réinitialiser ?** après un échec de connexion et ownCloud peut alors réinitialiser automatiquement le mot de passe. Cependant, si vous utilisez une authentification en lecture seule, comme LDAP ou Active Directory, ceci ne fonctionne pas. Dans ce cas, vous devez spécifier une URL personnalisée dans le fichier `config.php` pour diriger l'utilisateur vers un serveur qui peut gérer la réinitialisation de mot de passe automatique:

```
'lost_password_link' => 'https://exemple.org/lien/vers/réinitialisation',
```

Authentification utilisateur avec IMAP, SMB et FTP

Vous pouvez configurer des services de gestion utilisateurs additionnels dans le fichier de configuration d'ownCloud config/config.php en utilisant la syntaxe suivante :

```
<?php

"user_backends" => array (
    0 => array (
            "class"     => ...,
            "arguments" => array (
                                0 => ...
                                ),
    ),
),
```

Note

Une configuration correcte et non bloquante de SELinux est nécessaire pour que ces services puissent fonctionner. Veuillez consulter *Configuration SELinux*.

Actuellement, l'application « External user support » (user_external), qui doit être activéer en premier (voir *Installation et gestion des applications*) fournit les services suivants :

IMAP

Fournit l'authentification sur des serveurs IMAP.

- **Classe** : OC_User_IMAP
- **Arguments** : une chaîne de boîte à lettres tel que défini dans la documentation PHP
- **Dépendances** : php-imap (voir *Manuel d'installation pour Linux*)
- **Exemple**

```
<?php

"user_backends" => array (
    0 => array (
            "class"     => "OC_User_IMAP",
            "arguments" => array (
                               0 => '{imap.gmail.com:993/imap/ssl}'
                               ),
    ),
),
```

SMB

Fournit l'authentification sur des serveurs Samba.

- **Classe** : OC_User_SMB
- **Arguments** : le serveur Samba sur lequel s'authentifier
- **Dépendances** : le module PHP smbclient ou smbclient (voir *SMB/CIFS*)
- **Exemple:**

```
<?php

"user_backends" => array (
    0 => array (
            "class"     => "OC_User_SMB",
            "arguments" => array (
                               0 => 'localhost'
                               ),
    ),
),
```

FTP

Fournit l'authentification sur des serveurs FTP.

- **Classe** : OC_User_FTP
- **Arguments** : le serveur FTP sur lequel s'authentifier
- **Dépendances** : php-ftp (voir *Manuel d'installation pour Linux*)
- **Exemple**

```
<?php

"user_backends" => array (
    0 => array (
            "class"     => "OC_User_FTP",
            "arguments" => array (
                               0 => 'localhost'
```

```
                                ),
    ),
),
```

Authentification utilisateur avec LDAP

ownCloud fournit une application LDAP permettant aux utilisateurs LDAP (y compris Active Directory) d'apparaître dans le listes d'utilisateurs d'ownCloud. Ces utilisateurs pourront s'authentifier sur ownCloud avec leurs identifiants LDAP, vous n'avez donc pas à créer de comptes ownCloud séparés pour ceux-ci. Vous pourrez gérer leur appartenance aux groupes, leur quota et leur permissions de partage comme n'importe quel autre utilisateur d'ownCloud.

Note

Le module PHP LDAP est requis. Il est fourni par le paquet `php5-ldap` sur Debian/Ubuntu, et `php-ldap` sur CentOS/Red Hat/Fedora. PHP 5.4+ est nécessaire pour ownCloud 8.1.

L'application LDAP gère :

- les groupes LDAP ;
- le partage de fichiers avec les utilisateurs et groupes d'ownCloud ;
- l'accès via WebDAV et les clients de synchronisation ownCloud ;
- le versionnement, le stockage externe et les autres fonctionnalités d'ownCloud ;
- la connexion transparente à Active Directory, sans configuration supplémentaire ;
- les groupes primaires dans Active Directory ;
- la détection automatique des attributs LDAP tels que la base DN, l'adresse électronique et le numéro de port du serveur LDAP ;
- l'accès en lecture seule sur votre serveur LDAP (la modification ou la suppression des utilisateurs sur votre serveur LDAP n'est pas gérée).

Warning

L'application LDAP n'est pas compatible avec l'application `User backend using remote HTTP servers`. Vous ne pouvez pas utiliser les deux en même temps.

Note

Une configuration correcte et non bloquante de SELinux est nécessaire pour que le service LDAP fonctionne. Veuillez consulter *Configuration SELinux*.

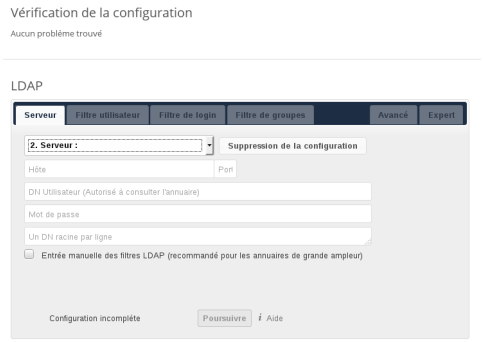
Configuration

Vous devez tout d'abord activer l'application `LDAP user and group backend` sur la page Applications d'ownCloud. Puis, rendez-vous sur la page d'administration pour la configurer.

Le panneau de configuration LDAP dispose de quatre onglets. Le premier onglet (« Serveur ») doit être correctement configuré pour pouvoir accéder aux autres onglets. Un indicateur vert apparaît quand la configuration est correcte. Passez le curseur de la souris au-dessus des champs du formulaire pour voir apparaître des info-bulles d'aide.

Onglet serveur

Commencez par l'onglet « Serveur ». Vous pouvez configurer plusieurs serveurs. Vous devez au minimum fournir le nom d'hôte du serveur LDAP. Si votre serveur nécessite une authentification, saisissez les identifiants sur cet onglet. ownCloud essaiera de détecter automatiquement le numéro de port du serveur et la base DN. La base DN et le numéro de port sont obligatoires, donc si ownCloud ne peut les détecter, vous devrez les saisir manuellement.



Configuration du serveur :

Configurez un ou plusieurs serveurs LDAP. Cliquez sur le bouton **Suppression de la configuration** pour supprimer la configuration en cours.

Hôte :

Le nom d'hôte ou l'adresse IP du serveur LDAP. Cela peut être aussi une URI **ldaps://** . Si vous saisissez le numéro de port, la détection du serveur sera plus rapide.

Exemples :

- *annuaire.ma-societe.com*
- *ldaps://annuaire.ma-societe.com*
- *annuaire.ma-societe.com:9876*

Port :

Le port sur lequel se connecter au serveur LDAP. Le champ est désactivé au commencement d'une nouvelle configuration. Si le serveur LDAP fonctionne sur un port standard, le port sera détecté automatiquement. Si vous utilisez un port non standard, ownCloud essaiera de le détecter. S'il échoue, vous devrez le saisir manuellement.

Exemple:

- *389*

DN utilisateur :

Le DN de l'utilisateur qui a les droits de faire des recherches sur l'annuaire LDAP. Laissez ceci vide pour l'accès anonyme. Nous recommandons d'avoir un utilisateur spécial pour faire ces recherches.

Exemple :

- *uid=owncloudsystemuser,cn=sysusers,dc=ma-societe,dc=com*

Mot de passe :

Le mot de passe de l'utilisateur ci-dessus. Laissez vide pour l'accès anonyme.

Base DN :

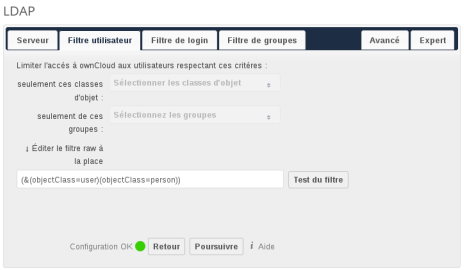
Le DN de la racine du serveur LDAP, à partir de laquelle les utilisateurs et les groupes seront recherchés. Vous pouvez saisir plusieurs racines, une par ligne. (Les DN des racines des utilisateurs et des groupes peuvent être définis dans l'onglet « Avancé »). Ce champ est obligatoire. ownCloud essaie de déterminer la base DN en fonction du DN utilisateur fourni ou à partir de l'hôte. Vous devrez le saisir manuellement si ownCloud n'arrive pas à la détecter.

Exemple :

- *dc=ma-societe,dc=com*

Filtre utilisateur

Utilisez ceci pour contrôler quels utilisateurs LDAP seront affichés comme utilisateurs ownCloud sur votre serveur ownCloud. Les utilisateurs LDAP qui ont un accès mais ne sont pas listés comme utilisateurs (s'il en existe) seront des utilisateurs cachés. Vous pouvez passer outre les champs du formulaire et saisir le flitre directement.



seulement ces classes d'objets :

ownCloud déterminera les classes d'objets typiquement disponibles pour les objets utilisateurs dans votre annuaire LDAP. ownCloud sélectionnera automatiquement la classe d'objet renvoyant le plus grand nombre d'utilisateurs. Vous pouvez saisir plusieurs classes d'objets.

seulement de ces groupes :

Si votre serveur LDAP gère `member-of-overlay` dans les filtres LDAP, vous pouvez définir que seuls les utilisateurs d'un ou plusieurs groupes sont autorisés à apparaître dans les listes d'utilisateurs d'ownCloud. Par défaut, aucune valeur n'est sélectionnée. Vous pouvez sélectionner plusieurs groupes.

Si votre serveur LDAP ne gère pas `member-of-overlay` dans les filtres LDAP, le champ est désactivé. Veuillez contacter votre administrateur LDAP.

Éditer le filtre raw à la place :

Cliquer sur ce lien permet de saisir le filtre directement. Exemple:

```
(&(objectClass=inetOrgPerson)(memberOf=cn=owncloudusers,ou=groups,
dc=ma-societe,dc=com))
```

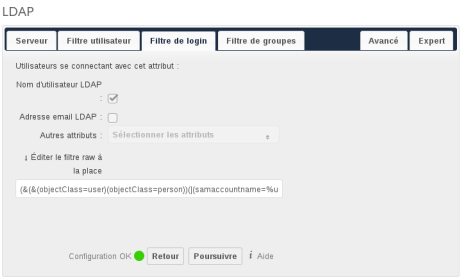
x utilisateurs trouvés :

Ceci est un indicateur fournissant le nombre approximatif d'utilisateurs qui seront listés dans ownCloud. Ce nombre se met à jour automatiquement après chaque modification du filtre.

Filtre de login

Les paramètres du filtre de login déterminent quels utilisateurs LDAP peuvent se connecter à ownCloud et quels attributs correspondront au compte de connexion (par exemple le nom d'utilisateur LDAP/AD, l'adresse électronique). Vous pouvez sélectionner plusieurs attributs. Vous pouvez également passer outre les champs du formulaire et saisir le filtre directement si vous le voulez.

Vous pouvez passer outre les paramètres définis dans l'onglet « Filtre utilisateur » en saisissant le filtre directement.



Nom d'utilisateur LDAP :

Si cette case est cochée, la valeur du login sera comparée au noom d'utilisateur dans l'annuaire LDAP. L'attribut correspondant qui est généralement *uid* ou *samaccountname* sera détecté automatiquement par ownCloud.

Adresse email LDAP :

Si cette case est cochée, la valeur du login sera comparée à une adresse électronique dans l'annuaire LDAP. Plus spécifiquement, les attributs *mailPrimaryAddress* et *mail*.

Autres attributs :

Cette liste à sélection multiple permet de sélectionner d'autres attributes pour la comparaison. La liste est générée automatiquement à partir des attributs de l'objet utilisateur dans votre annuaire LDAP.

Éditer le filtre raw à la place :

Cliquer sur ce lien permet de saisir le filtre directement.

Le terme privilégié (placeholder) **%uid** est remplacé par le nom de login saisi par l'utilisateur lors de la connexion.

Exemples :

- only username:

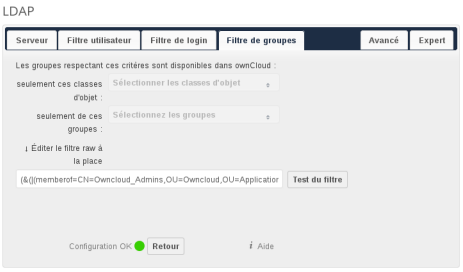
```
(&(objectClass=inetOrgPerson)(memberOf=cn=owncloudusers,ou=groups,
dc=ma-societe,dc=com)(uid=%uid)
```

- username or email address:

```
((&(objectClass=inetOrgPerson)(memberOf=cn=owncloudusers,ou=groups,
dc=ma-societe,dc=com)(|(uid=%uid)(mail=%uid)))
```

Filtres de groupes

Par défaut, aucun groupe LDAP n'est disponible dans ownCloud. Les paramètres de cet onglet détermineront quels groupes seront disponibles dans ownCloud.



seulement ces classes d'objet :

ownCloud déterminra quelles classes d'objet sont typiquement disponibles pour les objets groupes dans votre serveur LDAP. ownCloud ne listera que les classes d'objets qui renvoient au moins un objet groupe. Vous pouvez sélectionner plusieurs classes d'objets. Les classes d'objets typiques sont « group » et « posixGroup ».

seulement de ces groupes :

ownCloud générera une liste des groupes disponibles trouvés dans votre serveur LDAP. Vous pourrez alors sélectionner le ou les groupes qui pourront accéder à votre serveur ownCloud.

Éditer le filtre raw à la place :

Cliquer sur ce lien permet de saisir le filtre directement.

Exemple:

- *objectClass=group*
- *objectClass=posixGroup*

x groupes trouvés :

Ceci est un indicateur fournissant le nombre approximatif de groupes qui seront listés dans ownCloud. Ce nombre se met à jour automatiquement après chaque modification du filtre.

Avancé

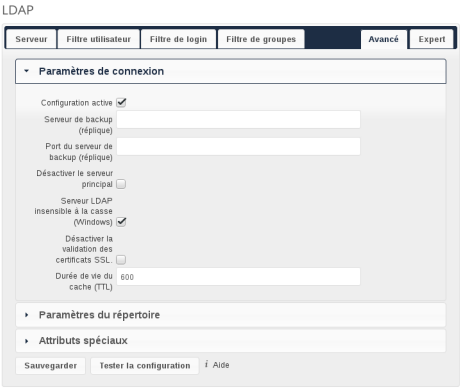
L'onglet « Avancé » contient des options qui ne sont pas obligatoires pour avoir une connexion opérationnelle. Il fournit des contrôle pour désactiver la configuration, configurer des serveurs de réplique et diverses options pour améliorer les performances.

Les paramètres avancés sont structurés en trois parties :

- Paramètres de connexion

- Paramètres du répertoire
- Attribus spéciaux

Paramètres de connexion



Configuration active :

Active ou désactive la configuration en cours. Par défaut, elle est désactivée. Quand ownCloud réussit le test de connexion, elle est automatiquement activée.

Serveur de backup (réplique) :

Si vous avez un serveur LDAP de secours, saisissez les paramètres de connexion ici. ownCloud se connectera alors automatiquement sur ce serveur si le serveur principal ne peut être contacté. Le serveur de secours doit être une réplique du serveur principal afin que les objets UUID correspondent.

Exemple :

- *annuaire2.ma-societe.com*

Port du serveur de backup (réplique) :

Le port de connexion du serveur LDAP de secours. Si aucun port n'est donnée, mais seulement l'hôte, alors le numéro de port du serveur principal sera utilisé.

Exemple :

- *389*

Désactiver le serveur principal :

Vous pouvez forcer la désactivation du serveur principal et forcer ownCloud à se connecter sur le serveur de secours. Ceci est utile pour effectuer une maintenance sur le serveur principal.

Désactiver la validation des certificats SSL :

Désactive la vérification des certificats SSL. À n'utiliser que pour faire des tests !

Durée de vie du cache (TTL) :

Un cache est introduit pour éviter du trafic LDAP inutile, par exemple les noms d'utilisateur pour éviter d'interroger le serveur LDAP pour chaque page et accélérer le chargement de la page Utilisateurs. L'enregistrement de la configuration vide le cache. La durée est exprimée en secondes.

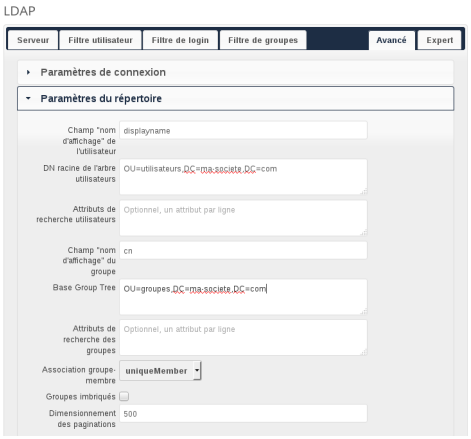
Veuillez noter que la plupart des requêtes PHP nécessite une nouvelle connexion au serveur LDAP. Si vous avez besoin de requêtes PHP à jour, nous recommandons de définir une durée de vie de 15 secondes par exemple plutôt que de désactiver totalement le cache.

Exemples :

- dix minutes : *600*
- une heure : *3600*

Consulter la section « Cache » ci-dessous pour des informations détaillées sur la gestion du cache.

Paramètres du répertoire



Champ "nom d'affichage" de l'utilisateur :

L'attribut qui doit être utilisé pour le nom d'affichage dans ownCloud.

- Exemple : *displayName*

DN racine de l'arbre utilisateurs :

Le DN de la racine LDAP, à partie de laquelle les utilisateurs seront recherchés. Ce doit être un DN complet, quel que soit ce que vous avez saisi dans le DN racine dans les paramètres de base. Vous pouvez définir plusieurs racines, une par ligne.

- Exemple :

  cn=programmeurs,dc=ma-societe,dc=com
  cn=designers,dc=ma-societe,dc=com

Attributs de recherche utilisateurs :

Ces attributs sont utilisés quand des recherches d'utilisateurs sont effectuées, par exemple dans le dialogue de partage. Le nom d'affichage de l'utilisateur est l'attribut par défaut. Vous pouvez indiquer plusieurs attributs, un par ligne.

Si un attribut n'est pas disponible pour un objet utilisateur, l'utilisateur ne sera pas listé et ne pourra pas se connecter. Cela affecte également l'attribut « displayName ». Si vos changez la valeur par défaut, vous devez indiquer ici l'attribut pour le nom d'affichage.

- Exemple :

  displayName
  mail

Champ "nom d'affichage" du groupe :

L'attribut qui doit être utilisé pour le nom de groupe ownCloud. ownCloud autorise un jeu limité de caractères (a-zA-Z0-9.-_@). Une fois que le nom du groupe a été assigné, il ne peut être modifié.

- Example : *cn*

Base Group Tree :

Le DN de la racine LDAP, à partie de laquelles les groupes seront recherchés. Ce doit être un DN complet, quel que soit ce que vous avez saisi dans le DN racine dans les paramètres de base. Vous pouvez définir plusieurs racines, une par ligne.

- Exemple :

  cn=barcelone,dc=ma-societe,dc=com
  cn=madrid,dc=ma-societe,dc=com

Attributs de recherche des groupes :

Ces attributs sont utilisés quand des recherches de groupes sont effectuées, par exemple dans le dialogue de partage. Le nom d'affichage du groupe est l'attribut par défaut. Vous pouvez indiquer plusieurs attributs, un par ligne.

Si vous avez changé la valeur par défaut, l'attribut de nom d'affichage du groupe ne sera pas pris en compte, à moins de le spécifier.

- Exemple :

  cn
  description

Association groupe-membre :

L'attribut utilisé pour indiquer l'appartenance aux groupes, c'est-à-dire l'attribut utilisé par les groupes LDAP pour se référer à leurs utilisateurs.

ownCloud détecte la valeur automatiquement. Vous ne devriez le changer que si vous avez une très bonne raison et que vous savez ce que vous faites.

- Exemple : *uniquemember*

Attributs spéciaux

Champ du quota :

ownCloud peut lire un attribut LDAP et définir le quota de l'utilisateur en fonction de sa valeur. Indiquer l'attribut ici et il renverra des valeurs compréhensibles, comme « 2 GB ». Les quotas définis dans LDAP ont la priorité sur ceux définis dans la page de gestion des utilisateurs d'ownCloud.

- Exemple : *ownCloudQuota*

Quota par défaut :

Remplace le quota par défaut d'ownCloud pour les utilisateurs n'ayant pas de quota défini dans le champ « Champ du quota ».

- Exemple : *15 GB*

Champ Email :

Définit l'adresse électronique de l'utilisateur à partir de son attribut LDAP. Laisser vide pour avoir le comportement par défaut.

- Exemple : *mail*

Convention de nommage du répertoire utilisateur :

Par défaut, le serveur ownCloud crée le répertoire utilisateur dans le répertoire data d'ownCloud et lui donne le nom de l'utilisateur, par exemple `/var/www/owncloud/data/alice`. Vous pourriez vouloir changer ce paramètre et le nommer en fonction d'une valeur d'attribut LDAP. L'attribut peut également renvoyer un chemin absolu, par exemple `/mnt/storage43/alice`. Laisser vide pour avoir le comportement par défaut.

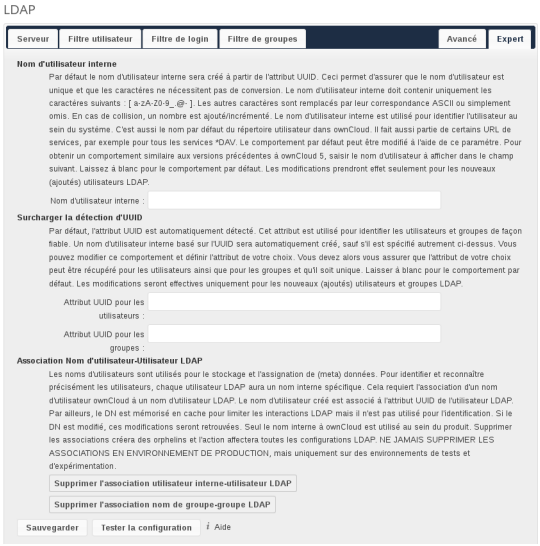
- Exemple : *cn*

Dans les nouvelles installations d'ownCloud (8.0.10, 8.1.5, 8.2.0 et suivantes), la convention de nommage du répertoire utilisateur est mise en œuvre. Cela signifie qu'une fois définie la convention de nommage du répertoire utilisateur, (obtenir le répertoire utilisateur à partir d'un attribut LDAP), le répertoire doit être disponible pour tous les utilisateurs. Si ce n'est pas le cas pour un utilisateur, alors celui-ci ne pourra pas se connecter. De plus, le système de fichiers ne sera pas défini pour cet utilisateur, et donc ses partages de fichiers ne seront pas disponibles pour les autres utilisateurs.

Dans les installations ownCloud existantes, l'ancien comportement s'applique toujours, c'est-à-dire que le nom d'utilisateur est utilisé pour le répertoire utilisateur quand l'attribut LDAP n'est pas défini. Vous pouvez changer ceci en mettant en œuvre la convention de nommage du répertoire utilisateur avec la commande `occ` dans ownCloud 8.2, comme dans cet exemple pour Ubuntu:

```
sudo -u www-data php occ config:app:set user_ldap enforce_home_folder_naming_rule --value=1
```

Expert



Warning

Dans les paramètres Expert, le comportement de base peut être adapté à vos besoins. La configuration doit être bien testée avant d'être mise en production.

Nom d'utilisateur interne

Le nom d'utilisateur interne est l'identifiant ownCloud pour les utilisateurs LDAP. Par défaut, il est créé à partir de l'attribut UUID. L'attribut UUID permet d'assurer que le nom d'utilisateur est unique et que les caractères n'ont pas besoin d'être convertis. Seuls les caractères suivants sont acceptés : [a-zA-Z0-9_.@-]. Les autres caractères sont remplacés par leurs équivalents ASCII ou tout simplement omis.

Le service LDAP s'assure qu'il n'y a pas de doublons de noms d'utilisateurs internes dans ownCloud, c'est-à-dire qu'il vérifie tous les autres services de gestion d'utilisateurs (y compris les utilisateurs locaux d'ownCloud). En cas de collisions, un nombre aléatoire(entre 1000 et 9999) est ajouté à la valeur récupérée. Par exemple, si « alice » existe, le nom d'utilisateur suivant pourra être « alice_1337 ».

Le nom d'utilisateur interne est le nom par défaut du répertoire utilisateur dans ownCloud. C'est aussi une partie des URL distantes, comme par exemple tous les services *DAV.

Vous pouvez changer tout cela en utilisant le paramètre « Nom d'utilisateur interne ». Laisser vide pour conserver le comportement par défaut. Les modifications n'affecterons que les nouveaux utilisateurs LDAP connectés à owncloud.

- Example : *uid*

Surcharger la détection d'UUID

Par défaut, ownCloud détecte automatiquemnt l'attribut UUID. L'attribut UUID est utilisé pour identifier de façon unique les utilisateurs et les groupes LDAP. Le nom d'utilisateur interne est créé sur la base de l'UUID, sauf indications contraires.

Vous pouvez passer outre ce paramètre et indiquer l'attribut de votre choix. Vous devez vous assurer que l'attribut choisi peut être récupéré pour les utilisateurs et les groupes et qu'il est unique. Laisser vide pour conserver le comportement par défaut. Les modifications ne prendront effet que sur les nouveaux groupes et utilisateurs LDAP connectés à ownCloud. Cela prendra effet également quand le DN d'un utilisateur ou d'un groupe change et qu'un ancien UUID est en cache, ce qui aura pour résultat un nouvel utilisateur ou groupe. Pour cette raison, le paramètre doit être mis en œuvre avant la mise en production d'ownCloud et le nettoyage des liaisons (voir la section `Correspondances utilisateur et groupe` ci-dessous).

- Example : *cn*

Association Nom d'utilisateur-Utilisateur LDAP

ownCloud utilise les noms d'utilisateurs comme clés pour stocker et assigner les données. Afin d'identifier précisément et de reconnaître les utilisateurs, chaque utilisateur LDAP a un nom d'utilisateur interne à ownCloud. Ceci nécessite une correspondance entre le nom d'utilisateur ownCloud et le nom d'utilisateur LDAP. Le nom d'utilisateur créé est associé à l'UUID de l'utilisateur LDAP. De plus, le DN est mis en cache pour réduire l'interaction avec LDAP, mais n'est pas utilisé pour l'identification. Si le DN change, le changement sera détecté par ownCloud en vérifiant la valeur de l'UUID.

Ceci s'applique aussi aux groupes.

Le nom interne ownCloud est utilisé partout dans ownCloud. La suppression des associations laissera des traces partout. Ne supprimer jamais les associations dans un environnement de production, mais seulement en test ou sur un serveur expérimental.

La suppression des associations n'est pas propre à la configuration en cours, elle affecte toutes les configurations !

Test de la configuration

Le bouton **Tester la configuration** vérifie les valeurs données dans les champs de saisie. Vous n'avez pas besoin de sauvegarder avant de tester. En cliquant sur le bouton, ownCloud essaiera d'établir la liaison en utilisant les paramètres actuellement déifinis dans les champs de saisie. Si la liaison échoue, vous verrez un bandeau jaune avec le message suivant : « La configuration est invalide. Veuillez consulter le fichier journal d'ownCloud pour plus de détails. ».

Quand le test de connexion est réussi, sauvegarder la configuration et vérifier que les utilisateurs et les groupes sont correctement récupérés sur la page Utilisateurs.

Intégration des avatars dans ownCloud

ownCloud gère les photos de profil utilisateur, aussi appelées avatars. Si un utilisateur a une photo stockée dans l'attribut *jpegPhoto* ou *thumbnailPhoto* de votre annuaire LDAP, elle sera utilisée comme avatar. Dans ce cas, l'utilisateur ne pourra pas modifier son avatar sur sa page personnelle car elle doit être chagée dans l'annuaire LDAP. L'attribut *jpegPhoto* est préféré à l'attribut *thumbnailPhoto*.

Si l'attribut *jpegPhoto* ou *thumbnailPhoto* n'est pas défini ou vide, les utilisateurs peuvent alors téléverser leur avatar sur leur page personnelle. Les avatars gérés dans ownCloud ne sont pas stockés dans le serveur LDAP.

L'attribut *jpegPhoto* ou *thumbnailPhoto* est vérifié une fois par jour pour s'assurer que la photo de l'annuaire LDAP est utilisée dans ownCloud. Les avatars LDAP ont la priorité sur les avatars d'ownCloud, et quand la photo de l'annuaire LDAP est supprimée, l'avatar le plus récent dans ownCloud la remplace.

Les photos récupérées dans l'annuaire LDAP sont automatiquement redimensionnées dans ownCloud. Ceci n'affecte que la présentation et la photo originale n'est pas modifiée.

Dépannage et astuce

Vérification de certificat SSL (LDAPS, TLS)

Une erreur courante avec les cetificats SSL est qu'ils peuvent ne pas être connus dans PHP. Si vous avez des problèmes avec la validation de certificats, assurez-vous que :

- le certificat du serveur concerné est bien installé sur le serveur ownCloud ;
- le certificat est indiqué dans le fichier de configuration LDAP (généralement */etc/ldap/ldap.conf*) ;
- d'utiliser LDAPS ; assurez-vous également que le port est correctement configuré (par défaut *636*).

Microsoft Active Directory

Par rapport aux précédentes versions d'ownCloud, aucun autre paramétrage n'a besoin d'être effectué pour qu'ownCloud fonctionne avec Active Directory. ownCloud trouve automatiquement la configuration correcte pendant le processus de paramétrage initial.

Permissions de lecture de memberOf

Si vous voulez autoriser `memberOf` comme filtre, vous pourriez avoir besoin de donner à l'utilisateur LDAP faisant les requêtes de l'utiliser. Pour Microsoft Active Directory, ceci est décrit ici.

Duplication de configurations de serveur

Dans le cas où vous avez une configuration opérationnelle et que voulez en créer une similaire ou prendre une « photo » des configurations avant de les modifier, vous pouvez faire ce qui suit :

1. rendez-vous dans l'onglet **Serveur** ;
2. dans la liste déroulante **Serveur** choisissez *Ajouter une configuration du serveur* ;
3. à la question *Récupérer les paramètres depuis une configuration récente du serveur ?*, répondez *Oui* ;
4. (facultatif) rendez-vous dans l'onglet **Avancé** et déselectionner la case **Configuration active** dans les *Paramètres de connexion* de sorte que la nouvelle configuration ne soit pas utilisée lors de l'enregistrement ;
5. cliquez sur le bouton **Sauvegarder**.

Vous pouvez maintenant modifier et activer la configuration.

Cache

Utilisation du cache pour accélérer les recherches. Consulter *Configuration de la mémoire cache*). Le cache d'ownCloud est rempli à la demande et demeure et demeure jusqu'à ce que la **Durée de vie du cache (TTL)** pour chaque requête unique expire. Les identifiants des utilisateurs ne sont pas mis en cache, donc, si vous voulez améliorer les temps de connexion, il faudra installer un serveur LDAP esclave pour partager la charge.

Vous pouvez ajuster la valeur de la **Durée de vie du cache (TTL)** pour trouver le bon équilibre entre performance et fraîcheur des données LDAP. Par défaut, toutes les requêtes LDAP sont mise en cache pendant 10 minutes, et vous pouvez modifier cette valeur avec le paramètre **Durée de vie du cache (TTL)**. Le cache répond à chaque requête identique à une requête précédente pendant la durée de vie de la requête originale plutôt que d'interroger le serveur LDAP.

La **Durée de vie du cache (TTL)** est liée à chaque requête. Après l'expiration d'une entrée de cache, il n'y a pas de déclencheur automatique pour récupérer à jour, car le cache est alimenté seulement par les nouvelles requêtes, par exemple en ouvrant la page d'administration des utilisateurs ou en effectuant une recherche dan un dialogue de partage.

Il n'existe qu'un déclencheur qui est automatiquement déclenché par une tâche de fond qui conserve les données de `user-group-mappings` à jour et en cache.

Dans des circonstances normales, tous les utilisateurs ne sont pass chargés en même temps. Typiquement, le chargement des utilisateurs intervient quand les pages de résultats sont générées, par lot de 30 jusqu'à ce que la limite soit atteinte ou qu'il n'y ait plus de résultats. Pour que ceci fonctionne pour un serveur ownCloud et LDAP, **Paged Results** doit être géré, ce qui suppose d'utiliser PHP au moins dans sa version 5.4.

ownCloud se souvient de la configuration LDAP à laquelle appartient un utilisateur. Cela signifie que chaque requête sera toujours dirigée vers le bon serveur, sauf si l'utilisateur n'existe plus, par exemple à cause de la migration d'un serveur ou parce qu'il est injoignable. Dans ce cas, les autres serveurs recevront aussi la requête.

Indexation LDAP

Activer l'indexation. Le choix des attributs à indexer dépend de votre configuration et du serveur LDAP utilisé. Consulter index openLDAP pour openLDAP, et Comment indexer un attribut dans Active Directory pour Active Directory. Le guide openLDAP est particulièrement utile pour savoir quels attributs indexer.

Utilisation d'une base DN précise

Plus la base DN sera précise, plus les requêtes LDAP seront rapides car le serveur devra parcourir moins de branches pour les recherches.

Utilisation de filtre précis

Utiliser de bons filtres pour limiter l'étendue des recherches LDAP, et pour diriger intelligemment le serveur dans ses recherches plutôt que de faire des recherches sur tous les objets.

Fonctionnement LDAP d'ownCloud

Certains aspects du fonctionnement du service LDAP d'ownCoud sont décrits ici.

Correspondances utilisateur et groupe

Dans ownCloud, le nom de l'utilisateur ou du groupe est utilisé pour avoir toutes les informations pertinentes assignées dans la base de données. Pour fonctionner de manière fiable, un nom d'utilisateur ou de groupe interne est créé et associé au DN et à l'UUID LDAP. Si le DN change dans LDAP, owncloud le détecte et il n'y aura alors pas de conflit.

Ces correspondances sont faites dans les tables `ldap_user_mapping` et `ldap_group_mapping` de la base de données. Le nom d'utilisateur est aussi utilisé pour le nom du répertoire utilisateur (sauf indication contraire dans *Convention de nommage du répertoire utilisateur*), qui contient les fichiers et les méta-données.

À compter de la version 5 d'ownCloud, le nom d'utilisateur et le nom d'affichage sont distincts. Ce n'est pas le cas pour les noms de groupe, c'est-à-dire qu'un nom de groupe ne peut être altéré.

Ceci signifie que votre configuration LDAP doit être finalisée et correcte avant de passer en production. Les tables de correspondances sont remplies rapidement, mais tant que vous n'êtes pas en production, vous pouvez les vider à tout moment. Ne faites pas cela en production.

Serveur de secours

Quand ownCloud n'arrive pas à contacter le serveur principal LDAP, il suppose que celui-ci est éteint et n'essaiera pas de s'y reconnecter pendant la durée spécifiée de la **Durée de vie du cache (TTL)**. Si vos avez un serveur de secours configuré, ownCloud se connectera alors sur celui-ci. Quand vous avez planifié une maintenance, cochez la case **Désactiver le serveur principal** pour éviter des tentatives de connexion inutiles.

Purge des utilisateurs LDAP

« LDAP User Cleanup » est une nouvelle fonctionnalité de l'application `LDAP user and group backend`. « LDAP User Cleanup » est une tâche de fond qui recherche automatiquement dans les tables de correspondances LDAP d'ownCloud, et véirifie si les utilisateurs LDAP sont encore disponibles. Tous les utilisateurs non disponibles sont marqués comme `deleted` dans la table `oc_preferences` de la base de données. Vous pouvez alors lancer une commande pour afficher cette table, afficher seulement les utilisateurs marqués `deleted`, et vous avez l'option de supprimer leurs données du répertoire data d'ownCloud.

Les éléments suivants sont supprimés :

- l'appartenance aux groupes locaux d'ownCloud ;
- les préférences utilisateur (table `oc_preferences` de la base de données) ;
- le répertoire utilisateur ownCloud ;
- les entrées correspondantes dans la table `oc_storages`.

Il y a deux prérequis pour que « LDAP User Cleanup » fonctionne :

1. Définir le paramètre `ldapUserCleanupInterval` dans le fichier `config.php` à la valeur désirée en minutes. Par défaut, 51 minutes.
2. Toutes les conexions LDAP doivent être activées et fonctionner correctement. Comme les utilisateurs peuvent exister sur plusieurs serveurs LDAP, vous voulez être sûr que tous vos serveurs LDAP sont joignables pour qu'un utilisateur présent sur un serveur temporairement indisponible ne soit pas marqué `deleted`.

Le processus en tâche de fond examine 50 utilisateurs à la fois, et s'exécute à l'intervalle défini dans `ldapUserCleanupInterval`. Par exemple, si vous avez 200 utilisateurs LDAP et que l'intervalle `ldapUserCleanupInterval` est de 20 minutes, le processus examinera les 50 premiers utilisateurs, puis 20 minutes plus tard, les 50 suivants et aisni de suite.

Il existe deux commandes `occ` à utiliser pour examiner la table des utilisateurs marqués `deleted` et les supprimer manuellement. La commande `occ` se trouve dans votre répertoire ownCloud, par exemple `/var/www/owncloud/occ` et doit être exécutée en tant qu'utilisateur HTTP. Pour en apprendre plus sur `occ`, consulter *Utilisation de la commande occ*.

Ces exemples sont applicables pour Ubuntu Linux :

1. `sudo -u www-data php occ ldap:show-remnants` affiche une table avec tous les utilisateurs qui ont été marqués comme supprimés et leurs données LDAP.
2. `sudo -u www-data php occ user:delete [user]` supprime les données utilisateur du répertoire data d'ownCloud.

Cet exemple montre à quoi ressemble la table des utilisateurs marqués `deleted`:

```
$ sudo -u www-data php occ ldap:show-remnants
+-----------------+-----------------+------------------+-----------------------------------
| ownCloud name   | Display Name    | LDAP UID         | LDAP DN
+-----------------+-----------------+------------------+-----------------------------------
| aaliyah_brown   | aaliyah brown   | aaliyah_brown    | uid=aaliyah_brown,ou=people,dc=com
| aaliyah_hammes  | aaliyah hammes  | aaliyah_hammes   | uid=aaliyah_hammes,ou=people,dc=com
| aaliyah_johnston| aaliyah johnston| aaliyah_johnston | uid=aaliyah_johnston,ou=people,dc=c
| aaliyah_kunze   | aaliyah kunze   | aaliyah_kunze    | uid=aaliyah_kunze,ou=people,dc=com
+-----------------+-----------------+------------------+-----------------------------------
```

Vous pouvez alors lancer la commande `sudo -u www-data php occ user:delete aaliyah_brown` pour supprimer l'utilisateur aaliyah_brown. Vous devez utiliser le nom d'utilisateur ownCloud.

Suppression des utilisateurs locaux d'ownCloud

Vous pouvez aussi utiliser la commande `occ user:delete [utilisateur]` pour supprimer un utilisateur ownCloud local. Ceci supprime son compte et ses données.

API de provisioning utilisateurs

L'API de provisioning acive un ensemble d'API qui peuvent être utilisés par des systèmes externes pour créer, modifier, supprimer des utilisateurs ou des groupes, gérer les quotas, etc. Les administrateurs de groupes peuvent aussi réaliser les mêmes actions pour les groupes qu'ils administrent. L'API permt aussi à un administrateur d'interroger ownCloud sur les applications actives, sur les informations des applications, et d'activer ou de désactiver une application à distance. Des requêtes HTTP peuvent être utilisées à l'aide d'un en-tête Basic Auth pour réaliser toutes les actions décrites précédemment. L'API de provisioning ext activée par défaut.

L'URL de base pour les appels de l'API est **owncloud_base_url/ocs/v1.php/cloud**.

Jeu d'instructions pour les utilisateurs

users / adduser

Crée un nouvel utilisateur sur le serveur ownCloud. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users

- Méthode HTTP : POST
- Argument POST : userid - chaîne, le nom d'utilisateur requis pour le nouvel utilisateur
- Argument POST : password - chaîne, le mot de passe requis pour le nouvel utilisateur

Codes d'état :

- 100 - succès

- 101 - données invalides
- 102 - le nom d'utilisateur existe déjà
- 103 - une erreur inconnue est survenue lors de l'ajout de l'utilisateur

Exemple

- POST `http://admin:secret@example.com/ocs/v1.php/cloud/users -d userid="Frank" -d password="frankspassword"`
- Crée l'utilisateur `Frank` avec le mot de passe `frankspassword`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
 <meta>
  <status>ok</status>
  <statuscode>100</statuscode>
  <message/>
 </meta>
 <data/>
</ocs>
```

users / getusers

Récupère une liste d'utilisateurs du serveur ownCloud. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users

- Méthode HTTP : GET
- Arguments url : search - chaîne, chaîne de recherche facultative
- Arguments url : limit - entier, valeur limite facultative
- Arguments url : offset - entier, valeur de déplacement facultative

Codes d'état :

- 100 - succès

Exemple

- GET `http://admin:secret@example.com/ocs/v1.php/cloud/users?search=Frank`
- Renvoie une liste d'utilisateurs correspondant à la chaîne de recherche.

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data>
    <users>
      <element>Frank</element>
     </users>
  </data>
</ocs>
```

users / getuser

Récupère des informations sur un utilisateur. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}

- Méthode HTTP : GET

Codes d'état :

- 100 - succès

Exemple

- GET `http://admin:secret@example.com/ocs/v1.php/cloud/users/Frank`
- Renvoie des informations sur l'utilisateur `Frank`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data>
    <email>frank@example.org</email>
    <quota>0</quota>
    <enabled>true</enabled>
  </data>
</ocs>
```

users / edituser

Modifie les attributs relatifs à un utilisateur. Les utilisateurs peuvent modifier leur adresse électronique, leur nom complet et leur mot de passe. Les administrateurs peuvent en plus modifier la valeur du quota. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}

- Méthode HTTP : PUT
- Argument PUT : clé, le champ à modifier (email, quota, display, password)
- Argument PUT : valeur, la nouvelle valeur pour le champ

Codes d'état :

- 100 - succès
- 101 - utilisateur non trouvé
- 102 - données invalides

Exemples

- `PUT PUT http://admin:secret@example.com/ocs/v1.php/cloud/users/Frank -d key="email" -d value="franksnewemail@example.org"`
- Met à jour l'adresse électronique pour l'utilisateur `Frank`
- `PUT PUT http://admin:secret@example.com/ocs/v1.php/cloud/users/Frank -d key="quota" -d value="100MB"`
- Met à jour le quota pour l'utilisateur `Frank`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

users / deleteuser

Supprime un utilisateur du serveur ownCloud. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}

- Méthode HTTP : DELETE

Codes d'état :

- 100 - succès
- 101 - échec

Exemple

- DELETE `http://admin:secret@example.com/ocs/v1.php/cloud/users/Frank`
- Supprime l'utilisateur `Frank`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

users / getgroups

Récupère une liste de groupes dont l'utilisateur spécifié est membre. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}/groups

- Méthode HTTP : GET

Codes d'état :

- 100 - succès

Exemple

- GET `http://admin:secret@example.com/ocs/v1.php/cloud/users/Frank/groups`
- Récupère la liste de groupes dont `Frank` est membre

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data>
    <groups>
      <element>admin</element>
      <element>group1</element>
    </groups>
  </data>
</ocs>
```

users / addtogroup

Ajoute l'utilisateur spécifié au groupe spécifié. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}/groups

- Méthode HTTP : POST
- Argument POST : groupid, chaîne - le groupe auquel ajouter l'utilisateur

Codes d'état :

- 100 - succès
- 101 - aucun groupe spécifié
- 102 - le groupe n'existe pas
- 103 - l'utilisateur n'existe pas
- 104 - privilèges insuffisants
- 105 - échec de l'ajout de l'utilisateur au groupe

Exemple

- POST `http://admin:secret@example.com/ocs/v1.php/cloud/users/Frank/groups -d groupid="newgroup"`
- Ajoute l'utilisateur `Frank` au groupe `newgroup`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

users / removefromgroup

Supprime l'utilisateur spécifié du groupe spécifié. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}/groups

- Méthode HTTP : DELETE

- ArgumentPOST : groupid, chaîne - le groupe duquel supprimer l'utilisateur

Codes d'état :

- 100 - succès
- 101 - aucun groupe spécifié
- 102 - le groupe n'existe pas
- 103 - l'utilisateur n'existe pas
- 104 - privilèges insuffisants
- 105 - échec de la suppression de l'utilisateur du groupe

Exemple

- DELETE `http://admin:secret@example.com/ocs/v1.php/cloud/users/Frank/groups -d groupid="newgroup"`
- Supprime l'utilisateur `Frank` du groupe `newgroup`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

users / createsubadmin

Rend un utilisateur administrateur d'un groupe. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}/subadmins

- Méthode HTTP : POST
- Argument POST : groupid, chaîne - le groupe dont l'utilisateur doit être administrateur

Codes d'état :

- 100 - succès
- 101 - l'utilisateur n'existe pas
- 102 - le groupe n'existe pas
- 103 - erreur inconnue

Exemple

- POST `https://admin:secret@example.com/ocs/v1.php/cloud/users/Frank/subadmins -d groupid="group"`
- Rend l'utilisateur `Frank` administrateur du groupe `group`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
```

```
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

users / removesubadmin

Retire les droits d'administration d'un utilisateur spécifié sur un groupe spécifié. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}/subadmins

- Méthode HTTP : DELETE
- Argument DELETE : groupid, chaîne - le groupe duquel retirer les droits d'administration

Codes d'état :

- 100 - succès
- 101 - l'utilisateur n'existe pas
- 102 - l'utilisateur n'est pas administrateur du groupe / le groupe n'existe pas
- 103 - erreur inconnue

Exemple

- DELETE `https://admin:secret@example.com/ocs/v1.php/cloud/users/Frank/subadmins -d groupid="oldgroup"`
- Retire les droits d'administration de l'utilisateur `Frank's` pourle groupe `oldgroup`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

users / getsubadmingroups

Renvoie les groupes dont l'utilisateur est administrateur. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/users/{userid}/subadmins

- Méthode HTTP : GET

Codes d'état :

- 100 - succès
- 101 - l'utilisateur n'existe pas
- 102 - erreur inconnue

Exemple

- GET `https://admin:secret@example.com/ocs/v1.php/cloud/users/Frank/subadmins`
- Renvoie les groupes dont l'utilisateur `Frank` est administrateur

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
      <status>ok</status>
      <statuscode>100</statuscode>
    <message/>
  </meta>
  <data>
    <element>testgroup</element>
  </data>
</ocs>
```

Jeu d'instructions pour les groupes

groups / getgroups

Récupère une liste de groupes du serveur ownCloud. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/groups

- Méthode HTTP : GET
- Arguments url : search - chaîne, chaîne de recherche facultative
- Arguments url : limit - entier, valeur de limite facultative
- Arguments url : offset - entier, valeur de déplacement facultative

Codes d'état :

- 100 - succès

Exemple

- GET `http://admin:secret@example.com/ocs/v1.php/cloud/groups?search=adm`
- Renvoie une liste de groupes correspondant à la chaîne de recherche

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data>
    <groups>
      <element>admin</element>
    </groups>
  </data>
</ocs>
```

groups / addgroup

Ajoute un nouveau groupe.L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/groups

- Méthode HTTP : POST
- Argument POST : groupid, chaîne - le nom du nouveau groupe

Codes d'état :

- 100 - succès
- 101 - données invalides
- 102 - le groupe existe déjà
- 103 - échec de l'ajout du groupe

Exemple

- POST `http://admin:secret@example.com/ocs/v1.php/cloud/groups -d groupid="newgroup"`
- Ajoute un nouveau groupe nommé `newgroup`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

groups / getgroup

Récupère une liste des membres du groupe. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/groups/{groupid}

- Méthode HTTP : GET

Codes d'état :

- 100 - succès

Exemple

- POST `http://admin:secret@example.com/ocs/v1.php/cloud/groups/admin`
- Renvoie une liste des utilisateurs membres du groupe `admin`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data>
    <users>
```

```
      <element>Frank</element>
    </users>
  </data>
</ocs>
```

groups / getsubadmins

Renvoie la liste des administrateurs du groupe. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/groups/{groupid}/subadmins

- Méthode HTTP : GET

Codes d'état :

- 100 - succès
- 101 - le groupe n'existe pas
- 102 - erreur inconnue

Exemple

- GET `https://admin:secret@example.com/ocs/v1.php/cloud/groups/mygroup/subadmins`
- Renvoie la liste des administrateurs du groupe `mygroup`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <status>ok</status>
    <statuscode>100</statuscode>
    <message/>
  </meta>
  <data>
    <element>Tom</element>
  </data>
</ocs>
```

groups / deletegroup

Supprime un groupe. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/groups/{groupid}

- Méthode HTTP : DELETE

Codes d'état :

- 100 - succès
- 101 - le groupe n'existe pas
- 102 - échec de la suppression du groupe

Exemple

- DELETE `http://admin:secret@example.com/ocs/v1.php/cloud/groups/mygroup`
- Supprime le groupe `mygroup`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data/>
</ocs>
```

Jeu d'instructions pour les applications

apps / getapps

Renvoie une liste des applications installées sur le serveur ownCloud. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/apps/

- Méthode HTTP : GET
- Argument url : filter, chaîne - facultatif (`enabled` ou `disabled`)

Codes d'état :

- 100 - succès
- 101 - données invalides

Exemple

- GET `http://admin:secret@example.com/ocs/v1.php/cloud/apps?filter=enabled`
- Obtient la liste des applications activées

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data>
    <apps>
      <element>files</element>
      <element>provisioning_api</element>
    </apps>
  </data>
</ocs>
```

apps / getappinfo

Fournit des informations sur une application spécifique. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/apps/{appid}

- Méthode HTTP : GET

Codes d'état :

- 100 - succès

Exemple

- GET `http://admin:secret@example.com/ocs/v1.php/cloud/apps/files`
- Obtient les informations sur l'application `files`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
  <data>
    <info/>
    <remote>
      <files>appinfo/remote.php</files>
      <webdav>appinfo/remote.php</webdav>
      <filesync>appinfo/filesync.php</filesync>
    </remote>
    <public/>
    <id>files</id>
    <name>Files</name>
    <description>File Management</description>
    <licence>AGPL</licence>
    <author>Robin Appelman</author>
    <require>4.9</require>
    <shipped>true</shipped>
    <standalone></standalone>
    <default_enable></default_enable>
    <types>
      <element>filesystem</element>
    </types>
  </data>
</ocs>
```

apps / enable

Active une application. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/apps/{appid}

- Méthode HTTP : POST

Codes d'état :

- 100 - succès

Exemple

- POST `http://admin:secret@example.com/ocs/v1.php/cloud/apps/files_texteditor`
- Active l'application `files_texteditor`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
```

```
  </meta>
</ocs>
```

apps / disable

Désactive une application. L'authentification est faite en envoyant un en-tête HTTP d'authentification basique.

Syntaxe : ocs/v1.php/cloud/apps/{appid}

- Méthode HTTP : DELETE

Codes d'état :

- 100 - succès

Exemple

- DELETE `http://admin:secret@example.com/ocs/v1.php/cloud/apps/files_texteditor`
- Désactive l'application `files_texteditor`

Sortie XML

```
<?xml version="1.0"?>
<ocs>
  <meta>
    <statuscode>100</statuscode>
    <status>ok</status>
  </meta>
</ocs>
```

Gestion et partage de fichiers

Partage de fichiers

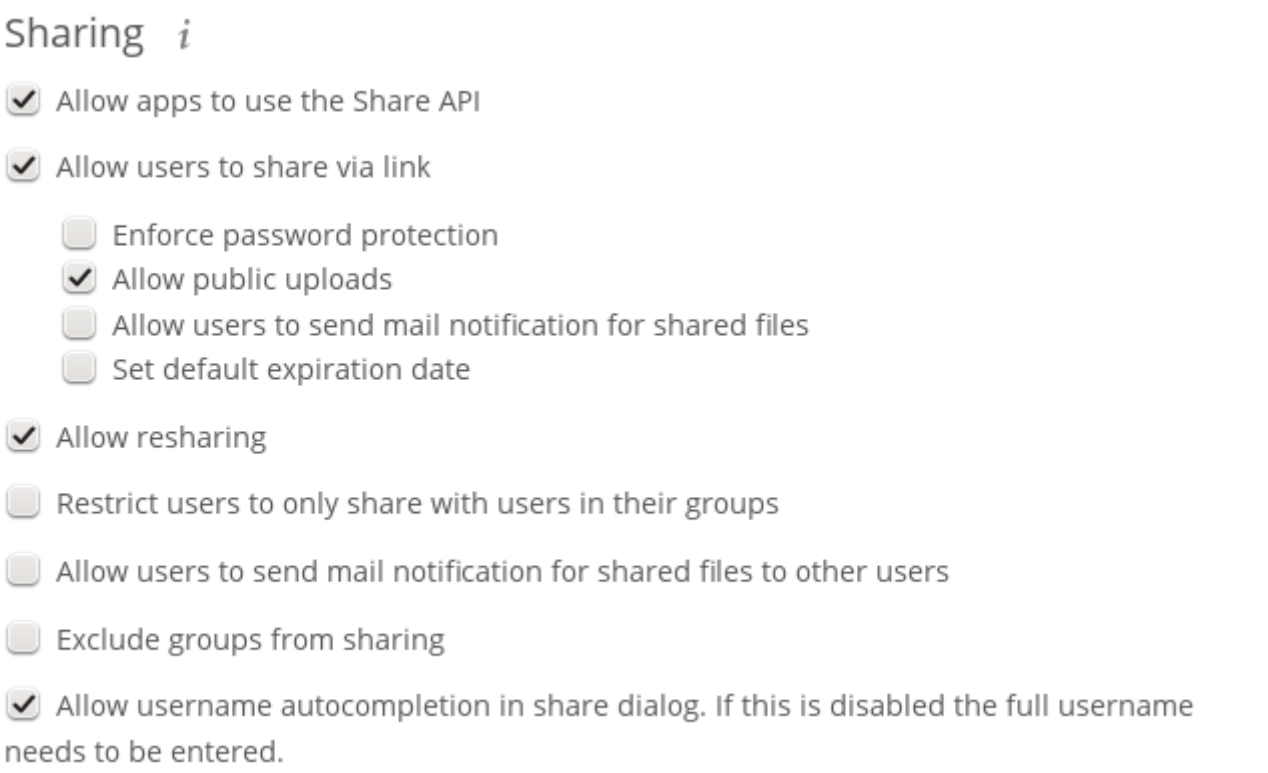
Les utilisateurs d'ownCloud peuvent partager les fichiers avec les utilisateurs de leurs groupes ownCloud ou d'autres utilisateurs sur le même serveur ownCloud, avec des utilisateus ownCloud sur *d'autres serveurs ownCloud* et créer des partages publics avec des personnes qui ne sont pas des utilisateurs d'ownCloud. Vous pouvez contrôler un certain nombre d'autorisations sur les partages de fichiers :

- permettre aux utilisateurs de partager des fichiers ;
- permettre aux utilisateurs de créer des partages publics ;
- demander un mot de passe pour les partages publics ;
- autoriser les téléversements sur les partages publics ;
- forcer une date d'expiration sur les partages publics ;
- autoriser le repartage ;
- restreindre le partage à certains groupes seulement ;
- autoriser les notifications par courriel pour les nouveaux partages publics ;
- exclure des groupes de la création de partages.

Note

ownCloud Édition Entreprise propose une application de gestion de la politique des mots de passe des liens de partage ; voir *Politique de mot de passe des liens de partage*.

La configuration de la politique de partage se trouve sur la page d'administration dans la section Partage.



- Cocher la case `Autoriser les applications à utiliser l'API de partage` pour permettre aux utilisateurs de partager des fichiers. Si cette case n'est pas aucun utilisateur ne pourra créer de partages de fichiers.
- Cocher la case `Autoriser les utilisateurs à partager par lien` pour permettre la création de partages publics pour les personnes qui ne sont pas utilisateurs d'ownCloud à l'aide d'hyperliens.
- Cocher la case `Imposer la protection par mot de passe` pour obliger les utilisateurs à définir un mot de passe pour tous les liens de partages publics. Ceci ne s'applique pas aux utilisateurs locaux et aux partages de groupes.
- Cocher la case `Autoriser les téléversements publics` pour autoriser tout le monde à effectuer des téléversements sur les partages publics.
- Cocher la case `Autoriser les utilisateurs à envoyer des notifications de partage par e-mail` pour autoriser l'envoi de notifications à partir d'ownCloud (votre serveur ownCloud doit être configuré pour envoyer des courriels).
- Cocher la case `Spécifier une date d'expiration par défaut` pour définir une date d'expiration par défaut pour les partages publics.
- Cocher la case `Autoriser le repartage` pour permettre aux utilisateurs de repartager les fichiers.
- Cocher la case `N'autoriser les partages qu'entre membres de mêmes groupes` pour restreindre le partage aux membres d'un même groupe.

  > ***Note***
  >
  > Ce paramètre ne s'applique pas à la fonctionnalité de partage fédéré. Si le *Partage fédéré* est activé, les utilisateurs peuvent encore partager des éléments avec tout utilisateur sur n'importe quelle instance (y compris celle sur laquelle ils sont) à l'aide de partage distant.

- Cocher la case `Autoriser les utilisateurs à envoyer des notifications de partage par e-mail` permet aux utilisateurs d'envoyer un courriel de notification à chaque utilisateur avec qui le fichier ou dossier est partagé.

- Cocher la case `Empêcher certains groupes de partager` pour empêcher les membres de groupes spécifiques de créer des partages de fichiers dans ces groupes. Quand cette case est cochée, une liste déroulante de tous les groupes apparaît pour choisir les groupes. Les membres des groupes exclus peuvent toujours recevoir des partages mais ne peuvent pas en créer.
- Cocher la case `Activer l'autocomplétion des noms d'utilisateurs dans la fenêtre de partage.` pour activer la complétion automatique des noms d'utilisateurs d'ownCloud.

Note

ownCloud ne préserve pas l'attribut « mtime » (heure de modification) des répertoires, bien qu'il mette à jour l'attribut « mtimes » des fichiers. Voir Mauvaise date de dossier lors de la synchronisation pour une discussion à ce sujet.

Transfert de fichiers à un autre utilisateur

Vous pouvez transférer des fichiers d'un utilisateur à un autre avec `occ`. Ceci est utilise quand vous devez supprimer un utilisateur. Assurez-vous de transférer les fichiers avant de supprimer l'utilisateur ! Ceci transfère tous les fichiers de l'utilisateur 1 vers l'utilisateur 2, ainsi que les partages et les informations de méta-données associées aux fichiers (partages, étiquettes, commentaires, etc.). Le contenu des corbeilles n'est pas transféré:

```
occ files:transfer-ownership utilisateur1 utilisateur2
```

Voir *Utilisation de la commande occ* pour une référence complète sur la commande `occ`.

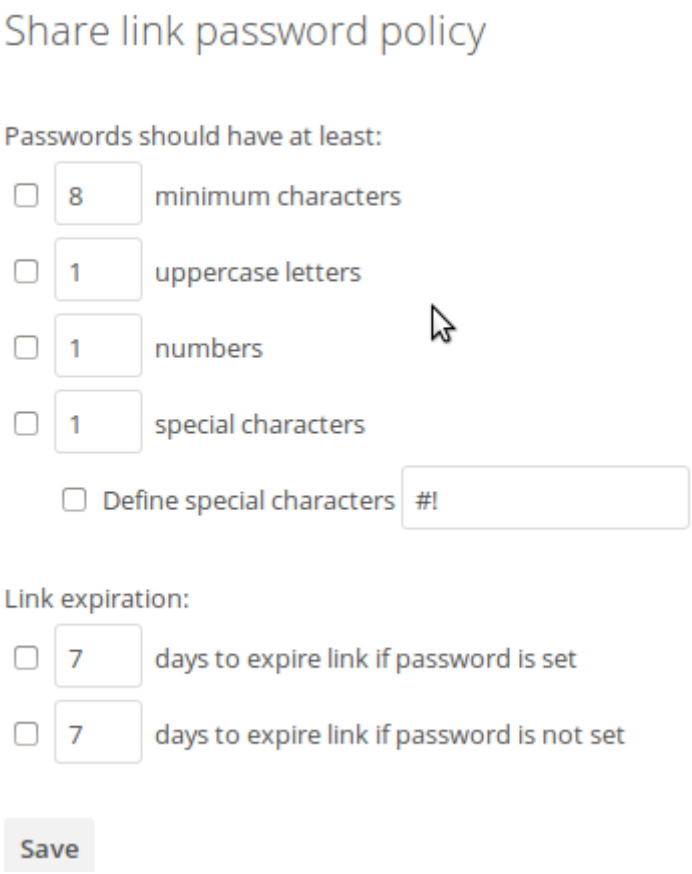
Création de partage de fichiers persistants

Quand un utilisateur est supprimé, ses fichiers sont aussi supprimés. Comme vous pouvez l'imaginez, cela représente un problème s'il a créé un partage de fichiers qui doit être conservé, car il sera aussi supprimé. Dans ownCloud, les fichiers sont liés à leurs utilisateurs. Ce qui survient au propriétaire des fichiers survient également aux fichiers.

Une solution est de créer un partage persistant pour vos utilisateurs. Vous pouvez conserver la propriété des fichiers ou créer un utilisateur spécial pour les partages de fichiers persistants. Créez un dossier partagé de la façon habituelle et partagez-le avec les utilisateurs et groupes qui doivent en bénéficier. Définissez les permissions appropriées sur celui-ci, et peu importe les utilisateurs ajoutés ou supprimés, le partage de fichiers demeurera. Ceci est possible car tous les fichiers ajoutés au partage ou modifiés dans celui-ci deviennent automatiquement la propriété du propriétaire du partage quel que soit l'utilisateur qui a ajouté ou modifié les fichiers.

Politique de mot de passe des liens de partage

Les utilisateurs d'ownCloud Édition Enterprise ont l'option d'activer l'application de politique de mots de passe des liens de partage. Ceci permet d'imposer la longueur d'un mot de passe, les types de caractères qu'il doit contenir et les dates d'expiration des liens de partage.



Veuillez noter que vous ne pouvez pas utiliser d'émojis comme caractère spécial avec MySQL car il ne sait gérer que les caractères UTF8 de un à trois octets et que les émojis nécessitent quatre octets.

Configuration du partage fédéré

Le partage fédéré est maintenant géré par l'application Fédération (9.0+). Après de l'activation de l'application Fédération, vous pouvez facilement et en toute sécurité créer des liens de partage de fichiers entre serveurs ownCloud, créant de fait un nuage de serveurs ownCloud.

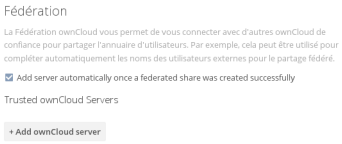
Partage avec ownCloud 8 et versions précédentes

Les partages fédérés directs (*Création d'un partage fédéré (9.0+ seulement)*) ne sont pas gérés dans ownCloud 8 et les versions précédentes. Vous devez donc créer des partages fédérés avec des liens publics (*Création de partages fédérés à l'aide du partage par lien public*).

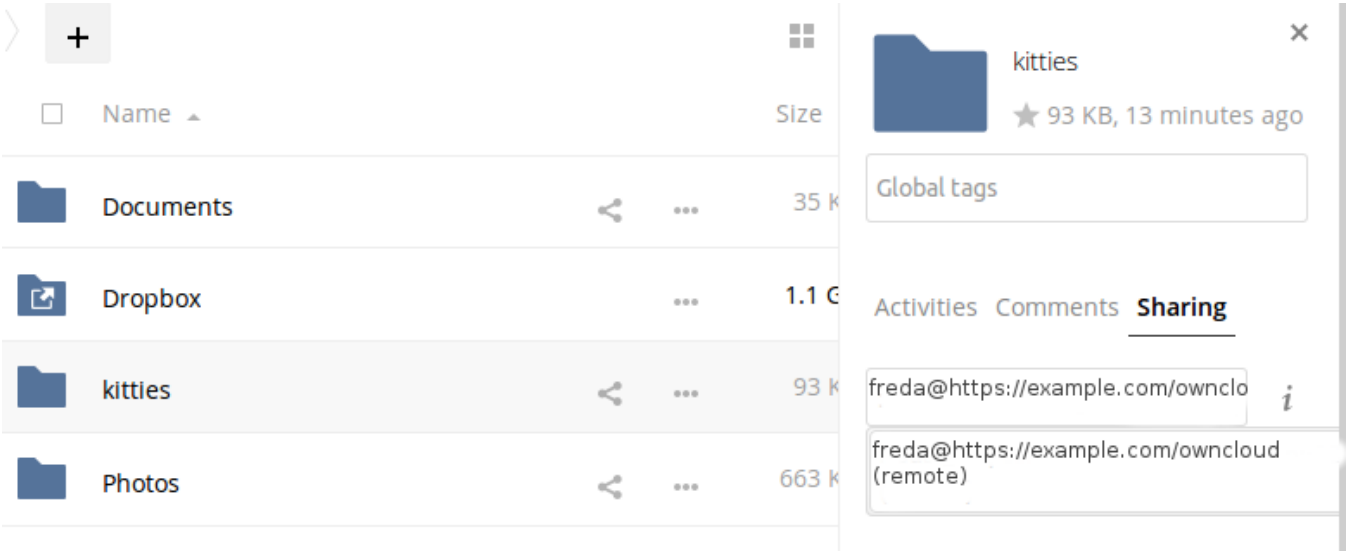
Création d'un partage fédéré (9.0+ seulement)

Suivez ces étapes pour créer un nouveau partage fédéré ebntre deux serveurs ownCloud 9.0+. Ceci ne nécessite pas d'action de la part de l'utilisateur sur le serveur distant. Tout se passe sur le serveur d'origine.

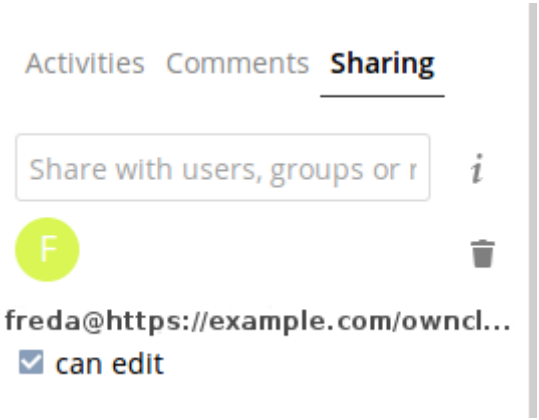
1. Activez l'application Fédération.
2. Rendez-vous dans l'administration d'ownCloud à la section « Partage ». Vérifiez que les options **Autoriser les utilisateurs de ce serveur à envoyer des partages vers d'autres serveurs** et **Autoriser les utilisateurs de ce serveur à recevoir des partages d'autres serveurs** soient activées.
3. Rendez-vous maintenant dans la section « Fédération ». Par défaut, **Add server automatically once a federated share was created successfully** est coché. L'application Fédération gère la création d'une liste de serveurs ownCloud de confiance, qui permet à ces serveurs d'échanger des répertoires utilisateur et d'effectuer l'auto-complétion des noms des utilisateurs externes lors de la création de partages. Si vous ne voulez pas activer cette fonctionnalité, décochez la case.



4. Puis, rendez-vous sur votre page Fichiers et sélectionnez un dossier à partager. Cliquez sur l'icône de partage, puis saisissez le nom d'utilisateur et l'URL de l'utilisateur sur le serveur ownCloud distant. Dans cet exemple, c'est : `freda@https://example.com/owncloud`. Quand ownCloud vérifie le lien, il l'affiche avec le libellé **(distant)**. Cliquez sur ce libellé pour établir le lien.



5. Une fois le lien vérifié, vous disposez d'une seule option de partage qui est **Peut modifier**.

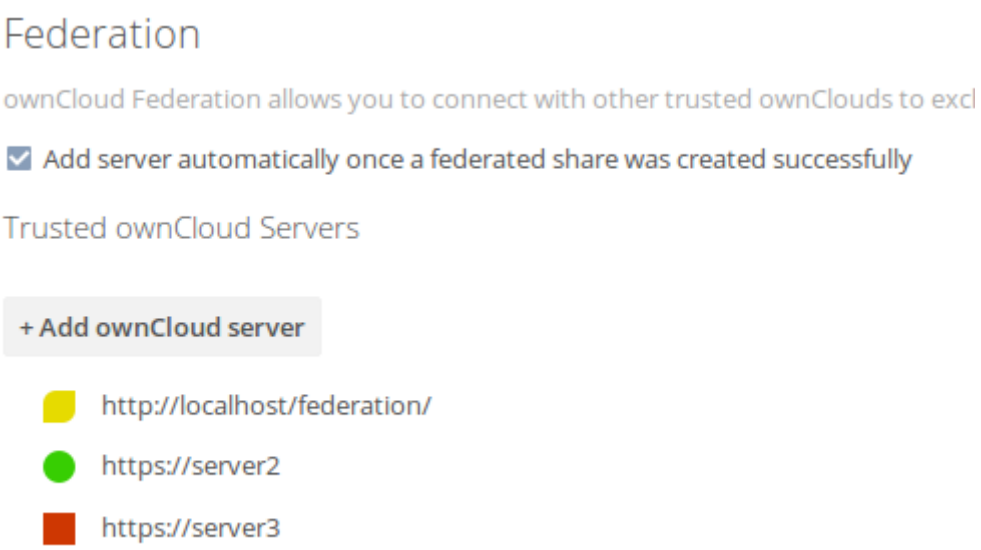


Vous pouvez déconnecter le partage à tout moment en cliquant sur l'icône représentant une corbeille.

Configuration des serveurs ownCloud de confiance

Vous pouvez créer une liste de serveurs ownCloud de confiance pour le partage fédéré. Ceci permet d'autoriser tous les serveurs ownCloud liés à partager les répertoires utilisateurs et à compléter automatiquement les noms d'utilisateurs dans les dialogues de partage. Si **Add server automatically once a federated share was created successfully** est activé sur la page d'administration, les serveurs seront automatiquement ajoutés à votre liste de confiance lors de la création de nouveaux partages fédérés.

Vous pouvez également ajouter les URL de serveurs ownCloud dans le champ **Add ownCloud Server**. La lumière jaune indique une connexion réussie, sans échange de noms d'utilisateur. La lumière verte indique une connexion réussie avec échange de noms d'utilisateurs. Une lumière rouge indique que la connexion a échoué.

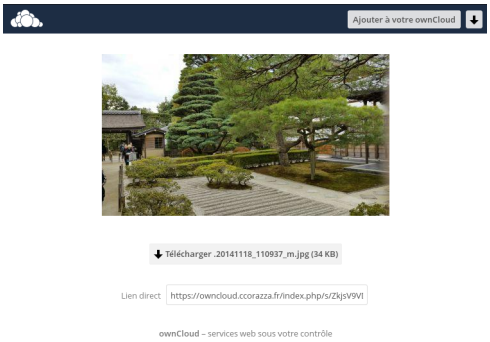


Création de partages fédérés à l'aide du partage par lien public

Vous devrez utiliser le partage par lien public avec ownCloud 8.x ou ses versions précédentes.

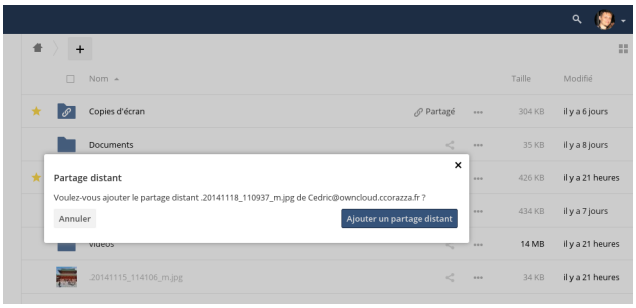
Cochez la case `Partager par lien public` pour afficher plus d'options de partage (qui sont décrites plus en détails dans *Partage de fichiers*). Vous pouvez créer un partage fédéré en autorisant ownCloud à créer un lien public pour vous et à envoyer un courriel à la personne avec laquelle vous voulez partager.

De manière facultative, vous pouvez définir un mot de passe et une date d'expiration. Quand votre correspondant recevra votre courriel, il devra cliquer sur le lien ou le copier dans un navigateur Web. Il verra alors une page affichant une imagette du fichier avec un bouton **Ajouter à votre ownCloud**.



Votre correspondant devra cliquer sur le bouton **Ajouter à votre ownCloud**. Sur l'écran suivant, il devra saisir l'URL de son serveur ownCloud puis appuyer sur la touche « Entrée ».

Votre correspondant devra effectuer une étape supplémentaire en confirmant la création du partage fédéré en cliquant sue le bouton **Ajouter un partage distant**.



Cochez la case `Partager par lien public` pour désaciver tout partage fédéré créé de cette manière.

Astuces de configuration

La section Partage de la page d'administration permet de contrôler la façon dont vos utilisateurs peuvent gérer les partages fédérés :

- Cocher la case `Imposer la protection par mot de passe` pour rendre obligatoires les mots de passe pour les partages.
- Cocher la case `Spécifier une date d'expiration par défaut` pour rendre obligatoire la date d'expiration des partages.
- Cocher la case `Autoriser les téléversements publics` pour autoriser les partages de fichiers bi-directionnels.

Votre serveur Apache doit avoir le module `mod_rewrite` activé et `trusted_domains` doit être correctement configuré dans `config.php` pour autoriser les connexions externes (voir *Assistant d'installation*). Envisagez aussi d'activer SSL pour chiffrer tout le trafic entre vos serveurs.

Votre serveur ownCloud crée le lien de partage à partir de l'URL que vous avez utilisée pour vous connecter au serveur .Assurez-vous donc de vous connecter en utilisant une URL accessible pour vos utilisateurs. Par exemple, si vous vous connectez à l'aide de l'adresse IP locale du serveur telle que `http://192.168.10.50`, votre URL de partage reseemblera à `http://192.168.10.50/owncloud/index.php/s/jWfCfTVztGlWTJe`, qui n'est pas accessible en dehors de votre réseau local. Ceci vaut également pour le nom du serveur : pour accéder au serveur en dehors de votre réseau local, vous devez utiliser le nom complet du serveur, tel que `http://monserveur.exemple.com`, au lieu de `http://monserveur`.

Téléversement de gros fichiers > 512 Mo

La taille de fichier maximale par défaut pour les téléversements est de 512 Mo. Vous pouvez augmenter cette limite en fonction de ce que permet votre système d'exploitation. Certaines limites matérielles ne peuvent être dépassées :

- < 2 Go sur une architecture 32 bits
- < 2 Go sur Windows (32 bits et 64 bits)
- < 2 Go avec un serveur en version 4.5 ou précédente
- < 2 Go avec Internet Explorer 6 à 8
- < 4 Go avec Internet Explorer 9 à 11

Les systèmes de fichiers 64 bits ont des limites beaucoup plus hautes. Veuillez consulter la documentation de votre système d'exploitation pour plus de détails.

Note

Le client de synchronisation ownCloud n'est pas affecté par ces limites car il téléverse les fichiers par petits bouts.

Configuration système

- Assurez-vous d'avoir la dernière version de PHP installée (au moins 5.4.9) ;
- désactivez les quotas utilisateur pour les rendre illimités ;
- votre répertoire ou partition temporaire doit être suffisamment grande pour accepter des téléversements parallèles de plusieurs utilisateurs. Par exemple si la taille de téléversement maximale est de 10 Go et que le nombre moyen d'utilisateurs effectuant des téléversements en même temps est de 100, l'espace temporaire doit être d'au moins : 100 × 10 Go.

Configuration du serveur Web

Note

ownCloud fournit son propre fichier `owncloud/.htaccess`. Parce que `php-fpm` ne peut pas lire les paramètres PHP dans `.htaccess`, ces paramètres doivent être définis dans le fichier `owncloud/.user.ini`.

Définissez les deux paramètres suivants dans le fichier php.ini correspondant (consultez la section **Fichier de configuration chargé** de *Informations et version de PHP* pour trouver les fichiers php.ini correspondants)

```
php_value upload_max_filesize = 16G
php_value post_max_size = 16G
```

Ajustez ces valeurs selon vos besoins. Si vos observez des délais de temporisation dépassés dans vos fichiers journaux, augmentez ces valeurs, exprimées en secondes:

```
php_value max_input_time 3600
php_value max_execution_time 3600
```

Le module Apache mod_reqtimeout peut aussi empêcher les gros téléversements de se terminer. Si vous utiliser ce module et que les téléversements de gros fichiers échouent, désactivez ce module dans la configuration d'Apache ou augmentez la valeur `RequestReadTimeout`.

Il existe aussi plusieurs autres options de configuration de votre serveur Web qui peuvent empêcher le téléversement de gros fichiers. Veuillez consulter la documentation de votre serveur Web pour savoir comment configurer ces valeurs correctement :

Apache

- LimitRequestBody
- SSLRenegBufferSize

Apache avec mod_fcgid

- FcgidMaxRequestInMem
- FcgidMaxRequestLen

Note

Si vous utilisez Apache 2.4 avec mod_fcgid, à ce jour, février/mars 2016, `FcgidMaxRequestInMem` nécessite encore d'être augmenté significativement par rapport à sa valeur par défaut pour éviter les erreurs de segmentation lors des téléversements de gros fichiers. Ce n'est pas le paramètre adapté mais sert de moyen de contournement pour le bogue 51747 : Apache avec mod_fcgid.

L'augmentation significative de la valeur de `FcgidMaxRequestInMem` ne sera peut-être plus nécessaire une fois le bogue 51747 corrigé.

nginx

- client_max_body_size
- fastcgi_read_timeout
- client_body_temp_path

Depuis nginx 1.7.11, une nouvelle option de configuration fastcgi_request_buffering est disponible. Définir cette option à `fastcgi_request_buffering off;` dans votre configuration nginx pourrait vous aider avec les délais de temporisation (timeout) pendant le téléchargement. De plus, cela peut aider si vous manquez d'espace disponible sur la partition /tmp de votre système.

Pour plus d'informations sur la configuration de nginx pour augmenter la taille limite des téléversements, consultez également cette page du wiki.

Note

Assurez-vous que `client_body_temp_path` pointe vers une partition avec une taille adaptée pour votre taille de fichier téléversé, et sur la même partition que `upload_tmp_dir` ou `tempdirectory` (voir ci-dessous). Pour des performances optimales, placez-les sur des disques différents dédiés à l'espace d'échange et au stockage temporaire.

Si votre site se trouve derrière un frontal nginx (pour de l'équilibrage de charge par exemple) :

Par défaut, les téléchargements seront limités à 1 Go à cause de `proxy_buffering` et `proxy_max_temp_file_size` sur le frontal.

- Si vous avez accès à la configuration du frontal, désactivez proxy_buffering ou augmentez proxy_max_temp_file_size la valeur par défaur de 1 Go.
- Si vous n'avez pas accès à la configuration du frontal, définissez l'en-tête X-Accel-Buffering à `add_header X-Accel-Buffering no;` sur votre site.

Configuration de PHP

Si vous ne voulez pas utiliser les fichiers `.htaccess` ou `.user.ini` d'ownCloud, vous pouvez configurer à la place PHP. Assurez-vous de mettre en commentaire les lignes `.htaccess` concernant la taille de téléversement le cas échéant.

Si vous utilisez ownCloud sur un système 32 bits, la directive `open_basedir` de votre fichier `php.ini` décommentée.

Ajustez les deux paramètres suivants du fichier `php.ini` selon vos besoins:

```
upload_max_filesize = 16G
post_max_size = 16G
```

Indiquez à PHP le répertoire temporaire que vous voulez utiliser:

```
upload_tmp_dir = /var/big_temp_file/
```

Output Buffering doit être désactivé dans les fichiers `.htaccess`, `.user.ini` ou `php.ini`, sans quoi, PHP renverra des erreurs relatives à la mémoire :

- `output_buffering = 0`

Configuration d'ownCloud

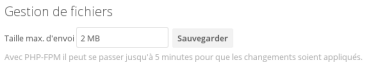
Une alternative à l'utilisation du paramètre PHP `upload_tmp_dir` est possible (par exemple, si vous n'avez pas accès au fichier `php.ini`). Vous pouvez configurer le répertoire temporaire pour les fichiers téléversés en utilisant le paramètre `tempdirectory` du fichier `config.php` (voir *Paramètres de Config.php*).

Si vous avez configuré le paramètre `session_lifetime` dans le fichier `config.php` (voir *Paramètres de Config.php*), assurez-vous qu'il n'est pas trop bas. La valeur de ce paramètre (en secondes) doit être au moins équivalente à la durée que prend le téléversement du fichier le plus long. Si vous n'êtes pas sûr, enlevez-le

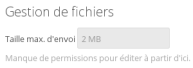
complètement de votre configuration en le réinitialisant à sa valeur par défaut, tel qu'indiqué dans le fichier `config.sample.php`.

Configuration des limites dans l'interface graphique

Si tous les prérequis indiqués dans cette documentation sont en place, un administrateur peut modifier les limites pour la taille des téléversements en utilisant la boîte de saisie `Gestion de fichiers` dans l'interface d'administration d'ownCloud.



Selon votre environnement, vous pourriez avoir un message indiquant que vous n'avez pas les autorisations suffisantes pour modifier ce paramètre.



Pour pouvoir modifier ce paramètre dans l'interface graphique, vous devez vous assurez que :

- le serveur Web peut utiliser le fichier `.htaccess` fourni par ownCloud (Apache seulement) ;
- l'utilisateur utilisé pour exécuter le serveur Web ait les permissions d'écriture sur les fichiers `.htaccess` et `.user.ini`

La *Renforcement des permissions de répertoires* peut empêcher l'accès en écriture à ces fichiers. En tant qu'administrateur, il vous appartient de décider si vous voulez avoir la possibilité d'utiliser cette boîte de saisie ou une installation d'ownCloud plus sécurisée où vous devrez modifier manuellement cette valeur dans les fichiers `.htaccess` et `.user.ini` décrits ci-dessus.

Problèmes généraux de téléversements

Divers facteurs peuvent provoquer une restriction de la taille de téléversement. Par exemple :

- `LVE Manager` de `CloudLinux` définit une limite `I/O limit` ;
- certains services comme `Cloudflare` sont aussi connus pour provoquer des problèmes de téléversements ;
- les limites de téléversements mises en œuvre par les serveurs proxy utilisés par vos clients ;
- d'autres modules de serveurs Web comme ceux décrits dans *Dépannage général*.

Configuration de l'application collaborative Documents

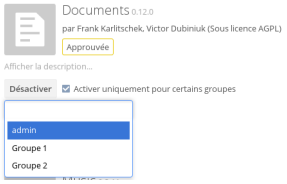
L'application Documents gère l'édition de documents dans ownCloud, sans avoir besoin de lancer une application externe. L'application Documents propose les fonctionnalités suivantes :

- l'édition collaborative de documents par plusieurs utilisateurs ;
- la création de document dans ownCloud ;
- le téléversement de documents ;
- le partage et la modification de fichiers dans le navigateur Web, et leurs partage dans ownCloud ou à l'aide d'un lien public.

Les formats de fichiers pris en charge sont : *.odt*, *.doc* et *.docx*. *.odt* est géré nativement dans ownCloud, et vous devez avoir LibreOffice ou OpenOffice installé sur le serveur ownCloud pour convertir les documents *.doc* et *.docx*.

Activation de l'application Documents

Rendez-vous sur la pages Applications et cliquez sur le bouton `Activer`. Vous pouvez aussi autoriser l'accès à l'application Documents pour certains groupes seulement. Par défaut, elle est disponible pour tous les groupes.



Consulter la section « Édition collaborative de documents » dans le manuel utilisateur pour apprendre comment créer et partager des documents avec l'application Documents.

Activation et test de la prise en charge de MS Word

Rendez-vous dans le menu d'administration. Après avoir choisi `Local` ou `Externe` cliquez sur le bouton `Appliquer et essayer`. Si vous avez une installation de LibreOffice ou OpenOffice opérationnelle, une icône verte `Sauvegardé` devrait apparaître.

Dépannage

Si les tests mentionnés échouent, veuillez vous assurer que :

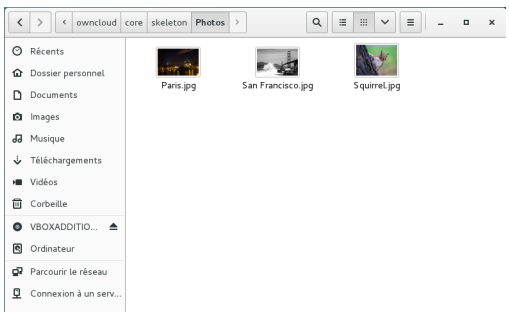
- les fonctions PHP `escapeshellarg` et `shell_exec` ne soient pas désactivées dans votre configuration PHP ;
- le binaire de libreoffice/openoffice se trouve dans le « PATH » et que l'utilisateur du serveur Web peut l'exécuter ;
- la configuration SELinux ne bloque pas l'exécution du binaire ;
- le paramètre PHP `open_basedir` est correctement configuré pour autoriser l'accès au binaire.

Vous pouvez trouver d'autres indices sur les dysfonctionnement dans le fichier `data/owncloud.log`.

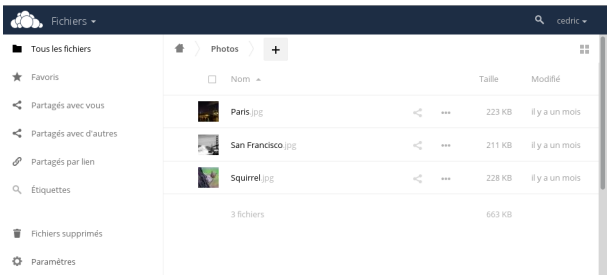
Mise à disposition de fichiers par défaut

vous pouvez distribuer un ensemble de fichiers et de répertoires par défaut à tous les utilisateurs en les plaçant dans le répertoire owncloud/core/skeleton de votre serveur ownCloud. Ces fichiers apparaîtront seulement pour les nouveaux utilisateurs lors de leur première connexion. Les utilisateurs existants ne verront pas les nouveaux fichiers si ceux-ci ont été copiés dans ce répertoire après leur première connexion. Les fichiers du répertoire skeleton sont copiés dans le répertoire « data » de chaque utilisateur. Ils peuvent donc les modifier ou les supprimer sans affecter les fichiers originaux.

Cette copie d'écran montre un ensemble de photos dans le répertoire skeleton.



Ils apparaissent sur la page Fichiers ownCloud de l'utilisateur comme n'importe quel autre fichier.



Configuration additionnelle

L'option de configuration `skeletondirectory` disponible dans le fichier `config.php` (voir *Paramètres de Config.php*), permet de configurer le répertoire où seront placés les fichiers à distribuer. Ces fichiers seront copiés dans le répertoire data des nouveaux utilisateurs. Laisser le répertoire vide pour ne copier aucun fichier.

Configuration du stockage externe (interface graphique)

L'application The External Storage Support permet d'accéder à des services de stockage externe ou à des équipements de stockage secondaire d'ownCloud. Vous pouvez aussi autoriser les utilisateurs à monter leurs propres services de stockage externe.

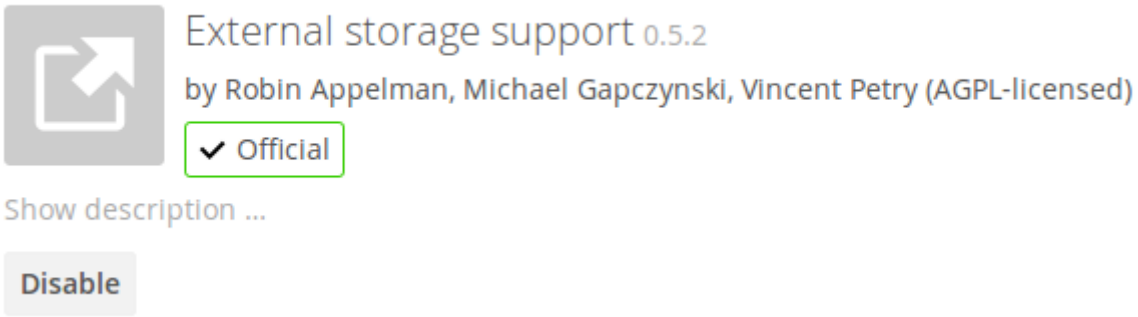
ownCloud 9.0 propose un nouveau jeu de *commandes occ pour la gestion du stockage externe*.

Nouveau également dans la version 9.0, une option pour les administrateurs d'ownCloud permettant d'activer ou de désactiver le partage sur des points de montage externes individuels (voir *Options de montage*). Ce type de partage est désactivé par défaut.

Activation de la gestion du stockage externe

> **Warning**
>
> Activer cette application désactive la case **Rester connecté** sur la page de connexion.

L'application External Storage Support s'active sur la page Applications.

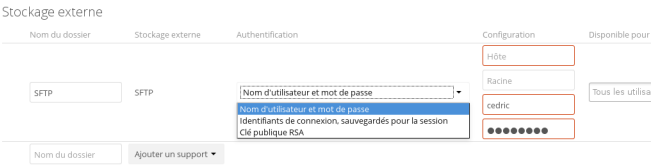


Configuration du stockage

Pour créer un nouveau point de montage externe, sélectionnez un service dans la liste déroulante **Ajouter un support**. Chaque service nécessite différentes options qui peuvent être définies dans les champs de configuration.

Chaque service peut accepter plusieurs méthodes d'authentification. Elles peuvent être sélectionnées dans la liste déroulante **Authentification**. Les différents services gèrent différents mécanismes d'authentification. Certains sont spécifiques à unn service, d'autres sont plus génériques. Voir *Mécanismes d'authentification pour le stockage externe* pour des informations détaillées.

Quand vous sélectionnez un mécanisme d'authentification, les champs de configuration s'adaptent au mécanisme. Le service SFTP, par exemple, gère **Nom d'utilisateur et mot de passe**, **Identifiants de connexion, sauvegardés pour la session** et **Clé publique RSA**.

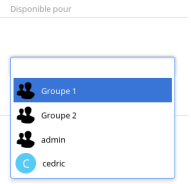


Les champs obligatoires sont identifiés par les champs dont la bordure est rouge. Quand tous les champs obligatoires ont été saisis, le stockage est automatiquement enregistré. Un point vers en regard de la ligne du stockage indique que le stockage est prêt à être utilisé. Une icône rouge ou jaune indiqque qu'ownCloud n'arrive pas à se connecter. Vous devez donc vérifier votre configuration ou la connectivité réseau.

S'il y a une erreur sur le stockage, il sera marqué indisponible pendant dix minutes. Pour vérifier à nouveau, cliquez sur l'icône colorée ou rechargez la page d'administration.

Permissions utilisateur et groupe

Un stockage configuré dans la page Personnel d'un utilisateur n'est disponible que pour l'utilisateur qui l'a créé. Un stockage créé dans la page d'administration est disponible pour tous les utilisateurs par défaut et peut être restreint à des utilisateurs et des groupes spécifiques dans le champ **Disponible pour**.



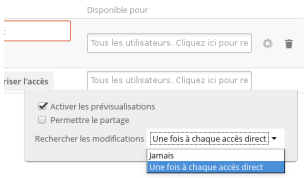
Options de montage

Passez la souris sur la droite d'une configuration de stockage pour dévoiler les boutons de paramètres et la corrbeille. Cliquez sur la corbeille pour supprimer le point de montage. Le bouton de paramètres permet de configurer chaque point de montage individuellement avec les options suivantes :

- Chiffrement
- Activer les prévisualisations
- Permettre le partage
- Rechercher les modifications (Jamais, Une fois à chaque accès direct)

La case à cocher **Chiffrement** n'est visible que lorsque l'application Encryption est activée.

Permettre le partage permet à l'administrateur d'ownCloud d'activer ou de désactiver le partage des points de montage individuels. Quand le partage est désactivé, les partages sont conservés en interne, de sorte que lorsque vous les réactiver, ils sont de nouveau disponibles. Le partage est désactivé par défaut.



Utilisation de certificat auto-signé

Lors de l'utilisation de certificat auto-signé pour les points de montage externes, le certificat doit être importé dans ownCloud. Veuillez consulter *Importation de certificats SSL personnel ou global* pour plus d'informations.

Services de stockages disponibles

Les services suivants sont proposés par les applications de stockage externe. D'autres applications peuvent proposer leurs propres services, qui ne sont pas indiqués ici.

Amazon S3

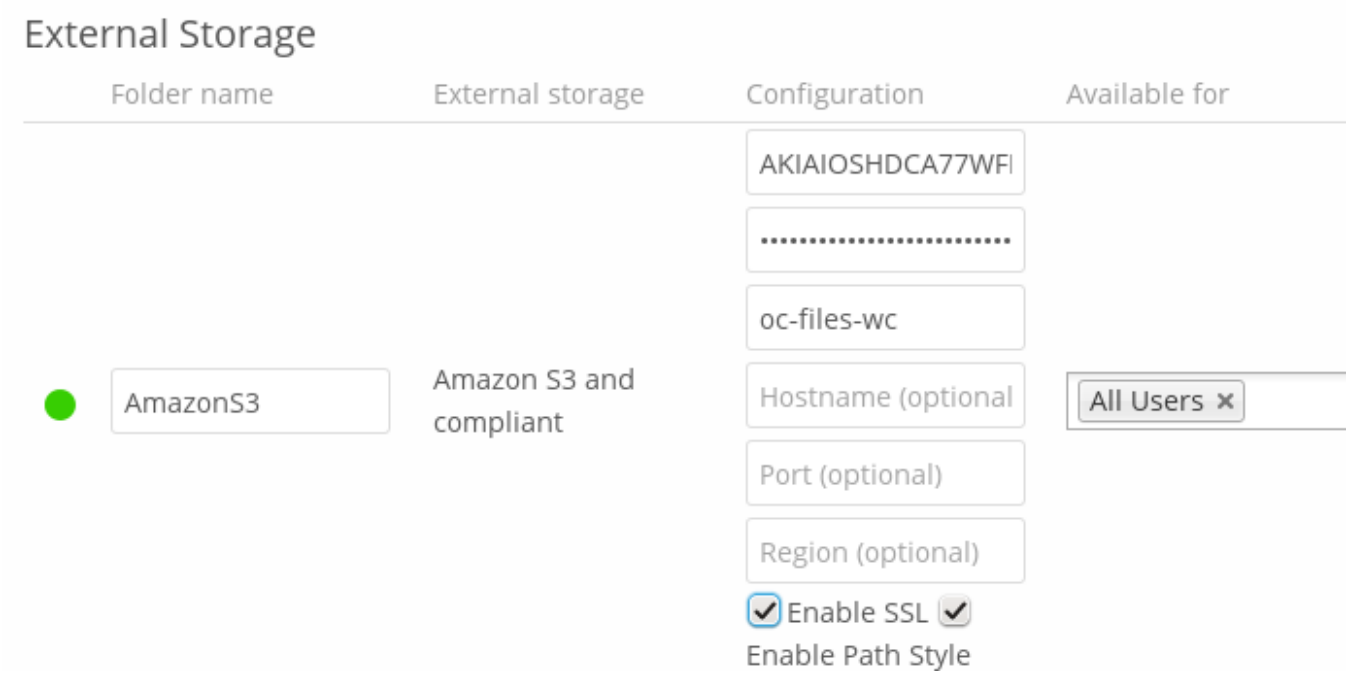
Pour connecter vos compartiments (buckets) Amazon S3 à ownCloud, vous aurez besoin :

- d'une clé d'accès S3 ;
- d'un mot de passe S3 ;
- d'un nom de compartiment.

Dans le champ **Nom du dossier**, saisir un nom de dossier local pour votre point de montage S3. S'il n'existe pas, il sera créé.

Dans le champ **Disponible pour**, saisir les utilisateurs ou les groupes ayant la permission d'accéder à votre point de montage S3.

La case à cocher `Activer SSL` active les connexions HTTPS. Utiliser HTTPS est toujours fortement recommandé.



De manière facultative, vous pouvez écraser le nom d'hôte, le port et la région de votre serveur S3, ce qui est nécessaire pour les serveurs non Amazon tels que Ceph Object Gateway.

Accès par path n'est habituellement pas nécessaired (et d'ailleurs, c'est en fait incompatible avec les nouveaux centres de données d'Amazon), mais peut être utilisé avec des serveurs non Amazon pour lesquels l'infrastructure DNS ne ne peut être contrôlée. Ordinairement, les requêtes seront faites avec

`http://bucket.hostname.domain/`, mais avec « Accès par path » activé, les requêtes seront faites avec `http://hostname.domain/bucket`.

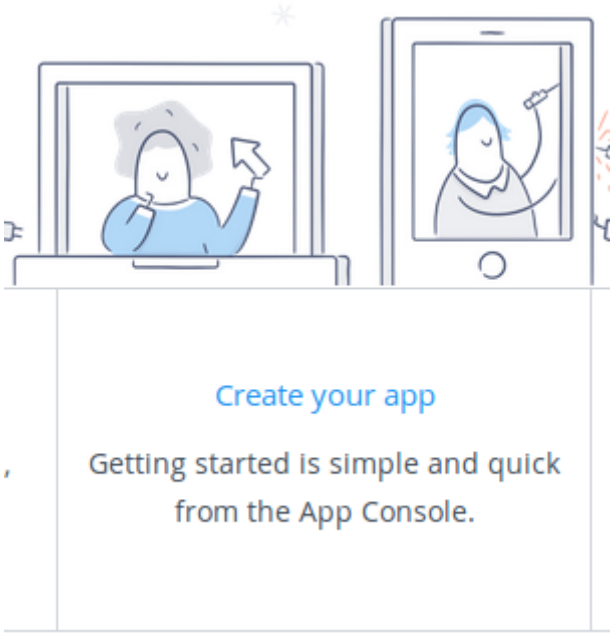
Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage supplémentaires et pour des informations.

Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les schémas d'authentification.

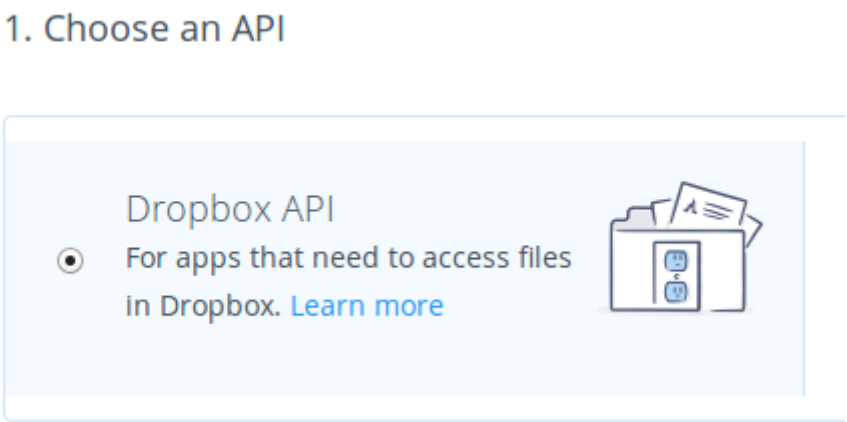
Dropbox

Bien que Dropbox sache gérer le nouveau OAuth 2.0, ownCloud utilise OAuth 1.0. Vous pouvez donc ignorer toute référence à OAuth 2.0 dans la configuration de Dropbox.

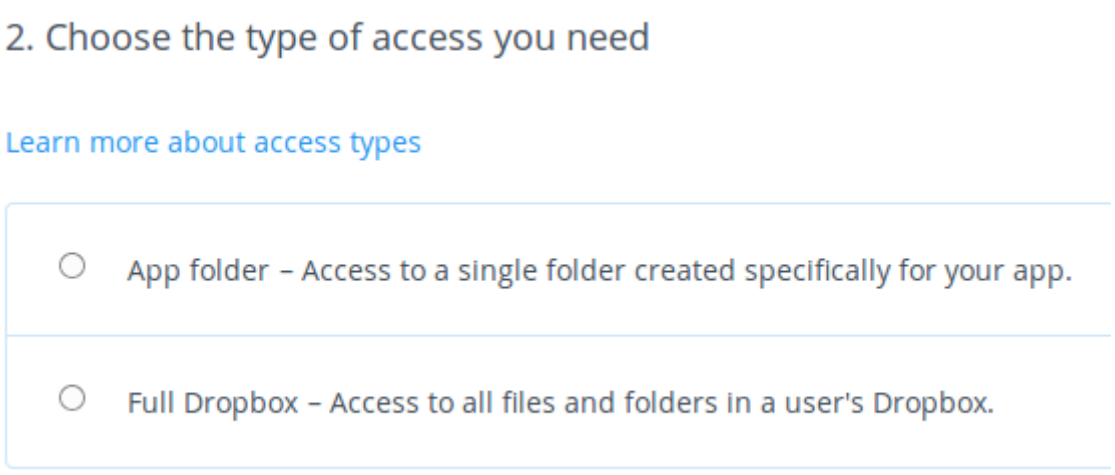
La connexion à Dropbox nécessite un peu plus de travail car vous devez créer une application Dropbox. Connectez-vous sur la page des développeurs de Dropbox et cliquez sur **Create Your App** :



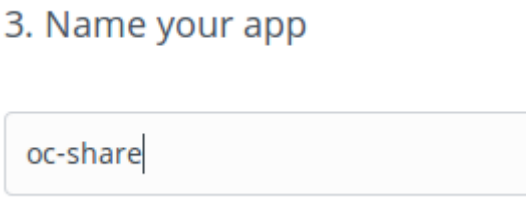
Ensuite, pour **Choose an API** cochez **Dropbox API**.



L'option suivante permet de choisir les dossiers à partager, ou tout partager dans votre Dropbox.

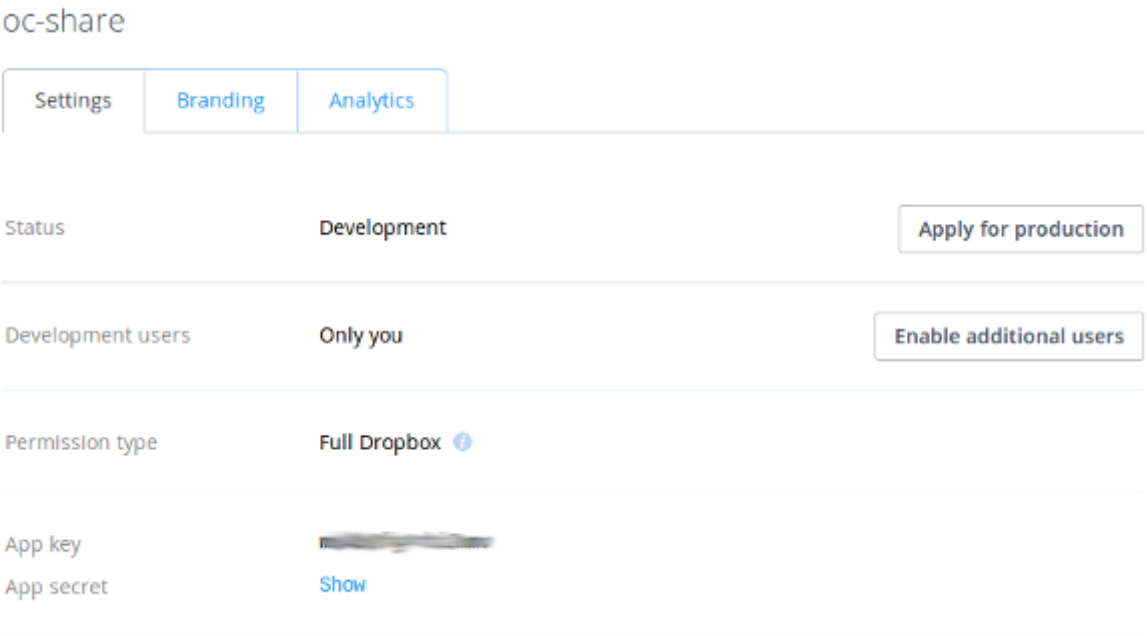


Saisissez ensuite votre nom d'application. Vous pouvez choisir tout ce que vous voulez.



Cliquez ensuite sur le bouton **Create App**.

Vous êtes maintenant sur la page de votre application, qui affiche les paramètres et les options. Ne cliquez pas sur **Development (Apply for production)** car ceci est destiné aux applications que vous voulez rendre publiques.

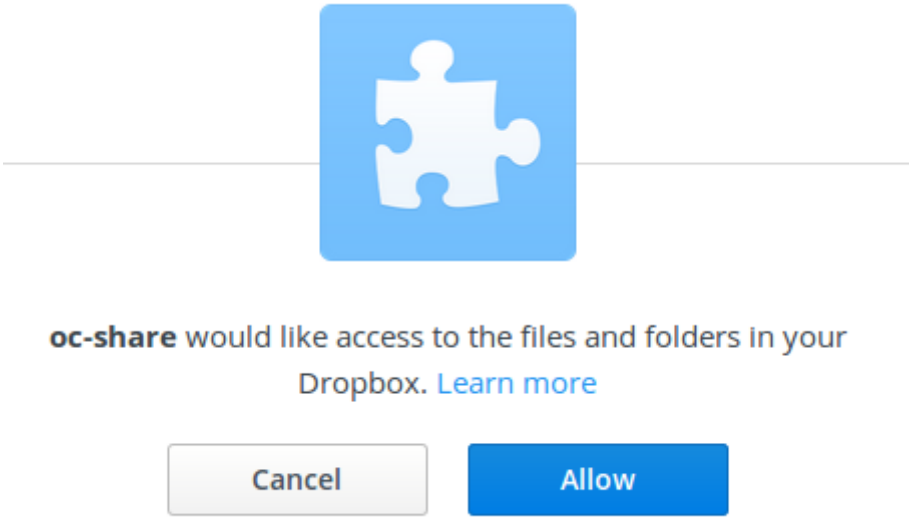


Cliquez sur **Enable additional users** pour permettre à plusieurs utilisateurs d'ownCloud d'accéder à votre nouveau partage Dropbox.

Rendez-vous maintenant sur la page d'administration d'ownCloud. Votre configuration ownCloud nécessite seulement le nom du montage local, la **Clé d'application** et le **Mot de passe de l'application** et les utilisateurs ou groupes ayant accès au partage. Cliquez sur l'icône représentant un engrenage à droite pour définir des options supplémentaires.

Après avoir saisi le nom du point de montage local et les champs **App Key** et **App Secret**, cliquez sur **Autoriser l'accès**.

Si vous n'êtes pas déjà connecté à Dropbox, vous serz invité à vous connecter et à autoriser l'accès. Ceci n'arrive qu'une seule fois, lors de la création du nouveau partage. Cliquez sur **Autoriser** et c'est fini.



Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

FTP/FTPS

Pour se connecter à un serveur FTP, les informations suivantes sont nécessaires :

- un nom de dossier pour le point de montage local (le dossier sera créé s'il n'existe pas) ;
- l'URL du serveur FTP ;
- le numéro de port (par défaut : 21) ;
- le nom d'utilisateur FTP et son mot de passe ;
- le nom du dossier distant (le répertoire FTP à monter dans ownCloud). Par défaut, ownCloud indique le répertoire racine. Si vous précisez un sous-dossier, vous ne devez pas laisser de barre de fraction (« / ») à la fin du chemin. Par exemple : `public_html/images`.

Votre nouveau point de montage est disponible par défaut pour tous les utilisateurs. Vous pouvez restreindre l'accès en indiquant les utilisateurs ou groupes autorisés dans le champ **Disponible pour**.

De manière facultative, ownCloud peut utiliser FTPS (FTP avec SSL) en cochant la case **Sécurisation ftps://**. Ceci nécessite une configuration complémentaire avec votre certificat racine si le serveur FTP utilise un certificat auto-signé (voir *Importation de certificats SSL personnel ou global*).

> ***Note***
>
> Le stockage externe `FTP/FTPS` nécessite que le paramètre PHP `allow_url_fopen` soit défini à `1`. Si vous rencontrez des problèmes de connexion, assurez-vous que ce paramètre ne soit pas défini à `0` dans votre fichier `php.ini`. Consulter *Informations et version de PHP* pour en apprendre plus sur la façon de trouver le bon fichier `php.ini` à modifier.

Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

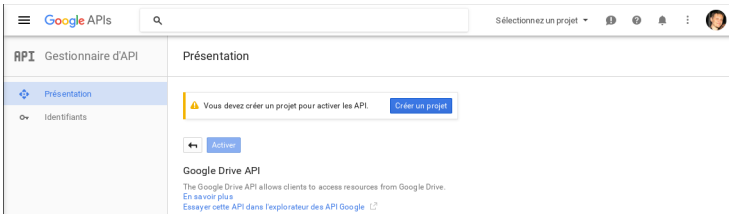
FTP utilise le mécanisme d'authentification par mot de passe ; consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

Google Drive

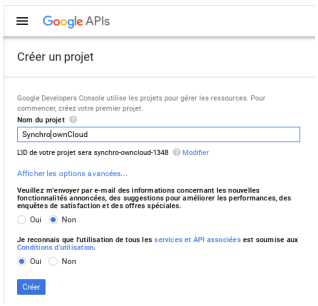
ownCloud utilise OAuth 2.0 pour se connecter à Google Drive. Ceci nécessite de la configuration chez Google pour obtenir un identifiant d'application et un mot de passe, car ownCloud l'enregistre comme une application.

Toutes les applications qui accède à une API Google doivent être enregistrées dans Google Cloud Console. Suivez soigneusement les indications car l'interface de Google est très dense et il est facile de s'y perdre.

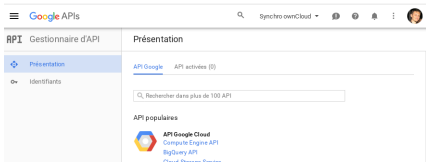
Si vous avez déjà un compte Google, pour Groups, Drive ou Mail, vous pouvez utiliser votre compte existant pour vous connecter dans Google Cloud Console. Après la connexion, cliquez sur le bouton **Créer un projet**.



Donner un nom à votre projet et acceptez l'ID par défaut ou créez le vôtre en cliquant sur **Modifier**, puis cliquer sur le bouton **Créer**.

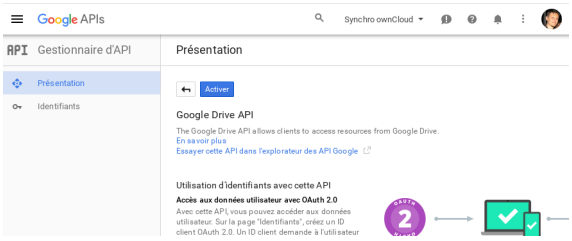


Vous êtes alors renvoyé sur votre tableau de bord.



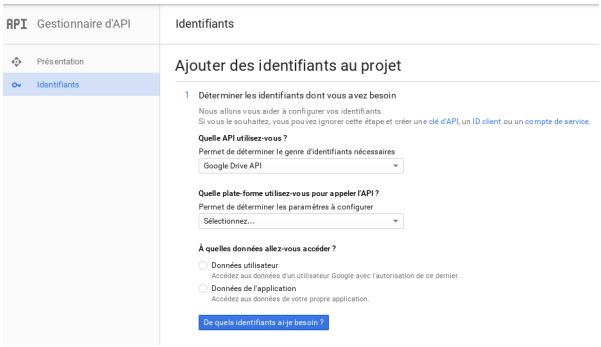
Il y a beaucoup d'API Google ; Cherchez la section **API Google Apps** et cliquez sur **Drive API**.

Drive API vous amène sur l'écran de gestion des API. Cliquez sur le bouton **Activer**.

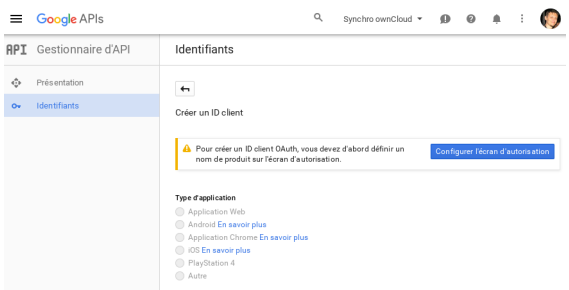


Vous devez maintenant créer vos identifiants. Cliquez alors sur **Accéder à "Identifiants"**.

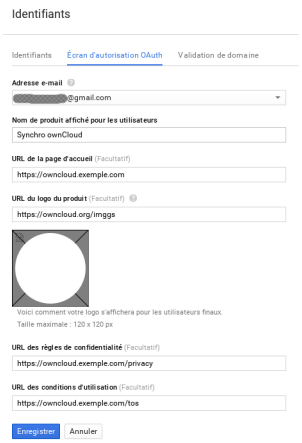
Google avertit que vous devez créer des identifiants. Nous utiliserons 0Auth 2.0.



Cliquez sur le lien **ID client**.



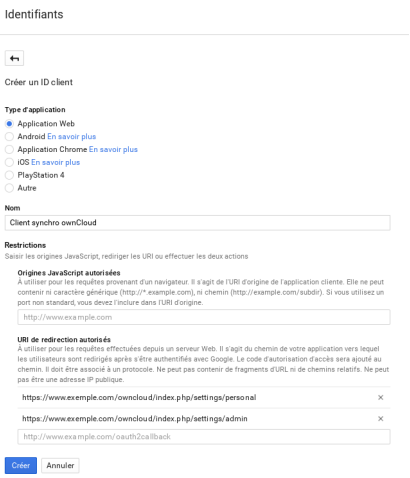
Cliquez alors sur le bouton **Configurer l'écran d'autorisation**. Il s'agit de l'écran d'information que Google affiche lors de la première connexion de votre nouvelle application à ownCloud. Remplissez alors les champs du formulaire. Votre logo doit être hébergé car vous ne pouvez pas le téléverser : saisissez son URL. Quand tous les champs sont saisis, cliquez sur le bouton **Enregistrer**.



L'écran **Créer un ID client** s'affiche alors. Cochez le bouton radio **Application Web** et saisissez le nom de votre application. **Origines JavaScript autorisées** est l'URL de la racine de votre domaine, par exemple : `https://www.exemple.com` sans barre de fraction (« / ») à la fin. Vous avez besoin de deux **URI de redirection autorisés** qui doivent être sous cette forme:

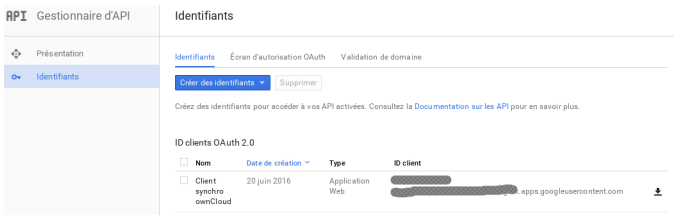
```
https://www.exemple.com/owncloud/index.php/settings/personal
https://www.exemple.com/owncloud/index.php/settings/admin
```

Remplacez `https://www.exemple.com/owncloud/` par l'URL de votre serveur ownCloud, puis cliquez sur le bouton **Créer**.



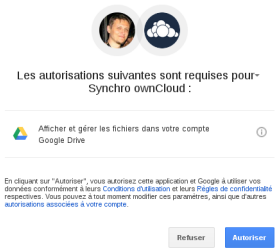
Google affiche maintenant votre **ID client** et votre **Code secret client**. Cliquez sur le bouton **OK**.

Vous pouvez les retrouver à tout moment dans votre console Google ; il suffit de cliquer sur le nom de votre application.



Vous disposez maintenant de tout ce qui est nécessaire pour monter votre Drive Google dans ownCloud.

Rendez-vous dans la section Stockage externe de votre page d'administration, créez un nouveau nom de dossier et saisissez votre ID client et votre code secret client, puis, cliquez sur **Autoriser l'accès**. Votre page d'autorisation apparaît lorsque ownCloud réussit à établir la connexion. Cliquez alors sur **Autoriser**.



Quand la lumière verte apparaît confirmant une connexion réussie, c'est terminé.

Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

Local

Les stockage locaux fournissent un accès à tout répertoire sur le serveur ownCloud. Puisque c'est un risque de sécurité significatif, le stockage local ne peut être configuré que dans la page d'administration d'ownCloud. Les utilisateurs non administrateur ne peuvent pas créer de point de montage local.

Utilisez ce qui suit pour monter un répertoire de votre serveur ownCloud se trouvant en dehors du répertoire `data/` d'ownCloud. Ce répertoire doit être accessible en lecture et en écriture par l'utilisateur de votre serveur HTTP. Ci-après, un exemple de permissions et de propriétaire pour un système Ubuntu Linux:

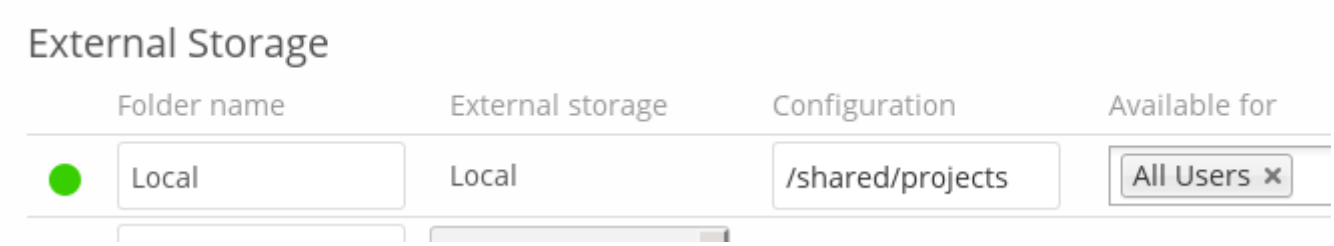
```
sudo -u www-data chown -R www-data:www-data /localdir
sudo -u www-data chmod -R 0750 /localdir
```

Consulter *Renforcement des permissions de répertoires* pour des informations sur les permissions de fichier et comment trouver l'utilisateur HTTP *Informations et version de PHP*.

Dans le champ **Nom du dossier**, saisissez le nom du dossier que vous voulez voir apparaître sur votre page Fichiers dans ownCloud.

Dans le champ **Configuration**, saisissez le chemin d'accès complet au répertoire que vous voulez monter.

Dans le champ **Disponible pour**, indiquez les utilisateurs ou groupes ayant la permission d'accéder au montage. Par défaut, tous les utilisateurs ont accès.



Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

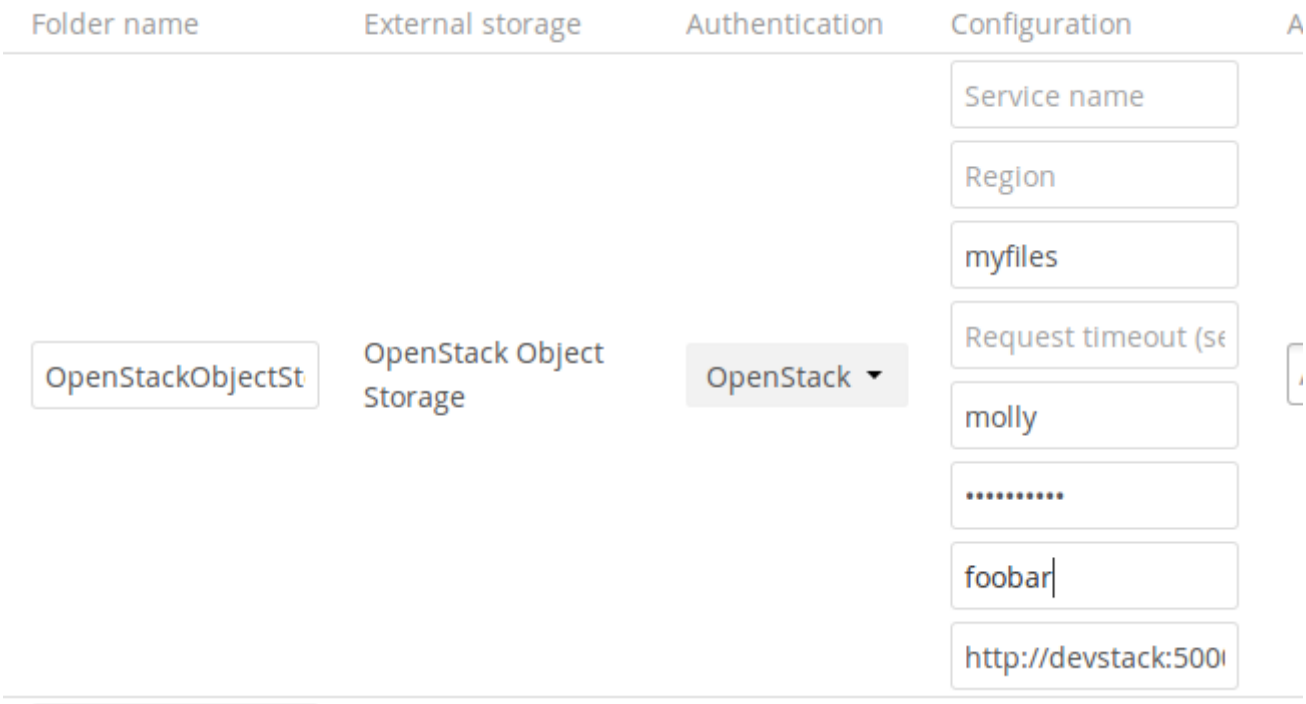
Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

Stockage OpenStack Object

OpenStack Object Storage est utilisé pour se connecter à un serveur OpenStack Swift ou à Rackspace. Deux mécanismes d'authentification sont disponibles : l'un est le mécanisme OpenStack générique et le second est uniquement utilisé pour Rackspace, un fournisseur de stockage objet qui utilise le protocole Swift d'OpenStack.

Le mécanisme d'authentification d'OpenStack utilise le protocole OpenStack Keystone v2. La configuration nécessite les informations suivantes :

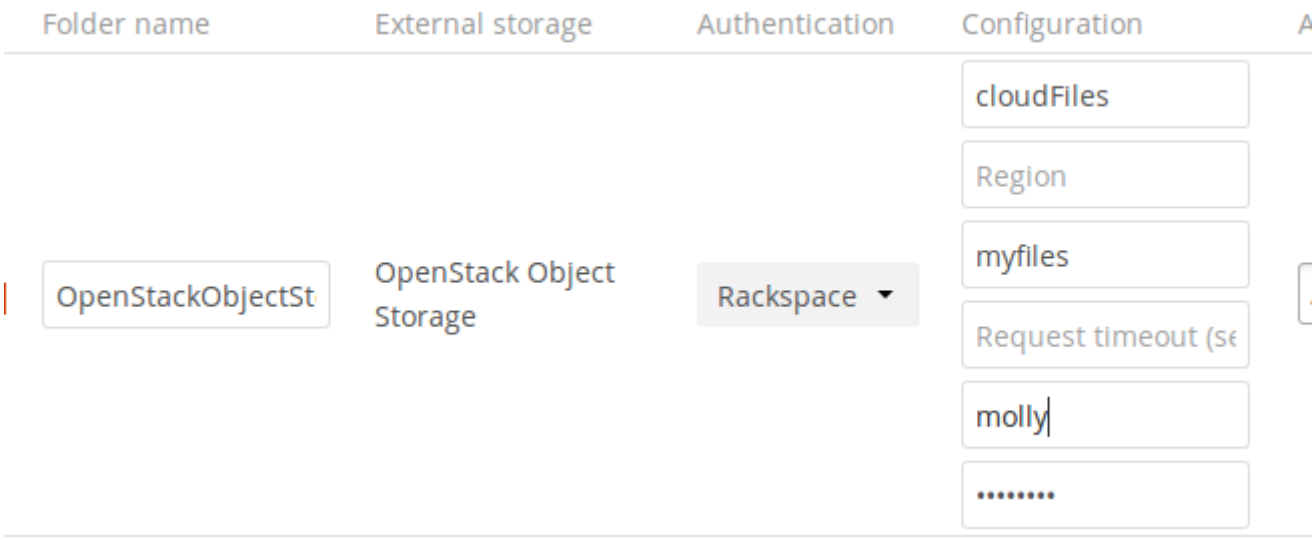
- **Compartiment** : il est propre à chaque utilisateur ; c'est un peu comme un sous-répertoire du stockage total. Le compartiment (bucket) sera créé s'il n'existe pas ;
- **Nom d'utilisateur** du compte ;
- **Mot de passe** du compte ;
- **Nom du tenant** du compte. (Un tenant est similaire à un groupe d'utilisateurs) ;
- **Identity Endpoint URL** : l'URL pour se connecter au compte OpenStack.



Le mécanisme d'authentification Rackspace nécessite :

- un **compartiment** ;
- un **nom d'utilisateur** ;
- une **clé d'API**.

Vous devez aussi saisir le terme **cloudFiles** dans le champ **Service Name**.



Il peut être nécessaire de spécifier une **Région**. Votre région doit être indiquée dans votre information de compte. Vous pouvez en apprendre plus sur les régions Rackspace sur About Regions.

La temporisation des requêtes HTTP est définie dans le champ **Request timeout** en secondes.

Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

ownCloud

Un stockage ownCloud est un stockage *WebDAV* spécialisé, avec des optimisations pour la communication de serveur ownCloud à serveur ownCloud. Consulter la documentation *WebDAV* pour en apprendre plus sur la manière de configurer ownCloud comme stockage externe.

Pour le champ **URL**, utilisez the chemin de la racine de votre installation ownCloud plutôt que le chemin WebDAV. Si votre serveur se trouve sur `https://exemple.com/owncloud`, utilisez `https://exemple.com/owncloud` et non `https://exemple.com/owncloud/remote.php/dav`.

Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

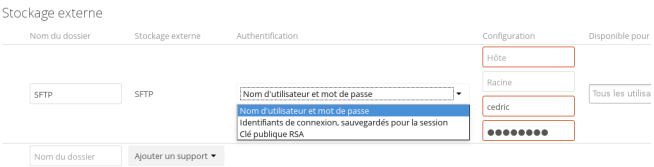
Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

SFTP

Le service SFTP d'ownCloud (FTP dans un tunnel SSH) gère l'authentification par mot de passe et par clé publique

Le champ **Hôte** est requis. Un port peut être spécifié dans le champ **Hôte** dans le format suivant : `nomhote.domaine:port`. Le port par défaut est le port 22 (SSH).

Pour l'authentification par clé publique, vous pouvez générer une paire de clés publique/privée pour votre configuration **Clé publique RSA**.



Après avoir généré vos clés, vous devez copier votre nouvelle clé publique sur le serveur de destination dans `.ssh/authorized_keys`. ownCloud utilisera alors sa clé privée pour s'authentifier sur le serveur SFTP.

Le **Sous-dossier distant** par défaut est le répertoire racine (`/`) du serveur SFTP distant. Vous pouvez saisir un autre répertoire si vous le voulez.

Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

SMB/CIFS

ownCloud peut se connecter aux serveurs de fichiers Windows ou tout autre serveur compatible SMB avec le service SMB/CIFS.

> ***Note***
>
> Le service SMB/CIFS nécessite que le module PHP smbclient soit installé sur le serveur ownCloud. Ce module est inclus dans la plupart des sistributions Linux. Dans Debian, Ubuntu, CentOS et Fedora, c'est `php-smbclient`. Consulter eduardok/libsmbclient-php si votre distribution ne le contient pas. Ce lien indique les archives des sources d'installation et la manière d'installer les paquets binaires.

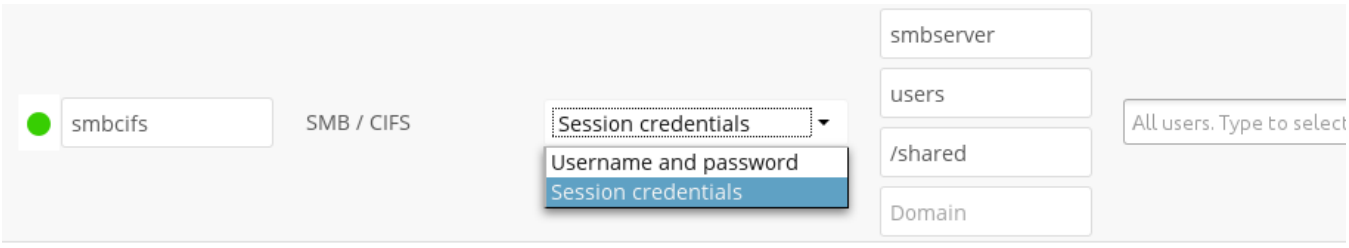
Vous avez aussi besoin d'avoir le client Samba installé sur votre système Linux. Celui est inclus dans toutes les distributions Linux. Dans Debian, Ubuntu et leurs dérivées, c'est `smbclient`. Dans SUSE, Red Hat, CentOS et leurs dérivées, c'est `samba-client`. Vous aurez aussi besoin de `which` et `stdbuf`, qui sont inclus dans la plupart des distributions Linux.

Vous avez besoin des informations suivantes :

- le nom du dossier pour le point de montage local ;
- l'hôte : l'URL du serveur Samba ;
- le nom d'utilisateur : le nom d'utilisateur ou le nom d'utilisateur et le pour se connecter au serveur Samba ;
- le mot de passe : le mot de passe pour se connecter au serveur Samba ;
- le partage : le partage à monter du serveur Samba ;
- le sous-dossier distant : le sous-dossier distant à monter du partage Samba (facultatif, par défaut « / »). Pour assigner le nom de connexion ownCloud automatiquement pour le sous-dossier, utilisez `$user` plutôt qu'un nom de dossier particulier ;
- et enfin les utilisateurs et groupes ownCloud qui pourront accéder au partage.

De manière facultative, vous pouvez indiquer un `Domaine`. Ceci est utile pour les cas ou le serveur SMB a besoin du domaine et du nom utilisateur, et pour les mécanismes d'authentification avancée quand les identifiants de session sont utilisés de sorte que le nom d'utilisateur ne puisse être changé. Il est concaténé au nom d'utilisateur de sorte que le service reçoive `domaine\nom_utilisateur`

> ***Note***
>
> Pour une fiabilité et des performances accrues, nous recommandons d'installer `libsmbclient-php`, un module PHP natif pour se connecter aux serveurs SMB. Il est disponibles sous le nom `php5-libsmbclient` dans les dépôts Open Build Service d'ownCloud.



Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

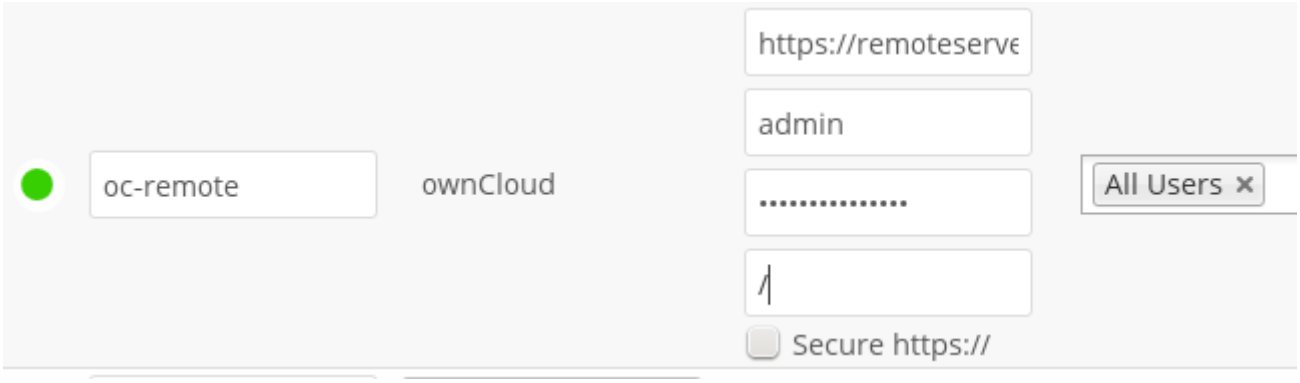
WebDAV

Utiliser ce service pour monter un répertoire de n'importe quel serveur WebDAV ou d'un autre serveur ownCloud.

Vous avez besoin des informations suivantes :

- le « Nom du dossier » : le nom de votre point de montage local ;
- l'URL du serveur WebDAV ou ownCloud ;
- le nom d'utilisateur et le mot de passe pour le serveur distant ;
- « Sécurisation https:// » : nous recommandons toujours https:// pour la sécurité, mais vous pouvez ne pas cocher cette case pour utiliser http://.

Facultatif : un `sous-dossier distant` peut être spécifié pour changer le répertoire de destination. Par défaut, le dossier racine est utilisé.



Note

Les utilisateurs de CPanel doivent installer Web Disk pour activer la fonctionnalité WebDAV.

Consulter *Configuration du stockage externe (interface graphique)* pour des options de montage et des informations supplémentaires.

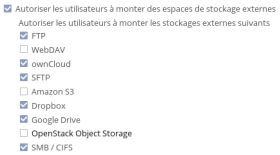
Consulter *Mécanismes d'authentification pour le stockage externe* pour plus d'informations sur les mécanismes d'authentification.

Note

Une configuration SELinux non bloquante est nécessaire pour que ces services fonctionnent. Veuillez vous référer à *Configuration SELinux*.

Permission de montage des espaces de stockage externes

Cochez la case **Autoriser les utilisateurs à monter des espaces de stockage externe** pour autoriser les utilisateurs à monter leurs propres espaces de stockage externe et vérifier les services que vous voulez autoriser. Attention, cela permet potentiellement à un utilisateur de faire des connexions vers d'autres services sur votre réseau !



Détection de fichiers ajoutés aux espaces de stockage externes

Nous recommandons de configurer la tâche de fond **Webcron** ou **Cron** (voir *Tâches d'arrière-plan*) pour permettre à ownCloud de détecter automatiquement les fichiers ajoutés aux espaces de stockages externes.

> **Note**
>
> Vous ne pouvez pas partager de fichiers sur des points de montage externes quand vous utilisez le mécanisme d'authentification **Identifiants de connexion, sauvegardés pour la session**. Cependant, il existe un moyen de contournement, et ceci en utilisant le mode cron Ajax. Voir *Mécanismes à base de mot de passe* pour plus d'informations.

ownCloud ne sera peut-être pas toujours en mesure de détecter ce qui a changé sur le support distant (les fichiers modifiés sans être passés par ownCloud), particulièrement si le fichier se trouve très profondément dans la hiérarchie de dossiers sur le support distant.

Vous pourriez avoir besoin de définir une tâche cron exécutant `sudo -u www-data php occ files:scan --all` (ou remplacez « --all » par le nom de l'utilisateur, voir aussi *Utilisation de la commande occ*) pour déclencher un balayage périodique des fichiers de l'utilisateur (par exemple, toutes les 15 minutes), qui inclut l'espace de stockage distant.

Configuration du stockage externe (fichier de configuration)

À compter de la version 9.0 d'ownCloud, le fichier data/mount.json pour configurer le stockage externe a été retiré et remplacé par un jeu de *commandes occ*.

Mécanismes d'authentification pour le stockage externe

Le système de stockage d'ownCloud accepte un ou plusieurs mécanismes d'authentification tels que les mots de passe, OAuth, basé sur des jetons ou des noms par exemple. Chaque mécanisme d'authentification peut être mis en œuvre par de multiple mécanismes d'authentification. Les différents mécanismes nécessitent des paramètres de configuration différents en fonction de leur comportement.

Mécanismes spéciaux

Le mécanisme d'authentification **Aucun** ne nécessite aucun paramètre de configuration et est utilisé quand le système ne nécessite pas d'authentification.

Le mécanisme d'authentification **Intégré** ne nécessite aucun paramètre de configuration, mais est utilisé pour les anciens stockages qui n'ont pas été migrés vers le nouveau système et qui n'utilise pas les mécanismes d'authentification génériques. Les paramètres d'authentification sont fournis directement par le système.

Mécanismes à base de mot de passe

Le mécanisme **Nom d'utilisateur et mot de passe** nécessite un nom d'utilisateur et un mot de passe définis manuellement. Ceux-ci sont passés directement au système.

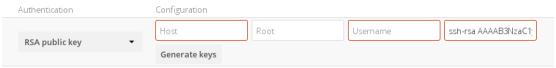
Le mécanisme **Identifiants de connexion, sauvegardés pour la session** utilise les identifiants de connexion d'ownCloud de l'utilisateur pour se connecter au stockage. Ils ne sont stockés nulle part sur le serveur mais dans la session utilisateur augementant ainsi la sécurité. Le revers est que le partage est désactivé quand ce mécanisme est en cours d'utilisation car ownCloud n'a pas d'accès aux identifiants de stockage, et le balayage de fichiers en arrière-plan ne fonctionne pas.

Note

Il existe un moyen de contournement permettant de partager en utilisant **Identifiants de connexion, sauvegardés pour la session**, et ceci en utilisant le mode cron Ajax (voir *Tâches d'arrière-plan*). Le mode cron Ajax nécessite que l'utilisateur utilise activement l'interface Web graphique d'ownCloud.

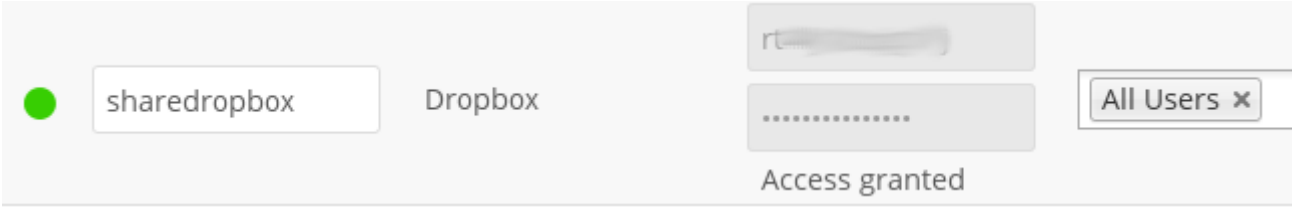
Mécanismes de clés publiques

Actuellement seul le mécanisme RSA est mis en œuvre, où une paire de clés publique privée est généré par ownCloud et la partie publique affichée dans l'interface graphique. Les clés sont générées au format SSH, et ont actuellement 1024 bits de longueur. Les clés peuvent être générées apr un bouton dans l'interface graphique.



OAuth

OAuth 1.0 et OAuth 2.0 sont tous deux mis en œuvre, mais actuellement limités aux interfaces de Dropbox et de Google Drive respectivement. Ces mécanismes nécessitent une configuration supplémentaire du fournisseur de service, où un identifiant et un mot de passe d'application sont fournis et saisis dans ownCloud. ownCloud peut alors réaliser une requête d'authentification pour établir la connexion au stockage.



Configuration du chiffrement

Le but principal du chiffrement du serveur ownCloud est de protéger les fichiers sur les services de stockage distants connectés à votre serveur ownCloud, tels que Dropbox et Google Drive. C'est un moyen simple et facile pour protéger vos fichiers sur un stockage distant.

Dans ownCloud 9.0 le chiffrement fait une distinction entre le chiffrement local et celui des stockages distants. Ceci permet de chiffrer vos stockages distants tels que Dropbox et Google sans avoir à chiffrer également vos fichiers locaux sur le serveur ownCloud.

Note

À compter de la version 9.0 d'ownCloud, le chiffrement authentifié pour tous les nouveaux fichiers chiffrés est géré. consulter https://hackerone.com/reports/108082 pour plus d'informations techniques sur l'impact.

Pour une sécurité maximale, assurez-vous de configurer les stockages externes avec l'option « Rechercher les modifications : Jamais ». Ceci permettra au serveur ownCloud d'ignorer les nouveaux fichiers qui ne sont pas ajoutés via ownCloud, de sorte qu'un administrateur d'un service de stockage distant malveillant ne pourra pas ajouter de fichiers sur votre stockage sans que vous le sachiez. Bien sûr, ce n'est pas approprié si votre stockage distant nécessite des changements externes à ownCloud qui sont légitimes.

ownCloud chiffre les fichiers stockés sur le serveur ownCloud et ceux des stockages distants connectés à votre serveur ownCloud. Le chiffrement et le déchiffrement sont réalisés par le serveur ownCloud. Tous les fichiers envoyés sur le stockage distant seront chiffrés par le serveur ownCloud, et déchiffrés lorsqu'ils sont récupérés avant de vous être présentés ou présentés aux personnes avec qui vous les partagez.

Note

Le chiffrement des fichiers augmente la taille d'environ 35% : vous devez donc prendre en compte cette information quand vous estimez la taille du stockage de votre serveur et quand vous définissez les quotas de stockage utilisateur. Les quotas des utilisateurs sont basés sur la taille des fichiers non chiffrés et non sur la taille des fichiers chiffrés.

Quand les fichiers des stockages externes sont chiffrés par ownCloud, vous ne pouvez pas les partagez directement à partir de votre service de stockage externe mais seulement par l'intermédiaire du serveur ownCloud car la clé de chiffrement des données ne quitte jamais le serveur ownCloud.

Le chiffrement d'ownCloud utilise une clé de chiffrement forte qui est débloquée par le mot de passe des utilisateurs. Vos utilisateurs n'ont pas besoin de retenir un mot de passe supplémentaire, mais juste de se connecter comme d'habitude. Le chiffrement ne concerne que le contenu des fichiers et pas le nom des fichiers ou des dossiers.

Important

Pensez à sauvegarder régulièrement toutes les clés de chiffrement pour empêcher toute perte de données irréversible.

Les clés de chiffrement sont stockées dans les répertoires suivants : ========== ========== Répertoires Description ========== ==========

`data/<utilisateur>/files_encryption` Clés privées des utilisateurs et toutes les autres clés nécessaires au déchiffrement des fichiers utilisateur.

`data/files_encryption` **Clés privées et toutes les autres clés nécessaires au** déchiffrement des fichiers stockés sur un stockage externe.

Verrouillage de fichier transactionnel

Le mécanisme de verrouillage de fichier transactionnel d'ownCloud verrouille les fichiers pour éviter la corruption de fichiers pendant les opérations normales. Il réalise les actions suivantes :

- opération à un niveau supérieur à celui du système de fichiers, de sorte que vous n'avez pas à utiliser un système de fichiers gérant le verrouillage ;
- verrouillage des répertoires parents de sorte qu'ils ne puissent être renommés pendant toute activité survenant sur des fichiers dans les répertoires ;
- libération des verrous après interruption des transactions de fichiers, par exemple, quand un client de synchronisation perd la connexion pendant un téléversement ;
- gestion correcte du verrouillage et de la libération de verrou des fichiers sur les fichiers partagés pendant les modifications effectuées par plusieurs utilisateurs ;
- gestion correcte des verrous sur les montages de stockages externes.

Ce pour quoi n'est pas fait le verrouillage de fichier transactionnel : * empêcher les collisions lors de l'édition collaborative de documents (consulter *Configuration de l'application collaborative Documents* pour en apprendre plus sur la collaboration avec l'applications Documents) ; * empêcher les utilisateurs d'éditer le même document ou prévenir que d'autres utilisateurs travaillent sur le même document.

Plusieurs utilisateurs peuvent ouvrir et éditer un fichier en même temps sans que le verrouillage de fichier transactionnel puisse l'empêcher. Par contre, il empêche l'enregistrement simultané par plusieurs utiliateurs d'un fichier.

Note

Le verrouillage de fichier transactionnel fait partie du code de base d'ownCloud et remplace l'ancienne application File Locking. Cette application a été retirée d'ownCloud dans la version 8.2.1. Si votre serveur ownCloud a toujours cette application, vous devez vous rendre sur votre page Applications pour vérifier qu'elle est bien désactivée ; l'application File Locking et l'application de verrouillage de fichier transactionnel ne peuvent opérer en même temps.

Le verrouillage de fichiers est activé par défaut et utilise le mécanisme de verrouillage de la base de données. Ceci provoque une charge significative sur votre base de données. Utiliser `memcache.locking` retire cette charge de la base de données et améliore les performances. Les administrateurs de serveurs ownCloud fortement sollicités doivent utiliser un mécanisme de memcache (consulter *Configuration de la mémoire cache*.)

Pour utiliser un memcache avec le verrouillage de fichier transactionnel, vous devez installer le serveur Redis et le module PHP correspondant. Après avoir installé Redis, vous devez modifier votre fichier `config.php` comme dans cet exemple:

```
'filelocking.enabled' => true,
'memcache.locking' => '\OC\Memcache\Redis',
'redis' => array(
     'host' => 'localhost',
     'port' => 6379,
     'timeout' => 0.0,
     'password' => '', // Optional, if not defined no password will be used.
      ),
```

Note

Pour une sécurité renforcée, il est recommandé de configurer Redis avec un mot de passe. Consulter http://redis.io/topics/security pour plus d'informations.

Si vous voulez configurer Redis pour qu'il écoute un socket Unix (ce qui est recommandé si Redis est installé sur le même serveur qu'ownCloud), utilisez cet exemple de configuration de `config.php`:

```
'filelocking.enabled' => true,
'memcache.locking' => '\OC\Memcache\Redis',
'redis' => array(
     'host' => '/var/run/redis/redis.sock',
     'port' => 0,
     'timeout' => 0.0,
      ),
```

Consultez le fichier `config.sample.php` pour voir les exemples de configuration et pour tous les mécanismes de memcache gérés.

Si vous utilisez Ubuntu, vous pouvez suivre ce guide pour une installation complète à partir de zéro.

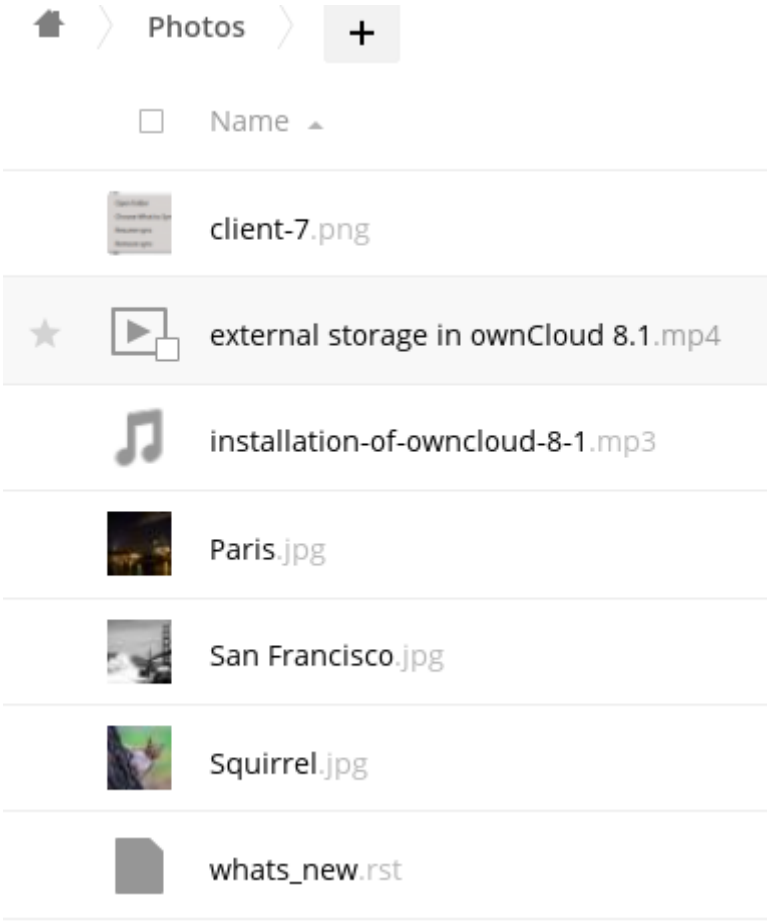
Vous pouvez en apprendre plus sur Redis sur le site Redis. Memcached, le système de cache mémoire distribué populaire, ne convient pas pour le nouveau verrouillage de fichiers car il n'est pas conçu pour stocker les verrous et les données peuvent disparaître du cache à tout moment. Redis est un magasin de clé-valeur, et garantit que les objets en cache sont disponibles aussi longtemps que nécessaire.

Pour les utilisateurs de Debian Jessie, veuillez consulter cette discussion Github si vous rencontrez des problèmes avec l'authentification LDAP.

Configuration des aperçus

Le système de vignettes d'ownCloud génère des aperçus des fichiers pour toutes les applications d'ownCloud qui affichent des fichiers, telles que Fichiers et Galerie.

L'image ci-dessous présente quelques exemples d'aperçus de fichiers de divers types.



Par défaut, ownCloud peut générer des aperçus pour les types de fichiers suivants :

- fichiers images ;
- fichiers MP3 ;
- documents texte.

Note

Les anciennes versions d'ownCloud géraient également la génération d'aperçus d'autres types de fichiers tels que PDF, SVG ou divers documents Office. Pour des raisons de sécurité, ces services ont été désactivés par défaut et ne sont plus considérés comme supportés. Bien que ces services soient toujours disponibles, nous vous déconseillons de les activer et ils ne sont pas documentés.

Paramètres

Veuillez noter que le système d'aperçus d'ownCloud fournit des réglages par défaut adaptés et qu'il n'est par conséquent généralement pas utile d'ajuster ces valeurs de configuration.

Désactivation des aperçus :

Dans certaines circonstances, par exemple si le serveur a des ressources limitées, vous pourriez envisager de désactiver la génération des aperçus. Si vous faites cela, les aperçus seront désactivés dans toutes les applications, y compris l'application Galerie, qui affichera alors des icônes génériques au lieu de vignettes.

Pour cela, définissez l'option `enable_previews` dans le fichier config.php à `false` :

```
<?php
  'enable_previews' => false,
```

Taille maximale des aperçus :

Il existe deux options de configuration pour définir la taille maximale des aperçus.

```
<?php
  'preview_max_x' => null,
  'preview_max_y' => null,
```

Par défauts ces deux options sont définies à « null ». « null » est l'équivalent de sans limite. Les valeurs numériques représentent la taille en pixels. Le code suivant limite la taille des aperçus à 100×100px :

```
<?php
  'preview_max_x' => 100,
  'preview_max_y' => 100,
```

« preview_max_x » représente l'axe des abscisses et « preview_max_y » celui des ordonnées.

Facteur de mise à l'échelle maximal :

Si beaucoup de petites images sont stockées dans l'instance ownCloud et que le systèmes génère des aperçus flous, vous pourriez envisager de définir un facteur de mise à l'échelle maximal. Par défaut, un facteur 10 est appliqué sur les images :

```
<?php
  'preview_max_scale_factor' => 10,
```

Si vous voulez désactiver la mise à l'échelle, vous pouvez définir cette valeur à « 1 » :

```
<?php
  'preview_max_scale_factor' => 1,
```

Si vous voulez désactiver le facteur de mise à l'échelle, vous pouvez définir cette valeur à « null » :

```
<?php
  'preview_max_scale_factor' => null,
```

Contrôle de version et rétention de fichiers

Le système de gestion de version supprime les anciennes versions de fichiers pour s'assurer que l'utilisateur ne soit pas à court d'espace disque. Le processus suivant est utilisé par défaut pour supprimer les anciennes versions :

- la première seconde, ownCloud conserve une version ;
- les 10 premières secondes, ownCloud conserve une version toutes les 2 secondes ;
- la première minute, ownCloud conserve une version toutes les 10 secondes ;
- la première heure, ownCloud conserve une version par minute ;
- les premières 24 heures, ownCloud conserve une version par heure ;
- les 30 premiers jours, ownCloud conserve une version par jour ;
- après les 30 premiers jours, ownCloud conserve une version par semaine.

Les versions sont ajustées selon ce processus chaque fois qu'une nouvelle version est créée.

L'application de gestion de version n'utilise jamais plus de 50% de l'espace disque libre de l'utilisateur. Si les versions stockées excèdent cette limite, ownCloud supprime les plus anciennes versions jusqu'à tomber à nouveau sous la limite des 50%.

Vous pouvez modifier ce comportement dans le fichier `config.php`. Le paramètre par défaut est `auto`, ce qui définit le comportement par défaut:

```
'versions_retention_obligation' => 'auto',
```

Options supplémentaires :

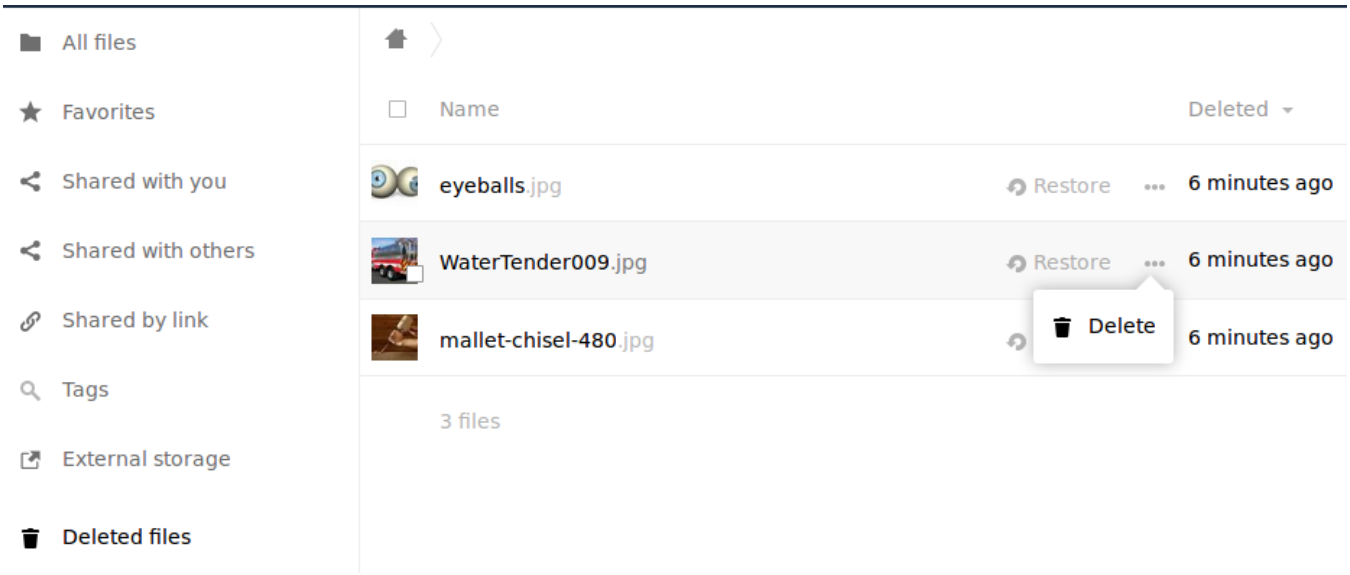
- `D, auto`
  Conserve les versions au moins « D » jours et applique les règles d'expiration à toutes les versions plus anciennes que « D » jours
- `auto, D`
  Supprime toutes les versions plus anciennes que « D » jours automatiquement et supprime les autres versions en fonction des règles d'expiration
- `D1, D2`
  Conserve les versions au moins « D1 » jours et suppriment celles excédant « D2 » jours.
- `disabled`
  Désactive l'application Versions ; aucun fichier ne sera supprimé.

Rétention de fichiers - Édition Entreprise

Les clients de l'Édition Entreprise disposent d'outils supplémentaires pour gérer les politiques de rétention de fichiers ; consulter *Advanced File Tagging With the Workflow App (Enterprise only)*.

Gestion de la corbeille

La corbeille d'ownCloud (`files_trashbin`) supprime définitivement les fichiers en fonction des quotas et de l'âge des fichiers des utilisateurs. Quand un utilisateur supprime un fichier, il n'est pas immédiatement effacé du serveur ownCloud mais il est déplacé dans la corbeille. L'utilisateur a alorls le choix de restaurer le fichier ou de le supprimer définitivement.



En tant qu'administrateur d'ownCloud, vous pouvez utiliser les commandes `occ` pour supprimer définitivement les fichiers de la corbeille manuellement, sans attendre la fin du processus normal d'expiration des fichiers:

```
trashbin
 trashbin:cleanup   purge les fichiers supprimés
 trashbin:expire    expiration des corbeilles utilisateurs
```

La commande `trashbin:cleanup` supprime les fichiers supprimés des utilisateurs spécifiés dans une liste séparés par des espaces, ou de tous les utilisateurs si aucun n'est spécifié. Cet exemple purge les fichiers supprimés de tous les utilisateurs:

```
sudo -u www-data php occ trashbin:cleanup
Remove all deleted files
Remove deleted files for users on backend Database
 user1
 user2
```

```
 user3
 user4
```

Cet exemple purge les fichiers supprimés des utilisateurs user2 et user4:

```
sudo -u www-data php occ trashbin:cleanup user2 user4
 Remove deleted files of user2
 Remove deleted files of user4
```

`trashbin:expire` supprime seulement les fichiers expirés en fonction du paramètre `trashbin_retention_obligation` du fichier `config.php`. Le paramètre par défaut est `auto`, et conserve les fichiers dans la corbeille 30 jours, puis supprime les plus anciens si de l'espace est nécessaire pour respecter le quota de stockage des utilisateurs. Les fichiers peuvent ne pas être supprimés si aucun espace n'est nécessaire.

Par défaut, les fichiers ayant expiré sont supprimés pour tous les utilisateurs, ou vous pouvez indiquer une liste d'utilisateurs séparés par des espaces:

```
sudo -u www-data php occ trashbin:cleanup user1 user2
 Remove deleted files of user1
 Remove deleted files of user2
```

Consulter la section **Fichiers supprimés** dans *Paramètres de Config.php*, et la section *Corbeille* dans *Utilisation de la commande occ*.

Configuration de la base de données

Conversion du moteur de base de données

Vous pouvez convertir une base de données SQLite en une base de données plus performante comme MySQL, MariaDB ou PostgreSQL avec l'outil en ligne de commande d'ownCloud. SQLite est utile pour faire des tests ou pour des serveurs ownCloud n'ayant qu'un seul utilisateur, mais ne supporte pas la charge pour des serveurs de production vaec beaucoup d'utilisateurs.

Note

ownCloud Édition Entreprise ne gère pas SQLite.

Lancement de la conversion

Tout d'abord, il faut créer la nouvelle base de données, appelée ici « nouvelle_bdd ». Dans le dossier racine d'ownCloud, lancer la commande

```
php occ db:convert-type [options] type nom_utilisateur hôte base_de_données
```

Les options :

- `--port="3306"` le port de la base de données (facultatif)
- `--password="mysql_user_password"` le mot de passe pour la nouvelle base de données. Si omis, l'outil vous le demandera (facultatif)
- `--clear-schema` effacement du schéma (facultatif)
- `--all-apps` par défaut, les tables pour les applications activées sont converties, utilisé pour convertir aussi les tables ds applications désactivées (facultatif)

Note : Le convertisseur recherche les applications dans les dossiers des applications configurées et utilise les définitions de schéma dans les applications pour créer les nouvelle table. Donc, les tables des applications supprimées ne seront pas converties, même avec l'option `--all-apps`

Par exemple

```
php occ db:convert-type --all-apps mysql oc_mysql_user 127.0.0.1 nouvelle_bdd
```

Pour réussir la conversion, vous devez saisir `yes` à l'invite de la question `Continue with the conversion?`

En cas de réussite, le convertisseur configurera automatiquement la nouvelle base de données dans votre fichier de configuration `config.php` d'ownCloud.

Tables non convertibles

Si vous effectuez la mise à jour de votre installation ownCloud, il pourrait exister d'anciennes tables qui ne sont plus utilisées. Le convertisseur vous indiquera lesquelles.

```
Les tables suivantes ne seront pas converties :
oc_permissions
...
```

Vous pouvez ignorer ces tables. Voici une liste des anciennes tables connues :

- oc_calendar_calendars
- oc_calendar_objects
- oc_calendar_share_calendar
- oc_calendar_share_event
- oc_fscache
- oc_log
- oc_media_albums
- oc_media_artists
- oc_media_sessions
- oc_media_songs
- oc_media_users
- oc_permissions
- oc_queuedtasks
- oc_sharing

Configuration de la base de données

ownCloud nécessite une base de données pour stocker les données administratives. Les moteurs de bases de données suivants sont actuellement gérés :

- MySQL / MariaDB
- PostgreSQL
- Oracle (ownCloud Édition Entreprise seulement)

MySQL ou MariaDB sont les moteurs de bases de données recommandés.

Prérequis

Choisir d'utiliser MySQL / MariaDB, PostgreSQL ou Oracle (ownCloud Édition Entreprise seulement) nécessite d'installer et de configurer d'abord la base de données. (Utilisateurs Oracle, consulter *Configuration de la base de données Oracle*.)

> ### *Note*
>
> Les étapes nécessaires à la configuration d'une base de données tierce ne sont pas abordées dans ce document. Veuillez vous référer à la documentation de votre base de données pour les instructions.

MySQL / MariaDB storage engine

Depuis ownCloud 7, seul InnoDB est supporté comme moteur de stockage. Il existe des hébergements partagés qui ne gèrent pas InnoDB mais seulement MyISAM. Exécuter ownCloud sur de tels environnements n'est pas supporté.

MySQL / MariaDB avec activation des journaux binaires

ownCloud utilise actuellement une transaction d'isolation `TRANSACTION_READ_COMMITTED` pour éviter des pertes de données pour des scénarios de forte charge (par exemple, en utilisant le client de synchronisation avec beaucoup de clients/utilisateurs et beaucoup d'opération en parallèle). Ceci nécessite que les journaux binaires soient correctement configurés ou désactivés en utilisant MySQL ou MariaDB. Votre système est affecté si vous observez ce qui suit dans le fichier journal pendant l'installation ou la mise à jour d'ownCloud :

> An unhandled exception has been thrown: exception 'PDOException' with message 'SQLSTATE[HY000]: General error: 1665 Cannot execute statement: impossible to write to binary log since BINLOG_FORMAT = STATEMENT and at least one table uses a storage engine limited to row-based logging. InnoDB is limited to row-logging when transaction isolation level is READ COMMITTED or READ UNCOMMITTED.'

Il existe deux solutions. La première est de désactiver les journaux binaires. Les journaux binaires enregistrent toutes les modifications dans votre base de données et le temps que prend chacune d'elle. Le but des journaux binaires est de permettre la réplication et de gérer les opérations de sauvegarde.

La seconde est de modifier la déclaration « BINLOG_FORMAT » dans le fichier de configuration ou dans le script de démarrage de votre base de données avec « BINLOG_FORMAT = MIXED ». Consulter Aperçu des journaux binaires et Les journaux binaires pour des informations détaillées.

Moteur de stockage MySQL / MariaDB niveau d'isolation de transaction « READ COMMITED »

Comme vu plus haut, ownCloud utilise le niveau d'isolation de transaction `TRANSACTION_READ_COMMITTED`. Certaines configurations de base de données forcent l'utilisation de niveaux d'isolation de transaction différents. Pour éviter les pertes de données dans les cas de forte charge (par ex. en utilisant un client de synchronisation avec beaucoup d'utilisateurs et d'opérations en parallèle), vous devez configurer le niveau d'isolation de transaction de manière appropriée. Veuillez consulter MySQL manual pour des informations détaillées.

Paramètres

Pour paramétrer la base de données pour ownCloud, veuillez suivre les instructions dans *Assistant d'installation*. Vous ne devriez pas avoir besoin de modifier les valeurs correspondantes dans le fichier config/config.php. Cependant, dans certains cas (par exemple si vous voulez connecter votre instance ownCloud à une base de données créée lors d'une installation précédente d'ownCloud), certaines modifications pourraient être nécessaires.

Configuration d'une base de données MySQL ou MariaDB

Si vous décidez d'utiliser une base de données MySQL ou MariaDB, assurez-vous :

- d'avoir installer et activer l'extension PHP pdo_mysql ;
- que **mysql.default_socket** pointe vers le bon socket (si la base de données est exécutée sur le même serveur qu'ownCloud).

Note

MariaDB est rétro-compatible avec MySQL. Toutes les instructions sont valables pour les deux. Vous n'avez pas besoin de remplacer mysql par quoi que ce soit.

La configuration de PHP dans le fichier /etc/php5/conf.d/mysql.ini doit ressembler à ceci :

```
# configuration for PHP MySQL module
extension=pdo_mysql.so

[mysql]
mysql.allow_local_infile=On
```

```
mysql.allow_persistent=On
mysql.cache_size=2000
mysql.max_persistent=-1
mysql.max_links=-1
mysql.default_port=
mysql.default_socket=/var/lib/mysql/mysql.sock  # Debian squeeze: /var/run/mysqld/mysqld.soc
mysql.default_host=
mysql.default_user=
mysql.default_password=
mysql.connect_timeout=60
mysql.trace_mode=Off
```

Vous devez maintenant créer l'utilisateur de la base de données et la base de données elle-même en utilisant l'interface de ligne de commande de MySQL. Les tables de la base de données seront créées lors de votre première connexion à ownCloud.

Pour démarrer l'interface en ligne de commande de MySQL, utilisez:

```
mysql -uroot -p
```

Il apparaît alors une invite **mysql>** ou **MariaDB [root]>**. Saisissez maintenant les lignes suivantes et confirmez en appuyant sur la touche « Entrée » :

```
CREATE DATABASE IF NOT EXISTS owncloud;
GRANT ALL PRIVILEGES ON owncloud.* TO 'utilisateur'@'localhost' IDENTIFIED BY 'mot_de_passe'
```

Vous pouvez maintenant quitter l'interface en saisissant:

```
quit
```

Une instance d'ownCloud configurée avec MySQL doit contenir le nom d'hôte sur lequel s'exécute la base de données, un nom d'utilisateur valide et un mot de passe pour y accéder ainsi que le nom de la base de données. Le fichier config/config.php créé par *Assistant d'installation* devrait par conséquent contenir des entrées similaires à ce qui suit :

```
<?php

  "dbtype"        => "mysql",
  "dbname"        => "owncloud",
  "dbuser"        => "utilisateur",
  "dbpassword"    => "mot_de_passe",
  "dbhost"        => "localhost",
  "dbtableprefix" => "oc_",
```

Base de données PostgreSQL

Si vous décidez d'installer une base de données PostgreSQL, assurez-vous d'avoir installer et activer l'extension PostgreSQL dans PHP. La configuration PHP dans le fichier /etc/php5/conf.d/pgsql.ini devrait ressembler à ceci :

```
# configuration for PHP PostgreSQL module
extension=pdo_pgsql.so
extension=pgsql.so

[PostgresSQL]
pgsql.allow_persistent = On
pgsql.auto_reset_persistent = Off
pgsql.max_persistent = -1
pgsql.max_links = -1
pgsql.ignore_notice = 0
pgsql.log_notice = 0
```

La configuration par défaut pour PostgreSQL (au moins sur Ubuntu 14.04) est d'utiliser la méthode d'authentification par pair. Vérifier le fichier /etc/postgresql/9.3/main/pg_hba.conf pour voir quelle méthode d'authentification est utilisée pour votre instance. Pour démarrer l'interface de ligne de commande de PostgreSQL, saisissez ce qui suit:

```
sudo -u postgres psql -d template1
```

Une invite **template1=#** apparaît alors. Saisissez maintenant les lignes suivantes et confirmez en appuyant sur la touche entrée :

```
CREATE USER utilisateur CREATEDB;
CREATE DATABASE owncloud OWNER utilisateur;
```

Vous pouvez quitter l'interface en saisissant:

```
\q
```

Une instance ownCloud configurée avec PostgreSQL doit contenir le chemin d'accès au socket sur lequel la base de données est exécutée, le nom d'hôte, l'utilisateur système utilisé par le processus PHP et un mot de passe vide pour y accéder et le nom de la base de données. Le fichier config/config.php créé par *Assistant d'installation* doit par conséquent contenir des entrées similaires à ce qui suit :

```
<?php

  "dbtype"        => "pgsql",
  "dbname"        => "owncloud",
  "dbuser"        => "utilisateur",
  "dbpassword"    => "",
  "dbhost"        => "/var/run/postgresql",
  "dbtableprefix" => "oc_",
```

Note

L'hôte pointe en fait vers le socket qui est utilisé pour se connecter à la base de données. Utiliser « localhost » ici ne fonctionnera pas si postgreSQL est configuré pour utiliser l'authentification par pair. Veuillez noter également qu'aucun mot de passe n'est indiqué, car cette méthode d'authentification n'utilise pas de mot de passe.

Si vous utilisez une autre méthode d'authentification, vous aurez besoin de suivre les étapes suivantes pour configurer la base de données. Vous devez créer un utilisateur de base de données et la base de données elle-même en utilisant l'interface de ligne de commande de PostgreSQL. Les tables de la base de données seront créées lors de votre première connexion à ownCloud.

Pour démarrer l'interface de commande de PostgreSQL, utilisez:

```
psql -hlocalhost -Upostgres
```

Une invite **postgres=#** apparaît alors. Saisissez maintenant les lignes suivantes et confirmez en appuyant sur la touche « Entrée » :

```
CREATE USER utilisateur WITH PASSWORD 'mot_de_passe';
CREATE DATABASE owncloud TEMPLATE template0 ENCODING 'UNICODE';
ALTER DATABASE owncloud OWNER TO utilisateur;
GRANT ALL PRIVILEGES ON DATABASE owncloud TO utilisateur;
```

Vous pouvez quitter l'interface de ligne de commande en saisissant:

```
\q
```

Une instance d'ownCloud configurée avec PostgreSQL doit contenir le nom d'hôte sur lequel la base de données est exécutée, un nom d'utilisateur valide et un mot de passe pour y accéder et le nom de la base de données. Le fichier config/config.php créé par *Assistant d'installation* doit contenir des entrées similaires à ce qui suit :

```
<?php

  "dbtype"        => "pgsql",
  "dbname"        => "owncloud",
  "dbuser"        => "utilisateur",
```

```
  "dbpassword"    => "mot_de_passe",
  "dbhost"        => "localhost",
  "dbtableprefix" => "oc_",
```

Dépannage

Comment passer outre l'erreur générale : « 2006 MySQL server has gone away »

La requête de base de données prend trop de temps et par conséquent, le serveur MySQL dépasse le délai de temporisation (time out). Il est aussi possible que le serveur rejette un paquet qui est trop gros. Veuillez consulter le manuel de votre base de données pour savoir comment augmenter la valeur des deux options de configuration `wait_timeout` et/ou `max_allowed_packet`.

Certains hébergeurs nautorisent pas l'accès à ces options de configuration. Dans ce cas, ownCloud propose une option de configuration `dbdriveroptions` dans votre fichier config/config.php où vous pouvez passer ces options au pilote de base de données. Veuillez vous référer à *Paramètres de Config.php* pour un exemple.

Comment savoir si le serveur MySQL/PostgreSQL est joignable ?

Pour vérifier la connectivité réseau du serveur, utilisez la commande « ping » avec le nom d'hôte du serveur (bdd.serveur.com dans cet exemple):

```
ping bdd.serveur.dom
```

```
PING bdd.serveur.dom (ip-address) 56(84) bytes of data.
64 bytes from votre-serveur.local.lan (192.168.1.10): icmp_req=1 ttl=64 time=3.64 ms
64 bytes from votre-serveur.local.lan (192.168.1.10): icmp_req=2 ttl=64 time=0.055 ms
64 bytes from votre-serveur.local.lan (192.168.1.10): icmp_req=3 ttl=64 time=0.062 ms
```

Pour une vérification plus détaillée, comme l'accès à la base de données elle-même, voir la question suivante.

Comment savoir si l'utilisateur créé peut accéder à la base de données ?

Le moyen le plus simple de tester si une base de données peut être jointe est de démarrer l'interface de ligne de commande :

MySQL :

En supposant que le serveur de base de données est installé sur le système à partir duquel vous exécutez la commande:

```
mysql -uUTILISATEUR -p
```

Pour accéder à une installation MySQL sur une machine différente, ajoutez l'option « -h » suivie du nom d'hôte:

```
mysql -uUTILISATEUR -p -h NOM_HÔTE
```

```
mysql> SHOW VARIABLES LIKE "version";
+---------------+--------+
| Variable_name | Value  |
+---------------+--------+
| version       | 5.1.67 |
+---------------+--------+
1 row in set (0.00 sec)
mysql> quit
```

PostgreSQL :

En supposant que le serveur de base de données est installé sur le système à partir duquel vous exécutez la commande:

```
psql -Uutilisateur -doncloud
```

Pour accéder à une installation PostgreSQL sur une machine différente, ajoutez l'option « -h » suivie du nom d'hôte:

```
psql -Uutilisateur -doncloud -h NOM_HÔTE
```

```
postgres=# SELECT version();
PostgreSQL 8.4.12 on i686-pc-linux-gnu, compiled by GCC gcc (GCC) 4.1.3 20080704 (prerelease
(1 row)
postgres=# \q
```

Commandes SQL utiles

Afficher les utilisateurs de bases de données:

```
MySQL      : SELECT User,Host FROM mysql.user;
PostgreSQL : SELECT * FROM pg_user;
```

Afficher les bases de données disponibles:

```
MySQL      : SHOW DATABASES;
PostgreSQL : \l
```

Afficher les tables d'ownCloud dans la base de données:

```
MySQL      : USE owncloud; SHOW TABLES;
PostgreSQL : \c owncloud; \d
```

Quitter la base de données:

```
MySQL      : quit
PostgreSQL : \q
```

Gestion des types MIME

Alias de type MIME

ownCloud permet de créer des alias pour les types MIME de sorte que vous pouvez afficher des icônes personnalisées pour les fichiers. Par exemple, vous pourriez vouloir une jolie icône pour les fichiers audio à la place de celle fournie par défaut.

Par défaut, ownCloud fournit `owncloud/resources/config/mimetypealiases.dist.json`. Ne modifiez pas ce fichier car il sera remplacé lors des mises à jour d'ownCloud. Créez plutôt votre propre fichier `owncloud/config/mimetypealiases.json` avec vos alias personnalisés. Il faut utiliser la même syntawe que celle du fichier `owncloud/resources/config/mimetypealiases.dist.json`.

Quand vos changements sont termins dans votre fichier `mimetypealiases.json`, utilisez la commande `occ` pour propager les changements dans le système. L'exemple suivant s'applique à Ubuntu Linux:

```
$ sudo -u www-data php occ maintenance:mimetype:update-js
```

Consulter *Utilisation de la commande occ* pour en apprendre plus sur la commande `occ`.

Voici quelques types MIME communs que vous pourriez vouloir modifier :

image
: Image générique

image/vector
: Image vectorielle

audio
: Fichier audio générique

x-office/document
: Document de traitement de texte

x-office/spreadsheet
: Document de tableur

x-office/presentation
: Document de présentation

text
: Document texte générique

text/code
: Source code

Correspondance de type MIME

ownCloud permet aux administrateurs de spécifier une correspondance entre l'extension d'un fichier et un type MIME. Par exemple, les fichiers se terminant par `mp3` correspondent au type `audio/mpeg`. ownCloud affichera alors l'icône audio.

Par défaut, ownCloud fournit `mimetypemapping.dist.json`. C'est un simple tableau json. Les administrateurs ne doivent pas modifier ce fichier car il sera remplacé lors des mises à jour d'ownCloud. Il faut créer un fichier nommé `mimetypemapping.json` et faire les modifications dans celui-ci. Ce fichier a la priorité sur celui fourni avec ownCloud.

Récupération de l'icône

Quand une icône est récupérée pour un type MIME, si le type MIME complet n'est pas trouvé, la recherche prendra la partie avant la barre de fraction. Soit un fichier de type MIME 'image/mon-image-perso', si l'icône n'existe pas pour le type MIME complet, l'icône pour 'image' sera alors utilisée. Ceci permet aux types MIME spécialisés de récupérer les icônes génériques quand les icônes appropriées ne sont pas disponibles.

Maintenance

Configuration du mode de maintenance

Vous devez mettre votre serveur ownCloud en mode maintenance avant de réaliser des mises à jour ou pour réaliser des diagnostics de dépannage ou de la maintenance. Veuillez lire *Utilisation de la commande occ* pour savoir comment mettre votre serveur dans les divers modes de maintenance (`maintenance:mode`, `maintenance:singleuser` et `maintenance:repair`) avec la commande `occ`.

`maintenance:mode` verrouille les sessions des utilsiateurs connectés et empêche les nouvelles connexions. C'estle mode utilisé pour les mises à jour. Vous devez lancer la commande `occ` en tant qu'utilisateur HTTP, comme dans cet exemple pour Ubuntu Linux:

```
$ sudo -u www-data php occ maintenance:mode --on
```

Vous pouvez également mettre votre serveur dans ce mode en modifiant le fichier config/config.php. Changez le paramètre `"maintenance" => false` pour `"maintenance" => true`:

```
<?php

 "maintenance" => true,
```

Puis remettez-le à `false` quand vous avez terminé.

Sauvegarder ownCloud

Pour sauvegarder une installation d'ownCloud, il y a quatre choses principales à retenir :

1. le dossier de configuration `config/` ;
2. le dossier de données `data/` ;
3. la base de données d'ownCloud ;
4. vos fichiers de thèmes personnalisés, si vous en avez (voir Thèmes ownCloud).

Quand vous installez votre serveur ownCloud à partir de nos paquets Open Build Service (ou des paquets de votre distribution, ce que nous ne recommandons pas) **ne sauvegardez pas les fichiers de votre serveur ownCloud**, qui sont les autres fichiers dans votre répertoire `owncloud/` comme `core/`, `3rdparty/`, `apps/`, `assets/`, `lib/`, et tout le reste. Si vous restaurez ces fichiers à partir d'une sauvegarde, ils peuvent ne pas être synchrnisés avec les versions du paquet courant et ne passeront pas la vérification d'intégrité du code. Cela peut aussi provoquer d'autres erreurs, comme des pages blanches.

Quand vous installez ownCloud à partir de l'archive, ceci ne sera pas un problème, vous pouvez sauvegarder sans risque toute votre installation d'ownCloud, à l'exception de la base de données. Les bases de données ne peuvent pas être copiées. Vous devez utiliser les outils de base de données pour faire une sauvegarde correcte de la base.

Pour restaurer votre installation ownCloud à partir d'une sauvegarde, veuillez consulter *Restauration d'ownCloud*.

Sauvegarde des répertoires config/ et data/

Copier simplement vos répertoires `config/` et `data/` en dehors de l'arborescence d'ownCloud. Cet exemple utilise `rsync` pour copier les deux répertoires vers `/backupdir`

```
rsync -Aax config data /oc-backupdir/
```

Il existe plusieurs moyens de sauvegarder des fichiers normaux, et vous pouvez utiliser la méthode à laquelle vous êtes habitué.

Sauvegarde de la base de données

Vous ne pouvez pas juste copier une base de données, mais vous devez utiliser les outils appropriés pour faire une sauvegarde correcte.

MySQL/MariaDB

MySQL ou MariaDB sont les moteurs de base de données recommandés. Pour sauvegarder MySQL/MariaDB :

```
mysqldump --single-transaction -h [serveur] -u [utilisateur] -p[mot-de-passe] [nom_base_de_d
```

Exemple

```
mysqldump --single-transaction -h mon_serveur -u utilisateur -pmot-de-passe nom_base_de_donn
```

SQLite

```
sqlite3 data/owncloud.db .dump > owncloud-bdd-svg_`date +"%Y%m%d"`.bak
```

PostgreSQL

```
PGPASSWORD="mot-de-passe" pg_dump [nom_base_de_données] -h [serveur] -U [utilisateur] -f own
```

Restauration de fichiers quand le chiffrement est activé

Si vous devez restaurer des fichiers à partir d'une sauvegarde lorsque le chiffrement était activé, voici comment faire.

Note

Ceci fonctionne seulement à partir de la version 8.2.7 d'ownCloud et suivantes. Veuillez noter que ceci **n'est pas supporté officiellement**. ownCloud supporte officiellement la restauration d'une sauvegarde complète ou pas de restauration du tout mais pas la restauration partielle.

1. restaurez le fichier à partir de la sauvegarde ;
2. restaurez les clés de chiffrement du fichier à partir de la sauvegarde ;

3. exécutez `occ files:scan` ; ceci permet au scanner de trouver le fichier (veuillez noter que dans la base de données il aura la valeur « size » définie avec la taille du fichier chiffré, ce qui est incorrect et de taille supérieure et que le drapeau « encrypted » sera défini à 0) ;
4. modifiez le drapeau « encrypted » à 1 dans la base de données pour tous les *fichiers* dans `files/path`, mais **pas** les répertoires (définir ce drapeau à 1 indique à l'application de chiffrement que le fichier est chiffré et nécessite un traitement) ;

Note

Il n'est pas nécessaire de mettre à jour le drapeau « encrypted » pour des fichiers dans « files_versions » ou « files_trashbin », car ils ne sont pas analysés ou trouvés par la commande `occ files:scan`.

5. téléchargez le fichier une fois en tant qu'utilisateur ; la taille du fichier sera alors corrigée automatiquement.

Ce processus pourrait ne pas être adapté à tous les environnements. Si ce n'est pas appliicable pour vous, vous aurez besoin de lancer la commande OCC qui réalise l'analyse (scan). Mais ceci nécessitera de connaître le mot de passe de l'utilisateur ou de la clé de récupération.

Comment mettre à jour votre serveur ownCloud

Il existe trois façons de mettre à jour votre serveur ownCloud :

- En utilisant votre *gestionnaire de paquets Linux* avec nos dépôts officiels ownCloud. Ceci est la méthode recommandée.
- Avec l'*application Updater* (Édition Serveur seulement). Recommandé pour les hébergeurs et pour les utilisateurs qui veulent un moyen aisé de suivre les différents canaux de mise à jour. (Ceci n'est pas disponible et pas supporté dans l'Édition Entreprise).
- Avec la *mise à jour manuelle* avec l'archive `.tar` d'ownCloud sur owncloud.org/install/.
- La mise à jour manuelle est aussi une option pour les hébergements partagés ; téléchargez et décompressez l'archive ownCloud sur votre PC. Supprimez tous les fichiers ownCloud existants sauf `data/` et `config/`, sur votre compte hébergé. Puis, téléversez les nouveaux fichiers ownCloud sur votre compte hébergé, en préservant les répertoires `data/` et `config/` existants.
- Les utilisateurs de l'Édition Entreprise utiliseront leurs dépôts Entreprise pour maintenir leurs serveurs ownCloud à jour, plutôt que le service Open Build Veuillez consulter *Installation et mise à jour de l'Édition Entreprise* pour plus d'informations.

Warning

Lors de la mise à jour de la version 9.0 vers la version 9.1 avec des agendas et des carnets d'adresses existants, veuillez consulter les *Notes de version ownCloud 9.1* de la version 9.0 pour des informations importantes sur les étapes de migration nécessaires pendant cette mise à jour.

Quand une mise à jour est disponible pour votre serveur ownCloud, vous verrez une notification en haut de votre interface Web ownCloud. Quand vous cliquez sur la notification, cela vous amène ici, sur cette page.

Il est recommandé de mettre à jour régulièrement votre serveur ownCloud, et d'installer toutes les mises à jour mineures et majeures sans en omettre une seule, car omettre une version augmente les risques d'erreurs. Les versions majeures sont 8.0, 8.1, 8.2 et 9.0. Les versions mineures sont par exemple 8.0.9 and 8.1.3. **L'omission de version majeure n'est pas supportée**.

La mise à jour suspend le service. Votre serveur ownCloud sera passé en mode maintenance et vos utilisateurs seront bloqués jusqu'à la fin de la mise à jour. Les grosses installations peuvent mettre plusieurs heures pour achever la mise à jour.

Warning

Le retour arrière n'est pas supporté et risque de corrompre vos données ! Si vous voulez revenir à une version antérieure, faites une nouvelle installation et restaurez vos données et votre base de données à partir d'une sauvegarde. Avant de faire cela, ouvrez un ticket de support (si vous avez souscrit un support payant) ou demandez de l'aide sur les forums d'ownCloud pour voir si votre problème peut être résolu sans retour arrière.

Prérequis

Vous devez faire des *sauvegardes régulières* et faire une sauvegarde avant toute mise à jour.

Faites alors une revue des applications tierces, si vous en avez, et assurez-vous de leur compatibilité avec la nouvelle versions d'ownCloud. Toute application qui n'est pas développée par ownCloud affiche une description en ce sens. **L'installation d'applications non supportées se fait à vos propres risques**. Ensuite, avant la mise à jour, vous devez désactiver toutes les applications tierces. À la fin de la mise à jour, vous pouvez les réactiver.

Versions précédentes d'ownCloud

Vous trouverez les versions précédentes d'ownCloud dans le fichier Changelog.

Retour arrière

Si vous devez revenir en arrière, consultez *Restauration d'ownCloud*.

Dépannage

Si vous faites la mise à jour d'ownCloud et que vous utilisez MySQL ou MariaDB avec les logs binaires activés, la mise à jour peut échouer avec ces erreurs dans le fichier journal de MySQL/MariaDB:

```
An unhandled exception has been thrown:
exception 'PDOException' with message 'SQLSTATE[HY000]: General error: 1665
Cannot execute statement: impossible to write to binary log since
BINLOG_FORMAT = STATEMENT and at least one table uses a storage engine limited
to row-based logging. InnoDB is limited to row-logging when transaction
isolation level is READ COMMITTED or READ UNCOMMITTED.'
```

Veuillez consulter *MySQL / MariaDB avec activation des journaux binaires* pour savoir comment configurer correctement votre environnement.

Parfois, *des fichiers ne s'affichent pas après une mise à jour*. Un nouveau balayage des fichiers peut aider:

```
sudo -u www-data php console.php files:scan --all
```

Consultez la page de support d'owncloud.org pour trouver plus de ressources pour les particuliers et pour les utilisateurs en entreprise.

Parfois, ownCloud peut rester *bloqué pendant la mise à jour*. Ceci est généralement dû à un processus prenant trop de temps et dépassant le délai de temporisation de PHP. Arrêtez alors le processus de mise à jour de cette façon:

```
sudo -u www-data php occ maintenance:mode --off
```

Puis démarrez le processus de mise à jour manuel:

```
sudo -u www-data php occ upgrade
```

Si cela ne fonctionne pas correctement, essayez la fonction de réparation:

```
sudo -u www-data php occ maintenance:repair
```

Test de migration

Avant de terminer la mise à jour, ownCloud lance d'abord une simulation en copiant toutes les tables de la base de données vers de nouvelles tables et en réalisant la mise à jour sur celles-ci pour s'assurer que la mise à jour se

passe correctement. Les tables copiées sont supprimées après la mise à jour. Cela prend donc deux fois plus de temps, ce qui sur de grosses installations peut prendre plusieurs heures. Vous pouvez omettre cette étape en utilisant l'option `--skip-migration-test`, comme dans cet exemple pour CentOS:

```
$ sudo -u apache php occ upgrade --skip-migration-test
```

Voir *Utilisation de la commande occ* pour en apprendre plus.

Migration de Encryption d'ownCloud 7.0 vers 8.0 et 8.0 vers 8.1

Le service de chiffrement a été modifié deux fois entre ownCloud 7.0 et 8.0 puis entre les versions 8.0 et 8.1. Si vous faites la mise à jour à partir de ces anciennes versions, veuillez vous référer à *upgrading_encryption_label* pour les instructions de migration.

Migration de Debian vers les paquets officiels d'ownCloud

À compter de mars 2016, Debian ne fournit plus de paquets ownCloud. Les utilisateurs de peuvent migrer vers les paquets officiels d'ownCloud en suivant ce guide : Mise à jour pour Debian Stable avec les paquets officiels.

Mise à jour d'ownCloud à l'aide des paquets

Mise à jour rapide

La meilleure méthode pour garder à jour ownCloud sur des serveurs Linux est de configurer votre système pour utiliser le dépôt ownCloud Open Build Service. Utilisez ensuite votre gestionnaire de paquets Linux pour installer les nouvelles versions d'ownCloud. Après l'installation des nouveaux paquets, il reste quelques étapes à réaliser pour terminer la mise à jour. Voici les étapes de base pour mettre à jour ownCloud :

Warning

Assurez-vos de ne pas suter de version majeure lors de la mise à jour à partir des dépôts. Par exemple, vous ne pouvez pas mettre à jour de la version 8.1.x vers la version 9.0.x directement,car vous omettriez la version majeure 8.2.x. Veuillez consulter *Mise à jour en sautant des versions* pour plus d'informations.

- *désactivez* toutes les applications tierces ;
- faites une *sauvegarde* ;
- mettez à jour les paquets ownCloud ;
- lancez la commande *occ upgrade* (facultatif : désactiver le *Test de migration* qui pourrait prendre beaucoup de temps sur de grosses installations ;
- *appliquez les permissions renforcées* sur vos répertoires ownCloud ;
- sortez le serveur ownCloud du *mode maintenance* ;
- réactivez les applications tierces.

Warning

Lors de la mise à jour de la version 9.0 vers la version 9.1 avec des agendas et des carnets d'adresses existants, veuillez consulter les *notes de version* de la version 9.0 pour des informations importantes sur les étapes de migration nécessaires pendant cette mise à jour.

Conseils de mise à jour

Mettre à jour ownCloud à partir du dépôt Open Build Service se fait comme n'importe quelle mise à jour Linux. Par exemple, pour Debian ou Ubuntu Linux, il s'agit de la commande système standard:

```
apt-get update && apt-get upgrade
```

Ou vous pouvez juste mettre à jour ownCloud avec cette commande:

```
apt-get update && apt-get install owncloud
```

Pour Fedora, CentOS et Red Hat Linux, utilisez `yum` pour voir les mises à jour disponibles:

```
yum check-update
```

Vous pouvez appliquer toutes les mises à jour avec cette commande:

```
yum update
```

Ou seulement ownCloud:

```
yum update owncloud
```

Votre gestionnaire de paquets Linux ne télécharge que les paquets ownCloud en cours. Alors, votre serveur ownCloud est mis immédiatement en mode maintenance. Vous pourriez ne pas voir ceci avant d'avoir rechargé la page d'ownCloud.

Utilisez alors `occ` pour terminer la mise à jour. Vous devez lancer `occ` en tant qu'utilisateur HTTP. Cet exemple s'applique à Debian/Ubuntu:

```
sudo -u www-data php occ upgrade
```

Cet exemple s'applique à CentOS/RHEL/Fedora:

```
sudo -u apache php occ upgrade
```

Facultatif : désactiver le *Test de migration* qui pourrait prendre beaucoup de temps sur de grosses installations.

Voir *Utilisation de la commande occ* pour en apprendre plus.

Définition de permissions renforcées pour les répertoires

Après la mise à jour, vérifiez que les permissions sont définies selon *Renforcement des permissions de répertoires*.

Mise à jour en sautant des versions

Il est vivement conseillé de mettre à jour votre installation ownCloud pour chaque mise à jour mineure (par ex. : 8.1.10), et de ne jamais omettre les mises à jour majeures (par ex. : ne sautez pas la version 8.2.x entre la 8.1.x et la 9.0.x). Si vous avez omis une mise à jour majeure, vous pouvez mettre à jour ownCloud en suivant ces étapes :

1. ajoutez le dépôt de la version en cours (par ex. : 8.1.x) ;
2. mettez à jour avec la dernière version mineure (par ex. : 8.1.10) à l'aide de votre gestionnaire de paquets ;
3. lancez la routine `occ upgrade` (voir Mise à jour rapide ci-dessus) ;
4. ajoutez le dépôt de la version majeure suivante (par ex. : 8.2.x) ;
5. mettez à jour votre version avec la dernière version majeure (par ex. : 8.2.8) à l'aide de votre gestionnaire de paquets ;;
6. lancez la routine `occ upgrade` (voir Mise à jour rapide ci-dessus) ;
7. recommencez à partir de l'étape 4 jusqu'à atteindre la dernière version majeure disponible (par ex. : 9.1.x).

Vous trouverez les dépôts des versions majeures précédentes d'ownCloud dans le fichier Changelog.

Mise à jour d'ownCloud avec l'application Updater

L'application Updater automatise beaucoup des étapes de la migration d'une installation ownCloud. Elle est utile pour les installations qui ne disposent pas d'accès root, tels que les hébergements partagés, et pour les installations avec peu d'utilisateurs et de données, et automatise les *installations manuelles*.

Warning

Lors de la mise à jour de la version 9.0 vers la version 9.1 avec des agendas et des carnets d'adresses existants, veuillez consulter les *Notes de version ownCloud 9.1* de la version 9.0 pour des informations importantes sur les étapes de migration nécessaires pendant cette mise à jour.

Nouveau dans la version 9.0, l'application Updater dispose d'*options de lignes de commande*.

Note

L'application Updater **n'est ni activée ni supportée** dans ownCloud Édition Entreprise.

L'application Updater **n'est pas incluse** dans les paquets Linux sur notre Open Build Service, mais seulement dans les archives tar et zip. Quand vous installez ownCloud à partir des paquets, vous devez les maintenir à jour avec votre gestionnaire de paquets.

Le retour arrière n'est pas supporté et risque de corrompre vos données ! Si vous voulez revenir à une ancienne version d'ownCloud, installez-la à partir de zéro et restaurez vos données à partir d'une sauvegarde. Avant de faire cela, ouvrez un ticket au support (si vous avez souscrit au support payant) ou demandez de l'aide dans les forums d'ownCloud pour voir si votre problème peut être résolu sans revenir dans la version précédente.

Vous devez mettre en place des sauvegardes régulières (voir *Sauvegarder ownCloud*), et faire une sauvegarde avant chaque mise à jour. L'application Updater ne fait pas de sauvegarde de votre base de données ou de votre répertoire de données.

L'application Updater réalise les opérations suivantes :

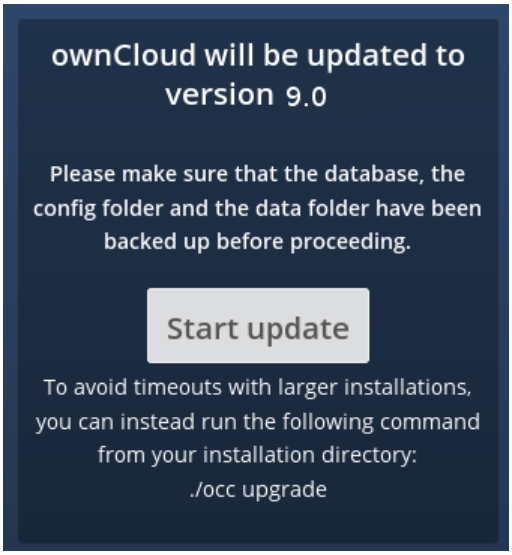
- création d'un répertoire `updater_backup` dans votre répertoire de données ownCloud ;
- téléchargement et extraction des paquets de mise à jour dans le répertoire `updater_backup/packageVersion` ;
- copie de votre instance ownCloud, à l'execption de votre répertoire de données, dans `updater_backup/currentVersion-randomstring` ;
- déplacement de tous les répertoires, à l'exception de `data`, `config` et `themes`, de votre instance vers `updater_backup/tmp` ;
- déplacement de tous les répertoires de `updater_backup/packageVersion` vers la nouvelle version ;
- copie de votre précédent fichier `config.php` vers le nouveau répertoire `config/`.

L'utilisation de l'application Updater pour mettre à jour votre installation se fait en quelques étapes :

1. vous devez observer une notification en haut de votre page ownCloud quand une mise à jour est disponible ;
2. quand bien même l'application Updater sauvegarde des répertoires importants, vous devez toujours avoir vos propres sauvegardes (voir *Sauvegarder ownCloud* pour des détails) ;
3. vérifiez que l'utilisateur HTTP de votre système puisse écrire dans tout le répertoire ownCloud (voir la section *Permissions pour la mise à jour* ci dessous) ;
4. rendez-vous sur la page d'administration et cliquez sur le bouton **Centre de mise à jour** sous Updater ; ceci vous redirige vers le panneau de contrôle de Updater ;
5. cliquez sur « Mise à jour » et lisez avec attention les messages. Si un problème est rencontré, ce sera indiqué. Le problème le plus courant concerne les permissions sur les répertoires. Votre utilisateur HTTP doit avoir les

permissions d'écriture sur tout le répertoire ownCloud (voir *Renforcement des permissions de répertoires*). Un autre problème courant concerne les règles SELinux (voir *Configuration SELinux*). Sinon, vous verrez des messages sur la vérification de votre installation et sur la nécessité de faire une sauvegarde ;

6. cliquez sur « Continuer » ; l'application réalisera alors les étapes restantes, ce qui prend quelques minutes ;
7. si les permissions de répertoires sont correctes, une sauvegarde a été faite et si le téléchargement de la nouvelle archive ownCloud a réussi, vous verrez l'écran suivant. Cliquez sur le bouton « Commencer la mise à jour » pour terminer la mise à jour :



Note

Si vous avez une installation d'ownCloud conséquente et que vous avez un accès au shell, vous devez utiliser la commande `occ upgrade`, en l'exécutant en tant que votre utilisateur HTTP, au lieu de cliquer sur le bouton de mise à jour, afin d'éviter les dépassements de temps de PHP.

Cet exemple s'applique à Ubuntu Linux

```
$ sudo -u www-data php occ upgrade
```

Facultatif : désactiver le *Test de migration* qui pourrait prendre beaucoup de temps sur de grosses installations.

Consulter *Utilisation de la commande occ* pour en apprendre plus.

8. elle s'exécute pendant quelques minutes, puis affiche un message de réussite qui disparaît après quelques secondes.

Actualisez votre page d'administration pour vérifier le numéro de version. Dans la section Updater de votre page d'administration, vous pouvez voir l'état en cours et les sauvegardes. Ce sont les sauvegardes de votre ancienne et de votre nouvelle installation (elles ne contiennent pas vos fichiers de données). Si votre mise à jour a fonctionné et que vous ne rencontrez aucun problème, vous pouvez supprimer les sauvegardes à partir de cet écran.

Si la mise à jour a échoué, vous devez mettre à jour manuellement (voir *Mise à jour manuelle*).

Permissions pour la mise à jour

Pour une sécurité renforcée, nous recommandons vivement de définir des permissions aussi strictes que possible sur votre répertoire ownCloud. Ces commandes doivent être exécutées immédiatement après l'installation initiale. Veuillez suivre les instructions dans *Renforcement des permissions de répertoires*.

Ces permissions strictes empêchent l'application Updater de fonctionner, car elle a besoin que tout le répertoire ownCloud appartienne à l'utilisateur HTTP. Exécutez ce script pour définir les permissions appropriées pour la mise à jour. Définissez la variable `ocpath` avec le chemin d'accès de votre répertoire ownCloud et les variables `htuser` et `htgroup` avec l'utilisateur et le groupe HTTP

```
#!/bin/bash
# Sets permissions of the owncloud instance for updating

ocpath='/var/www/owncloud'
htuser='www-data'
htgroup='www-data'

chown -R ${htuser}:${htgroup} ${ocpath}
```

Vous pouvez trouver l'utilisateur et le groupe HTTP dans les fichiers de configuration du serveur. Ou vous pouvez utiliser *Informations et version de PHP* (recherchez la ligne **Utilisateur/Groupe**).

- L'utilisateur et le groupe HTTP pour Debian/Ubuntu est : `www-data`.
- L'utilisateur et le groupe HTTP pour Fedora/CentOS est : `apache`.
- L'utilisateur et le groupe HTTP pour Arch Linux est : `http`.
- L'utilisateur HTTP pour openSUSE est : `wwwrun` et le groupe : `www`.

Quand la mise à jour est terminée, réappliquez immédiatement les permissions strites sur les répertoires en exécutant le script indiqué dans *Renforcement des permissions de répertoires*.

Options de ligne de commande

L'application Updater dispose d'options de ligne de commande pour automatiser les mises à jour, créer des points de contrôle et pour revenir sur de précédents points de contrôle. Vous devez l'exécuter en tant qu'utilisateur HTTP. Cet exemple s'applique à Ubuntu Linux et affiche les options de commande

```
sudo -u www-data php updater/application.php list
```

Consulter l'utilisation pour les commandes, comme dans cet exemple pour la commande `upgrade:checkpoint`

```
sudo -u www-data php updater/application.php upgrade:checkpoint -h
```

Vous pouvez afficher un résumé de l'aide

```
sudo -u www-data php updater/application.php --help
```

Quand vous l'exécutez sans option, elle lance une vérification du système

```
sudo -u www-data php owncloud/updater/application.php
ownCloud updater 1.0 - CLI based ownCloud server upgrades
Checking system health.
- file permissions are ok.
Current version is 9.0.0.12
No updates found online.
Done
```

Création d'un point de contrôle

```
sudo -u www-data php updater/application.php upgrade:checkpoint  --create
Created checkpoint 9.0.0.12-56d5e4e004964
```

Affichage des points de contrôle

```
sudo -u www-data php updater/application.php upgrade:checkpoint --list
```

Restauration d'un point de contrôle précédent

```
sudo -u www-data php owncloud/updater/application.php upgrade:checkpoint
 --restore=9.0.0.12-56d5e4e004964
```

Ajoutez une ligne comme celle-ci dans votre crontab pour créer automatiquement chaque jour des points de contrôle

```
2 15 * * * sudo -u www-data php /path/to/owncloud/updater/application.php
upgrade:checkpoint --create > /dev/null 2>&1
```

Manuel de mise à jour d'ownCloud

Warning

Lors de la mise à jour de la version 9.0 vers la version 9.1 avec des agendas et des carnets d'adresses existants, veuillez consulter les *Notes de version ownCloud 9.1* de la version 9.0 pour des informations importantes sur les étapes de migration nécessaires pendant cette mise à jour.

Toujours commencer par faire une sauvegarde et par désactiver les applications tierces.

Passer ensuite le serveur en mode maintenance. Ceci empêche les nouvelles connexions, verrouille les sessions des utilisateurs connectés et affiche un écran aux utilisateurs leur indiquant ce qu'il se passe. Il y a deux façons pour faire cela, et la méthode préférée et d'utiliser la *commande occ*, que vous devez lancer en tant qu'utilisateur HTTP. Cet exemple s'applique à Ubuntu Linux:

```
sudo -u www-data php occ maintenance:mode --on
```

L'autre façon est de modifier votre fichier `config.php` et de changer `'maintenance' => false,` pour `'maintenance' => true,`.

1. Sauvegardez la base de données ownCloud, le répertoire `data` et le fichier `config.php` (voir *Sauvegarder ownCloud*).
2. Téléchargez et décompressez la dernière version de l'archive d'ownCloud sur owncloud.org/install/ dans un répertoire vide en dehors de votre installation actuelle.

   ### Note

   Pour décompresser l'archive, utilisez : tar xjf owncloud-[version].tar.bz2

Note

Les utilisateurs de l'Édition Entreprise doivent télécharger la nouvelle archive ownCloud à partir de leur compte sur https://customer.owncloud.com/owncloud/

3. Arrêtez le serveur Web.
4. Renommez votre répertoire actuel ownCloud, par exemple en `owncloud-old`.
5. Décompresser la nouvelle archive crée un nouveau répertoire``owncloud`` contenant les nouveaux fichiers serveur. Copiez la répertoire et son contenu dans l'emplacement original de l'ancien serveur, par exemple `/var/www/`, pour avoir à nouveau `/var/www/owncloud`.
6. Copiez le fichier `config.php` de l'ancien répertoire ownCloud vers le nouveau.
7. Si vos conservez le répertoire `data/` dans le répertoire `owncloud/`, copiez-le aussi de l'ancien répertoire serveur vers le nouveau. Si vous le conservez en dehors du répertoire `owncloud/`, alors, vous n'avez rien à faire car son emplacement est indiqué dans votre fichier original `config.php` et qu'il n'est pas concerné par les étapes de mises à jour.
8. Si vous utilisez des applications tierces, regardez dans le nouveau dossier `owncloud/apps/` pour voir si elles y sont. Si ce n'est pas le cas, copiez-les de votre ancien répertoire `apps/` vers le nouveau. Assurez-vous que les permissions des répertoires de vos applications tierces soient les mêmes que celles des autres applications.
9. Redémarrez le serveur Web.
10. Lancez maintenant la mise à jour avec la commande `occ`, comme dans cet exemple pour CentOS Linux:

    ```
    sudo -u apache php occ upgrade
    ```

Facultatif : désactiver le *Test de migration* qui pourrait prendre beaucoup de temps sur de grosses installations.

Voir *Utilisation de la commande occ* pour en apprendre plus.

11. L'opération de mise à jour prend de quelques minutes à quelques heures en fonction de la taille de votre installation. Quand c'est terminé, vous verrez un message de réussite, ou d'erreur indiquant ce qui s'est mal passé.

En supposant que la mise à jour s'est bien passée, désactivez le mode de maintenance:

```
sudo -u www-data php occ maintenance:mode --off
```

Connectez-vous et regardez en bas de la page d'administration pour vérifier le numéro de version. Vérifiez les autres paramètres pour vous assurez qu'ils sont corrects. Rendez-vous sur la pages des applications et vérifiez les applications de base du système pour vous assurer que les bonnes sont activées. Réactivez ensuite les applications tierces. Appliquez ensuite les permissions renforcées sur les répertoires d'ownCloud (*Renforcement des permissions de répertoires*).

Restauration d'ownCloud

Quand vous installez ownCloud à partir des paquets, suivez ces étapes fpour restaurer votre installation ownCloud. Commencez par installer ownCloud dans un nouveau répertoire vide. Puis restaurez les éléments suivants à partir de votre sauvegarde (see *Sauvegarder ownCloud*) :

1. le dossier de configuration `config/` ;
2. le dossier de données `data/` ;
3. la base de données d'ownCloud ;
4. vos fichiers de thèmes personnalisés, si vous en avez (voir Thèmes ownCloud).

Quand vous installez ownCloud à partir d'une archive, vous pouvez sans risque restaurer toute l'installation ownCloud à partir de la sauvegarde, à l'exception de la base de données ownCloud. Les bases de données ne peuvent pas être copiées, mais vous devez utiliser les outils de base de données appropriés pour faire une restauration de base correcte.

Quand la restauration est terminée, consultez *Renforcement des permissions des répertoires*.

Restauration des répertoires

Copiez simplement vos répertoires de configuration et de données dans votre nouvel environnement. Vous pouvez utiliser cette commande, qui restaure la sauvegarde donnée en exemple dans *Sauvegarder ownCloud*

```
rsync -Aax config data /var/www/owncloud/
```

Il existe plusieurs moyens de sauvegarder des fichiers normaux, et vous pouvez utiliser la méthode à laquelle vous êtes habitué.

Restauration de la base de données

> **Note**
>
> Ce guide suppose que la sauvegarde soit nommée « owncloud-bdd-svg.bak ».

MySQL

MySQL est le moteur de base de données recommandé. Pour restaurer MySQL:

```
mysql -h [serveur] -u [utilisateur] -p[mot-de-passe] [nom_base_de_données] < owncloud-bdd-sv
```

SQLite

```
rm data/owncloud.db
sqlite3 data/owncloud.db < owncloud-bdd-svg.bak
```

PostgreSQL

```
PGPASSWORD="mot-de-passe" pg_restore -c -d owncloud -h [serveur] -U [utilisateur]
owncloud-bdd-svg.bak
```

Migration sur un autre serveur

Si le besoin s'en fait sentir, ownCloud peut être migré sur un serveur différent. Une raison classique est le changement de matériel ou le passage d'un serveur virtuel à un serveur physique. Tout les migrations doivent être effectuées avec ownCloud hors ligne et aucun accès. La migration en ligne est supportée pour ownCloud seulement en mettant en œuvre des systèmes des clusters et des solutions de haute disponibilité avant la première installations d'ownCloud.

Pour commencer, soyons clair sur ce cas d'usage. Une instance configurée d'ownCloud fonctionne de façon fiable sur un serveur. Si pour une raison (par exemple, une machine plus puissante est disponible mais une migration vers un environnement en cluster n'est pas nécessaire), l'instance a besoin d'être déplacée sur une nouvelle machine. En fonction de la taille de votre instance ownCloud, la migration peut prendre plusieurs heures. Il est supposé que les utilisateurs accèdent à l'instance ownCloud à l'aide d'un hôte virtuel (un enregistrement `CNAME` dans un DNS) qui peut être dirigé vers le nouvel emplacement. Il est aussi supposé que la méthode d'authentification (par exemple LDAP) reste la même après la migration.

Warning

À AUCUN MOMENT des modifications du système d'**ORIGINE** ne sont nécessaires **EXCEPTÉ** de mettre ownCloud en mode maintenance.

Ceci permet, si quelque chose se passait mal, de revenir à votre installation existante pour fournir à vos utilisateurs une d"instance d'ownCloud fonctionnelle pendant que vous déboguez le problème.

1. Montez la nouvelle machine avec la distribution Linux désirée. Dès lors, vous pouvez installer ownCloud manuellement avec l'archive compressée (voir *Manuel d'installation pour Linux*), ou avec le gestionnaire de paquets de votre distribution Linux (voir *Méthode d'installation préférée pour Linux*).
2. Sur la machine d'origine, passez en mode maintenance puis arrêtez ownCloud. Attendez 6 ou 7 minutes pour que tous les clients de synchronisation enregistrent que le serveur est en mode maintenance. Arrêtez l'application et/ou le serveur Web qui sert ownCloud (voir *Commandes de maintenance*).
3. Faites un export de la base de données et copiez-le sur la nouvelle machine. Importez-le dans la nouvelle base de données (voir *Sauvegarder ownCloud* et *Restauration d'ownCloud*).
4. Copiez SEULEMENT les fichiers de données, les fichiers de configuration et la base de données de l'instance d'origine vers la nouvelle machine (voir *Sauvegarder ownCloud* et *Restauration d'ownCloud*). Les fichiers de données doivent conserver leur horodatage d'origine (ceci peut être fait en utilisant `rsync` avec l'option `-t`), sans quoi, les clients re-téléchargeront tous les fichiers après la migration. Cette étape peut prendre plusieurs heures en fonction de votre installation.

Note

Vous devez conserver le chemin d'accès `data/` d'origine. Ne le modifiez pas !

1. Avec l'ancien serveur toujours en mode maintenance (confirmez !) et **AVANT** de changer l'enregistrement `CNAME` dans le DNS, démarrez la base de données, le serveur Web / serveur applicatif sur la nouvelle machine et pointez votre navigateur Web sur la nouvelle instance ownCloud. Vérifiez que l'avertissement du mode maintenance soit toujours actif, qu'une entrée soit écrite dans le fichier journal du serveur Web et de

celui d'ownCloud et qu'aucune erreur n'est signalée. Passez alors ownCloud hors du mode maintenance. Connectez-vous en tant qu'administrateur et vérifiez le bon fonctionnement d'ownCloud.

2. Modifiez ensuite l'enregistrement `CNAME` dans le serveur DNS pour que vos utilisateurs soient dirigés sur le nouveau serveur.

Problèmes et dépannage

Dépannage général

Si vous avez un problème d'installation, de configuration ou de maintenance d'ownCloud, veuillez vous référeraux canaux de support de la communauté :

- Les forums d'ownCloud

> ***Note***
>
> Les forums d'ownCloud ont une catégorie FAQ où chaque sujet correspond aux erreurs classiques et fréquentes.

- La liste de distribution utilisateur d'ownCloud
- Le canal IRC d'ownCloud `irc://#owncloud@freenode.net` sur freenode.net, également accessible par webchat.

Veuillez comprendre que tous ces canaux sont constitués d'utilisateurs comme vous qui s'entraident. Envisagez d'aider les autres sur les sujets que vous pouvez en retour de l'aide qui vous a été apportée. C'est le seul moyen de conserver une communauté comme ownCloud saine et durable !

Si vous utilisez ownCloud en entreprise ou dans des déploiements d'envergure, veuillez noter que ownCloud Inc. propose l'Édition Entreprise avec des options de support commercial.

Bogues

Si vous pensez avoir trouvé un bogue dans ownCloud, veuillez :

- chercher une solution (voir options ci-dessus) ;
- vérifier à nouveau votre configuration.

Si vous ne trouvez pas de solution, veuillez utiliser notre logiciel de suivi de bogues. Vous pouvez à cet effet générer un rapport de configuration avec la *commande config d'occ*, avec les mots de passe caviardés automatiquement.

Dépannage général

Vérifiez les *Prérequis système*, d'ownCloud et particulièrement les versions de navigateurs supportées.

Si vous rencontrez des avertissements d'`intégrité de code`, veuillez consulter *Signature du code*.

Désactivation des applications tierces / non livrées

Il peut être possible que des applications tierces, non fournies par ownCloud provoquent des problèmes. Désactivez toujours ces applications avant les mises à jour et pour diagnostiquer des dysfonctionnements. Veuillez consulter *Commandes pour les applications* pour savoir comment désactiver une application en ligne de commande.

Journaux d'ownCloud

Dans une installation standard d'ownCloud, le niveau de journalisation est défini à `Normal`. Pour diagnistiquer des problèmes, vous devez élever ce niveau à `All` dans le fichier `config.php`, ou à **Tout** dans la page d'administration d'ownCloud. Veuillez consulter *Configuration des fichiers journaux* pour plus d'informations sur ces niveaux de journalisation.

Certains journaux - par exemple la console JavaScript - nécessite que le débogage soit activé. Modifiez pour cela le fichier config/config.php et modifiez la ligne `'debug' => false,` pour `'debug' => true,`. Assurez-vous de repasser à `false` quand vous avez terminé.

Pour les problèmes JavaScript, vous devez aussi consulter la console JavaScript. Les principaux navigateurs disposent d'outils développeurs pour voir cette console, qui s'affiche généralement en appuyant sur la touche F12. Pour Firefox, nous recommandons l'installation de l'extension Firebug.

Note

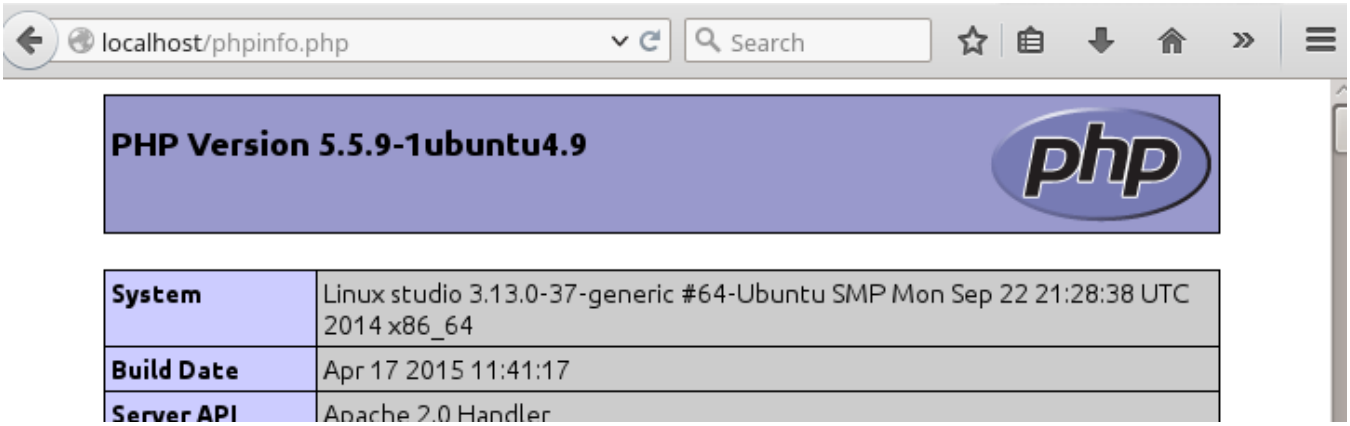
Le fichier journal d'ownCloud et situé dans le répertoire `data`` : ``owncloud/data/owncloud.log`.

Informations et version de PHP

Vous aurez besoin de connaître la version et la configuration de PHP. Pour cela, créez un fichier texte nommé **phpinfo.php** el placez-le dans la racine Web, par exemple `/var/www/html/phpinfo.php`. Elle peut être située ailleurs en fonction de votre distribution Linux ; consultez la documentation de votre distribution pour savoir où elle se situe. Ce fichier contient juste cette ligne:

```
<?php phpinfo(); ?>
```

Ouvrez ce fichier dans un navigateur Web à l'adresse `localhost/phpinfo.php` :



Votre version de PHP est située en haut de la page et le reste de la page contient d'abondantes informations système comme les modules actifs, les fichiers `.ini` utilisés et bin d'autres choses. Quand vous avez terminé de vérifier vos informations, vous devez supprimer le fichier `phpinfo.php` ou le déplacer en dehors de votre répertoire Web car c'est un risque de sécurité que d'exposer des informations aussi sensibles.

Débogage des problèmes de synchronisation

Warning

Le répertoire `data` sur le serveur est exclusif à ownCloud et vous ne devez pas le modifier manuellement.

Ne pas tenir compte de cela peut conduire à des comportements inattendus comme :

- des problèmes de synchronisation clients ;
- des changements non détectés en raison du cache de la base de données.

Si vos avez besoin de téléverser des fichiers du même serveur, veuille utiliser une ligne de commande WebDAV comme `cadaver` pour téléverser les fichiers vers l'interface WebDAV sur :

`https://exemple.com/owncloud/remote.php/dav`

Problèmes courants / messages d'erreur

Des problèmes courrants / messages d'erreur trouvés dans vos fichiers journaux sont décrits ci-dessous :

- `SQLSTATE[HY000] [1040] Too many connections` -> Vous devez augmenter le nombre limite de connexions à votre base de données, veuillez consulter le manuel de votre base de données pour plus d'informations.
- `SQLSTATE[HY000]: General error: 5 database is locked` -> Vous utilisez `SQLite` qui ne sait pas gérer beaucoup de requêtes en parallèle. Veuillez envisager de convertir votre base vers un autre système de base de données comme décrit dans *Conversion du moteur de base de données*.
- `SQLSTATE[HY000]: General error: 2006 MySQL server has gone away` -> La requête à la base de données prend trop de temps et par conséquent, le serveur MySQL dépasse le délai de temporisation. Il est également possible que le serveur rejette un paquet parce qu'il est trop gros. Veuillez consulter le manuel de votre base de données pour savoir comment augmenter les options de configuration `wait_timeout` et/ou `max_allowed_packet`.
- `SQLSTATE[HY000] [2002] No such file or directory` -> Il y a un problème d'accès au fichier de base de données SQLite dans votre répertoire `data` (`data/owncloud.db`). Veuillez vérifier les permissions du dossier/fichier ou s'il existe. Si vous utilisez MySQL, veuillez démarrer votre base de données.
- `Connection closed / Operation cancelled` ou `expected filesize 4734206 got 458752` -> Ceci peut être provoqué par un mauvais paramétrage de `KeepAlive` dans votre configuration Apache. Assurez-vous que `KeepAlive` est défini à `On` et essayez aussi d'augmenter les limites de `KeepAliveTimeout` et `MaxKeepAliveRequests`. Sous Apache avec `mod_php`, utiliser un module différent de `prefork` (*Module Multi-Processing (MPM)*) peut être une autre explication. De plus amples informations sont disponibles sur nos forums.
- `No basic authentication headers were found` -> Cette erreur est affichée dans votre fichier journal `data/owncloud.log`. Certains modules Apache tels que `mod_fastcgi`, `mod_fcgid` ou `mod_proxy_fcgi` ne passent pas les en-têtes d'authentification requis à PHP et la connexion à ownCloud par les clients WebDAV, CalDAV et CardDAV échoue. Vous trouverez des informations sur la manière de configurer coorectement votre environnement sur les forums.

Dépannage du serveur Web et problèmes PHP

Fichiers journaux

Quand des problèmes surviennent, la première étape est de consulter les fichiers journaux de PHP, du serveur Web et d'ownCloud lui-même.

Note

Dans les exemples suivants, il est supposé qu'il s'agit d'une installation par défaut Debian avec Apache2 et mod_php. Sur les autres serveurs Web, distributions Linux ou systèmes d'exploitation, ces chemins peuvent être différents.

- Le fichier journal d'Apache2 se situe dans `/var/log/apache2/error.log` ;
- le fichier journal de PHP peut être configuré dans votre fichier `/etc/php5/apache2/php.ini`. Vous devez définir la directive `log_errors` à `On` et choisir le chemin où stocker le fichier journal dans la directive `error_log` puis redémarrer le serveur Web ;
- le fichier journal d'ownCloud se situe dans le répertoire `data` : `/var/www/owncloud/data/owncloud.log`.

Serveur Web et modules PHP

Note

Lighttpd n'est pas supporté pour ownCloud et certaines fonctionnalités pourraient ne pas fonctionner du tout avec Lighttpd.

Il existe certains serveurs Web ou modules PHP connus pour provoquer divers problèmes, telle que la corruption de fichiers téléchargés ou téléversés. Voici un aperçu de ces modules :

1. Apache
   - mod_pagespeed ;
   - mod_evasive ;
   - mod_security ;
   - mod_reqtimeout ;
   - mod_deflate ;
   - libapache2-mod-php5filter (utiliser libapache2-mod-php5 à la place) ;
   - mod_spdy avec libapache2-mod-php5 / mod_php (utilisez fcgi ou php-fpm à la place) ;
   - mod_dav ;
   - mod_xsendfile / X-Sendfile (provoque la corruption des téléchargements s'il n'est pas configuré correctement).
2. NginX
   - ngx_pagespeed ;
   - HttpDavModule ;
   - X-Sendfile (provoque la corruption des téléchargements s'il n'est pas configuré correctement).
3. PHP
   - eAccelerator.

Dépannage WebDAV

ownCloud utilise SabreDAV et la documentation de SabreDAV est exhaustive et utile.

Consulter :

- la FAQ SabreDAV ;
- les serveurs Web (liste lighttpd comme non recommandé) ;
- la manipulation de gros fichiers (décrit un bogue de PHP dans les anciennes version de SabreDAV et des informations sur les problèmes de mod_security) ;
- les fichiers de 0 octet (raisons pour les fichiers vides sur le serveur) ;
- les clients (une liste exhaustive de clients WebDAV et les problèmes possibles pour chacun d'eux) ;
- Finder, le client WebDAV intégré d'OS X (décrit des problèmes de Finder sur divers serveurs Web).

Il existe aussi un sujet de forum très bien maintenu et faisant office de FAQ sur les forums d'ownCloud qui contient des informations additionnelles sur les problèmes avec WebDAV.

Error 0x80070043 "The network name cannot be found." while adding a network drive

Le client WebDAV natif de Windows peut dysfonctionner avec le message d'erreur suivant

```
Erreur 0x80070043 « Le nom de réseau ne peut être trouvé. » lors de l'ajout d'un lectuer rés
```

Un moyen de contourner cette erreur est de mettre à jour la configuration de votre serveur Web. Pour Apache, vous devez ajouter quelque chose de similaire à ceci (en adptant le chemin d'accès) dans la configuration générale du serveur Web ou dans le fichier vhost de votre serveur ownCloud, ou encore dans le fichier `.htaccess` placé à la racine du serveur ownCloud

```
RewriteEngine On
RewriteCond %{REQUEST_URI} ^(/)$ [NC]
RewriteCond %{REQUEST_METHOD} ^(OPTIONS)$
RewriteRule .* https://%{SERVER_NAME}/owncloud/remote.php/webdav/ [R=301,L]
```

Pour nginx, un exemple pourrait être comme ceci

```
location = / {
    if ($http_user_agent = DavClnt) {
        return 401;
    }
}
```

Dépannage de Contacts & Agenda

Service de découverte

Certains clients - particulièrement sur iOS et Mac OS X - ont des problèmes pour trouver la bonne URL de synchronisation, même quand elle est explicitement indiquée.

Si vous voulez utiliser des clients CalDAV et CardDAV avec ownCloud, il est important d'avoir une configuration correcte pour les URLs suivantes :

`https://exemple.com/.well-known/carddav`
`https://exemple.com/.well-known/caldav`

Elles doivent rediriger vos clients vers le frontal DAV correct. Si vous exécutez ownCloud dans la racine document de votre serveur Web, l'URL correcte est celle-ci :

`https://exemple.com/remote.php/dav`

et dans un sous-dossier `owncloud` :

`https://exemple.com/owncloud/remote.php/dav`

Pour le premier cas, le fichier .htaccess fourni avec ownCloud devrait faire le travil pour vous avec Apache. Vous devez seulement vcous assurer que le serveur Web utilise ce fichier. Pour NGINX, veuillez vous référer à *Exemples de configurations pour Nginx*.

Si votre instance est installée dans un sous-dossier appelé `owncloud` et que vous utilisez Apache, créez ou modifier le fichier .htaccess dans la racine document de votre serveur Web et ajoutez les lignes suivantes:

```
Redirect 301 /.well-known/carddav /owncloud/remote.php/dav
Redirect 301 /.well-known/caldav /owncloud/remote.php/dav
```

Maintenant, changez juste les paramètres de votre client pour utiliser juste :

`https://exemple.com`

au lieu de, par exemple :

`https://exemple.com/owncloud/remote.php/dav/principals/utilisateur`.

Il existe aussi plusieurs techniques pour remédier à cela, qui sont toutes décrites sur le site Web de Sabre DAV.

Impossible de mettre à jour les contacts ou les événements

Si vous obtenez une erreur :

`PATCH https://example.com/remote.php/dav HTTP/1.0 501 Not Implemented`

il est probable qu'elle soit provoquée par l'une des raisons suivantes :

Utilisation du reverse proxy/équilibreur de charge Pound

Au moment de la rédaction de ceci, Pound ne supportait pas le verbe HTTP/1.1. Pound est facilement corrigé pour gérer HTTP/1.1.

Serveur Web mal configuré

Votre serveur Web est mal configuré et bloque les méthodes DAV nécessaires. Veuillez consulter *Dépannage WebDAV* pour les étapes de dépannage.

La synchronisation client s'arrête

Une des raisons connues concerne les verrous. Ils doivent expirer automatiquement après une heure. Si ce n'est pas le cas (identifié par exemple avec des messages répétés `file.txt is locked` et/ou `Exception\\\\FileLocked` dans votre fichier data/owncloud.log), assurez-vous que le cron système soit actif, et non le cron Ajax (voir *Tâches d'arrière-plan*). Consulter https://github.com/owncloud/core/issues/22116 et https://central.owncloud.org/t/file-is-locked-how-to-unlock/985 pour une discussion et des informations supplémentaires sur ce sujet.

Autres problèmes

Certains services comme *Cloudflare* peuvent provoquer des problèmes en minimisant JavaScript et en le chargeant seulement quand c'est nécessaire. Si vous rencontrez des problèmes comme un bouton de connexion qui ne fonctionne pas ou la création de nouveaux utilisateurs qui échoue, assurez-vous de désactiver ce genre de service d'abord.

Signature du code

ownCloud gère la signature du code pour ownCloud et pour les applications. La signature du code donne à nos utilisateurs une couche de sécurité supplémentaire en assurant que personne d'autre que les personnes autorisées ne puissent mettre à disposition les mises à jour.

Cela assure aussi a*que les mises à jour ont exécutées correctement, de sorte qu'aucun ancien fichier ne reste en place et qu'ils soient tous correctement remplacés. Par le passé, des mises à jour invalides avaient provoqué beaucoup d'erreur lors de la mise à jour d'ownCloud.

FAQ

Pourquoi ownCloud a-t-il ajouté la signature de code ?

La signature du code donne à nos utilisateurs une couche de sécurité supplémentaire en assurant que personne d'autre que les personnes autorisées ne puissent mettre à disposition les mises à jour.

Verrouillons-nous ownCloud ?

Le projet ownCloud est open-source et le demeurera. Nous ne voulons pas ajouter de difficultés à nos utilisateurs pour exécuter ownCloud. Toute erreur de signature de code lors des mises à jour n'empêchera pas ownCloud de fonctionner, mais affichera un avertissement sur la page d'administration. Pour les applications non marquées « Officielle », la signature de code est facultative.

Plus open-source ?

Le projet ownCloud est open-source et le demeurera. La signature du code est un processus facultatif, bien que vivement recommandé. La vérification du code est activée pour ownCloud quand la branche de version a été définie à `stable`.

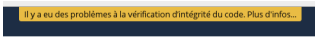
Pour les distributions personnalisées d'ownCloud, il est recommandé de modifier la branche pour autre chose que `stable`.

La signature du code est-elle obligatoire pour les applications ?

La signature de code est facultative pour toutes les applications tiers. Les applications marquées « Officielle » sur apps.owncloud.com nécessitent la signature de code.

Correction des messages d'intégrité de code invalide

Un message d'erreur d'intégrité de code (« Il y a eu des problèmes à la vérification d'intégrité du code. Plus d'infos… ») apparaît dans une bannière jaune en haut de l'écran dans l'interface Web d'ownCloud :



> ***Note***
>
> La bannière jaune n'est affichée que pour les administrateurs.

Cliquer sur ce lien vous aménera sur la page d'administration d'ownCloud, qui propose les options suivantes :

1. un lien vers cette entrée de documentation ;
2. la liste des fichiers non valides ;
3. une revérification.

Pour déboguer les problèmes provoqués par la vérification d'intégrité de code, cliquez sur « Liste des fichiers non valides… » et un document texte sera affiché listant les différents problèmes. Le contenu de ce fichier sera similaire à ce qui suit :

```
Technical information
=====================
The following list covers which files have failed the integrity check. Please read
the previous linked documentation to learn more about the errors and how to fix
them.

Results
=======
- core
      - INVALID_HASH
            - /index.php
            - /version.php
      - EXTRA_FILE
            - /test.php
- calendar
      - EXCEPTION
            - OC\IntegrityCheck\Exceptions\InvalidSignatureException
            - Signature data not found.
- tasks
      - EXCEPTION
            - OC\IntegrityCheck\Exceptions\InvalidSignatureException
            - Certificate has been revoked.

Raw output
==========
Array
(
    [core] => Array
        (
            [INVALID_HASH] => Array
                (
                    [/index.php] => Array
                        (
                            [expected] =>
                            f1c5e2630d784bc9cb02d5a28f55d6f24d06dae2a0fee685f3
                            c2521b050955d9d452769f61454c9ddfa9c308146ade10546c
                            fa829794448eaffbc9a04a29d216
                            [current] =>
                            ce08bf30bcbb879a18b49239a9bec6b8702f52452f88a9d321
                            42cad8d2494d5735e6bfa0d8642b2762c62ca5be49f9bf4ec2
```

```
                            31d4a230559d4f3e2c471d3ea094
                        )

                    [/version.php] => Array
                        (
                            [expected] =>
                            c5a03bacae8dedf8b239997901ba1fffd2fe51271d13a00cc4
                            b34b09cca5176397a89fc27381cbb1f72855fa18b69b6f87d7
                            d5685c3b45aee373b09be54742ea
                            [current] =>
                            88a3a92c11db91dec1ac3be0e1c87f862c95ba6ffaaaa3f2c3
                            b8f682187c66f07af3a3b557a868342ef4a271218fe1c1e300
                            c478e6c156c5955ed53c40d06585
                        )

                )

            [EXTRA_FILE] => Array
                (
                    [/test.php] => Array
                        (
                            [expected] =>
                            [current] =>
                            09563164f9904a837f9ca0b5f626db56c838e5098e0ccc1d8b
                            935f68fa03a25c5ec6f6b2d9e44a868e8b85764dafd1605522
                            b4af8db0ae269d73432e9a01e63a
                        )

                )

        )

    [calendar] => Array
        (
            [EXCEPTION] => Array
                (
                    [class] => OC\IntegrityCheck\Exceptions\InvalidSignature
                    Exception
                    [message] => Signature data not found.
                )

        )
    [tasks] => Array
        (
            [EXCEPTION] => Array
                (
                    [class] => OC\IntegrityCheck\Exceptions\InvalidSignatureException
                    [message] => Certificate has been revoked.
                )

        )

)
```

Dans la sortie d'erreur ci-dessus, on peut voir que :

1. dans le cœur d'ownCloud (c'est-à-dire le serveur ownCloud lui-même) les « index.php » et « version.php » n'ont pas la bonne version ;
2. dans le cœur d'ownCloud, le fichier supplémentaire « /test.php » a été trouvé ;
3. il n'a pas été possible de vérifier la signature de l'application Agenda.

4. Le certificat de l'application tâche a été révoqué.

Pour résoudre ce problème, les tâches suivantes doivent être réalisées :

1. téléversez les fichiers « index.php » et « version.php » corrects par exemple à partir de l'archive de votre version d'ownCloud ;
2. supprimez le fichier « test.php » ;
3. contactez le développeur de cette application : une nouvelle version de l'application contenant un fichier de signature valide doit être publiée ;
4. contactez le développeur de cette application : une nouvelle version de l'application signée avec une signature valide doit être publiée.

Pour d'autres moyens de recevoir de l'assistance, veuillez consulter https://owncloud.org/support/. Après avoir corrigé ces problèmes, cliquez sur « Relancer… ».

Note

Lors de l'utilisation d'un client FTP pour téléverser ces fichiers, assurez-vous d'utiliser le mode de transfert `binaire` plutôt qu'`ASCII`.

Vérifications

Elles sont déclenchées à l'installation et lors des mises à jour. Vous pouvez lancer les vérifications manuellement avec la commande `occ`. La première vérifie les fichiers du cœur d'ownCloud et la seconde ceux de l'application. Il n'existe pas encore de commande pour vérifier manuellement toutes les applications:

```
occ integrity:check-core
occ integrity:check-app $appid
```

Consulter *Utilisation de la commande occ* pour en apprendre plus sur la commande `occ`.

Erreurs

Warning

Veuillez ne pas modifier le fichier `signature.json` mentionné lui-même.

Les erreurs suivantes peuvent être rencontrées lors de la vérification de la signature du code :

- `INVALID_HASH`
  - Le fichier a un « hash » différent de celui mentionné dans `signature.json`. Ceci arrive généralement quand le fichier a été modifié après l'écriture des données de signature.
- `MISSING_FILE`
  - Le fichier ne peut être trouvé mais il a été indiqué dans le fichier `signature.json`. Un fichier a été oublié ou le fichier `signature.json` doit être modifié.
- `EXTRA_FILE`
  - Le fichier n'existe pas dans `signature.json`. Ceci arrive généralement quand un fichier a été enlevé et que `signature.json` n'a pas été mis à jour. Cela survient aussi si vous avez ajouté des fichiers supplémentaires dans le dossier d'installation d'ownCloud.
- `EXCEPTION`
  - Une autre exception a empêché la vérification du code. Il existe actuellemnt les exceptions suivantes :
    - `Signature data not found.`

- L'application est marquée comme devant être signée, mais aucun fichier `signature.json` n'a été trouvé dans le dossier `appinfo`.

- `Certificate is not valid.`
  - Le certificat n'a pas été délivré par l'autorité racine de certification officielle d'ownCloud.
- `Certificate is not valid for required scope. (Requested: %s, current: %s)`
  - Le certificat n'est pas valide pour l'application définie. Les certificats ne sont valides que pour l'identifiant d'application défini et ne peuvent être utilisé pour d'autres.
- `Signature could not get verified.`
  - Il y a eu un problème en vérifiant la signature de `signature.json`.
- `Certificate has been revoked.`
  - Le certificat utilisé pour signer cette application a été révoqué.

Édition Entreprise seulement

Installation de l'Édition Entreprise

Installation et mise à jour de l'Édition Entreprise

The recommended method for installing and maintaining your ownCloud Enterprise edition is with your Linux package manager. Configure your package manager to use the ownCloud Enterprise repository, import the signing key, and then install and update ownCloud packages like any other software package. Please refer to the `README - ownCloud Package Installation.txt` document in your account at Customer.owncloud.com account for instructions on setting up your Linux package manager.

After you have completed your initial installation of ownCloud as detailed in the README, follow the instructions in *Assistant d'installation* to finish setting up ownCloud.

To upgrade your Enterprise server, refer to *Comment mettre à jour votre serveur ownCloud*.

Manual Installation

Download the ownCloud archive from your account at https://customer.owncloud.com/owncloud, then follow the instructions at *Manuel d'installation pour Linux*.

SELinux

Linux distributions that use SELinux need to take some extra steps so that ownCloud will operate correctly under SELinux. Please see *Configuration SELinux* for some recommended configurations.

Applications gérées pour l'Édition Entreprise

See *Applications gérées dans ownCloud* for a list of supported apps.

> ***Note***
>
> 3rd party and unsupported apps must be disabled before performing a system upgrade. Then install the upgraded versions, and after the upgrade is complete re-enable them.

Clés de licence

Introduction

You'll need to install a license key to use ownCloud Enterprise Edition. There are two types of license keys: one is a free 30-day trial key. The other is a full license key for Enterprise customers.

You can download and try ownCloud Enterprise for 30 days for free, which auto-generates a free 30-day key. When this key expires your ownCloud installation is not removed, so when you become an Enterprise customer you can enter your new key to regain access. See How to Buy ownCloud for sales and contact information.

Configuration

Once you get your Enterprise license key, it needs to be copied to your ownCloud configuration file, `config/config.php` file like this example:

```
'license-key' => 'test-20150101-XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX-YYYYYY',
```

Each running instance of ownCloud requires a license key. Keys will work across upgrades without issue, so new keys will not be required when you upgrade your ownCloud Enterprise to a new version.

Configuration de la base de données Oracle

This document will cover the setup and preparation of the ownCloud server to support the use of Oracle as a backend database. For the purposes of testing, we are using Oracle Enterprise Linux as both the Web server that will host ownCloud, and as a host for the Oracle Database.

Outline of Steps

This document will cover the following steps:

- Setup of the ownCloud user in Oracle: This involves setting up a user space in Oracle for setting up the ownCloud database.
- Installing the Oracle Instant Client on the Web server (facilitating the connection to the Oracle Database).
- Compiling and installing the Oracle PHP Plugin oci8 module
- Pointing ownCloud at the Oracle database in the initial setup process

The document assumes that you already have your Oracle instance running, and have provisioned the needed resources. It also assumes that you have installed ownCloud with all of the prerequisites.

Configuring Oracle

Setting up the User Space for ownCloud

Step one, if it has not already been completed by your DBA, provision a user space on the Oracle instance for ownCloud. This can be done by logging in as a DBA and running the script below:

```
CREATE USER owncloud IDENTIFIED BY password;
ALTER USER owncloud DEFAULT TABLESPACE users TEMPORARY TABLESPACE temp QUOTA unlimited ON us
GRANT create session, create table, create procedure, create sequence, create trigger, creat
```

Substitute an actual password for `password`. Items like TableSpace, Quota etc. will be determined by your DBA.

Downloading and Installing the Oracle Instant Client

As our example system is Oracle Enterprise Linux, it is necessary to go to the Oracle site and download the Oracle Instant Client for your OS Distribution.

> ### *Note*
>
> Download the instant client and the instant client SDK and place them in a directory on the server, in this example they are RPM packages.

- Install the basic client from the RPM. Use the `rpm -ivh` command

- Install the SDK RPM package. Use the `rpm –ivh` command

At this point, the Oracle Instant client is installed on the ownCloud Host (in the home directory).

Install the OCI8 PHP Extension:

The next step is to compile and install the OCI8 PHP extension for connectivity to the Oracle Database.

- Create a folder for these bits on your server.
- Download the latest version of the extension from http://pecl.php.net/package/oci8.
- Unpack the OCI8 PHP extension and copy it over to the server.
- **There should be two things in the folder:**
    - `package.xml` file
    - `oci8-*.*.*` folder (folder will change based on version of the extension you downloaded).
- **Build the OCI8 module.**
    - Change (`cd`) to the folder where you have copied the downloaded and uncompressed OCI8 bits.
    - Run the following command (there will be a significant amount of output):

```
pecl build
```

    - Eventually the output will stop and ask for the *Oracle Home Directory*, just press enter.
- Change directory:

```
cd oci8-<version number>
```

- Type the following command:

```
./configure –with-oci8=instantcleint,/usr/lib/oracle/<version number>/client64/lib
```

- Again, there will be significant output
- Enter the following command to compile: `make`
- At this time there should be a folder called modules in the `oci8-<version_>` folder. Within this folder exists the `oci8.so` file.
- Copy this to the directory where the modules are stored in the PHP install. It depends on your distribution. This is the path for RHEL 6 and OEL 6:

```
cp oci8.so /usr/lib64/php/modules
```

- **Create an `.ini` file**
    - Navigate to the `php.d` directory: `cd /etc/php.d`
    - Edit a file called oci8.ini: `vi oci8.ini`
    - Make the file look as follows:

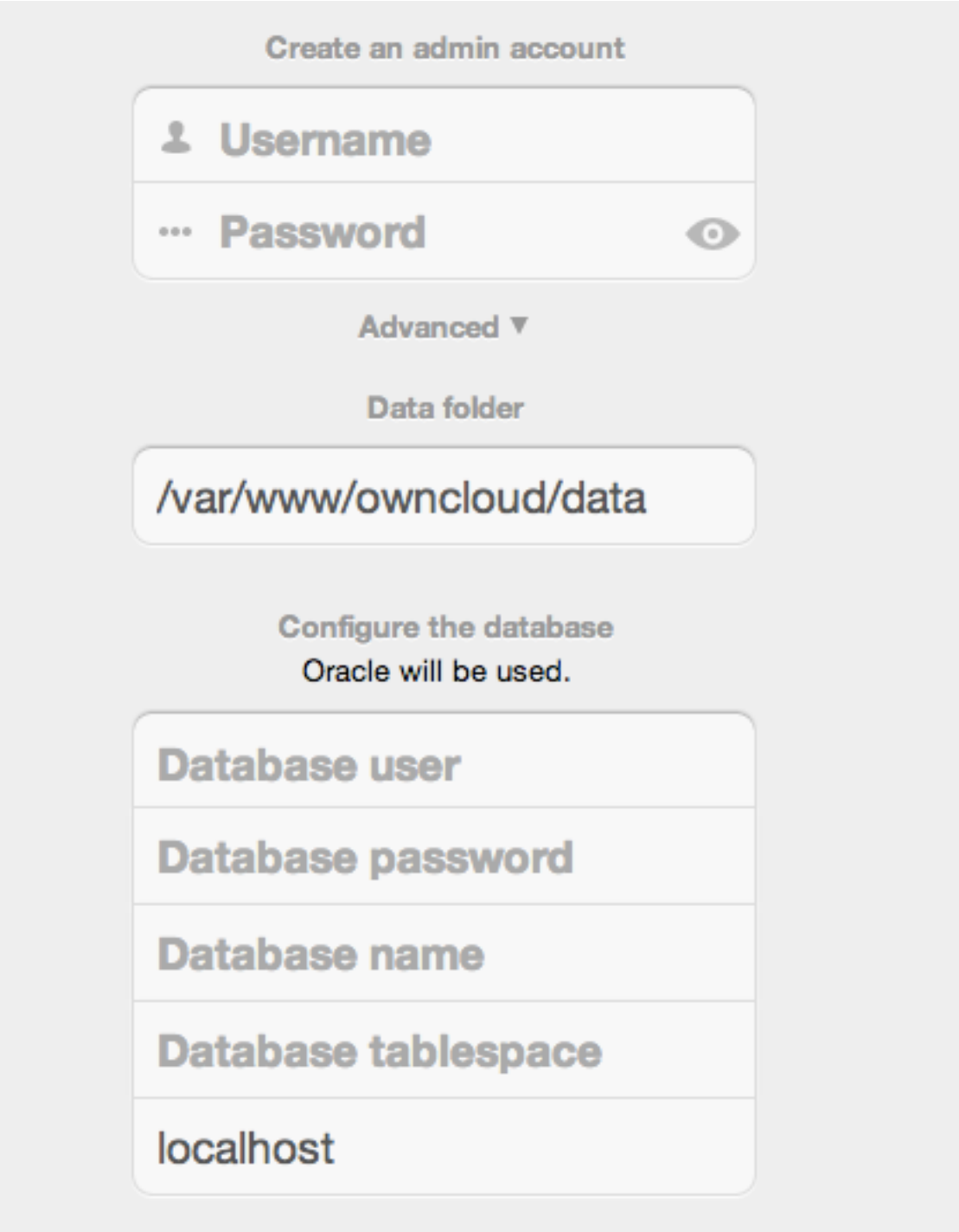
```
; Oracle Instant Client Shared Object
extension=oci8.so
```

    - Save the document

Configure ownCloud

The next step is to configure the ownCloud instance to point to the Oracle Database, again this document assumes that ownCloud has previously been installed.

Configuration Wizard



Database user

This is the user space created in step 2.1. In our Example this would be owncloud.

Database password

Again this is defined in the script from section 2.1 above, or pre-configured and provided to you by your DBA.

Database Name

Represents the database or the service that has been pre-configured on the TSN Listener on the Database Server. This should also be provided by the DBA. In this example, the default setup in the Oracle install was orcl (there is a TSN Listener entry for orcl on our database server).

This is not like setting up with MySQL or SQL Server, where a database based on the name you give is created. The oci8 code will call this specific service and it must be active on the TSN Listener on your Oracle Database server.

Database Table Space

Provided by the DBA. In this example the users table space (as is seen in the user creation script above), was used.

Configuration File

Assuming all of the steps have been followed to completion, the first run wizard should complete successfully, and an operating instance of ownCloud should appear.

The configuration file should look something like this:

```
<?php
$CONFIG = array (
'instanceid' => 'abcdefgh',
'passwordsalt' => '01234567890123456789',
'datadirectory' => '/var/data',
'dbtype' => 'oci',
'version' => '8.2.x.y',
'dbname' => 'orcl',
'dbhost' => '192.168.1.57',
'dbtableprefix' => 'oc_',
'dbuser' => 'owncloud1',
'dbpassword' => '********',
'installed' => true,
);
```

Useful SQL Commands

Is my Database Reachable?

On the machine where your Oracle database is installed, type:

```
sqlplus username
```

```
SQL> select * from v$version;

BANNER
--------------------------------------------------------------------------------
Oracle Database 11g Express Edition Release 11.2.0.2.0 - 64bit Production
PL/SQL Release 11.2.0.2.0 - Production
CORE  11.2.0.2.0      Production
TNS for Linux: Version 11.2.0.2.0 - Production
NLSRTL Version 11.2.0.2.0 - Production

SQL> exit
```

Show Database Users:

```
Oracle    : SELECT * FROM all_users;
```

Show available Databases:

```
Oracle    : SELECT name FROM v$database; (requires DBA privileges)
```

Show ownCloud Tables in Database:

```
Oracle    : SELECT table_name FROM user_tables;
```

Quit Database:

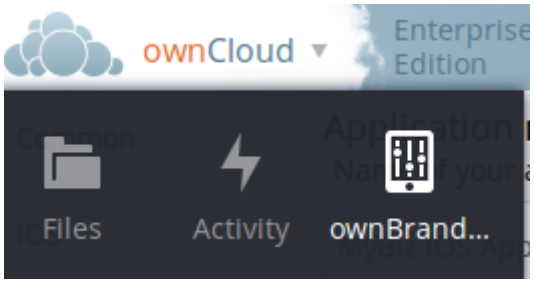
```
Oracle    : quit
```

Création de clients ownCloud aux couleurs de l'entreprise (Édition Entreprise)

Création d'applications clientes aux couleurs de l'entreprise (Édition Entreprise)

Overview

ownBrander is an ownCloud build service that is exclusive to Enterprise customers for creating branded Android and iOS ownCloud sync apps, and branded ownCloud desktop sync clients. You build your apps with the ownBrander app on your Customer.owncloud.com account, and within 24-48 hours the completed, customized apps are loaded into your account. You must supply your own artwork, and you'll find all the specifications and required elements in ownBrander.



Building a Branded Desktop Sync Client

See Building Branded ownCloud Clients (Enterprise Only) for instructions on building your own branded desktop sync client, and for setting up an automatic update service.

Your users may run both a branded and un-branded desktop sync client side-by-side. Both clients run independently of each other, and do not share account information or files.

Building a Branded iOS App

Building and distributing your branded iOS ownCloud app involves a large number of interdependent steps. The process is detailed in the Building Branded ownCloud Clients (Enterprise Only) manual. Follow these instructions exactly and in order, and you will have a nice branded iOS app that you can distribute to your users.

Building an Android App

Building and distributing your branded Android ownCloud app is fairly simple, and the process is detailed in Building Branded ownCloud Clients (Enterprise Only).

Dépôts de téléchargement de clients personnalisés

See *Dépôts de téléchargement personnalisés* to learn how test and configure custom download repository URLs for your branded clients.

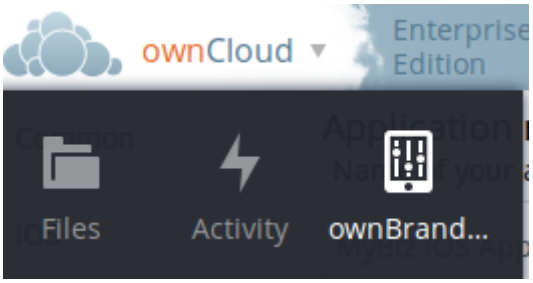
Personnalisation aux couleurs de l'entreprise (Édition Entreprise)

Thème personnalisé (Édition Entreprise)

Overview

ownBrander is an ownCloud build service that is exclusive to Enterprise edition customers for creating branded ownCloud clients and servers. You may brand your ownCloud server using ownBrander to easily build a custom theme, using your own logo and artwork. ownCloud has always been theme-able, but it was a manual process that required editing CSS and PHP files. Now Enterprise customers can use ownBrander, which provides an easy graphical wizard.

You need an Enterprise subscription, an account on Customer.owncloud.com, and the ownBrander app enabled on your account. When you complete the steps in the wizard the ownBrander service builds your new branded theme, and in 24-48 hours you'll see it in your account.



When you open the ownBrander app, go to the Web tab. You will see an introduction and the wizard, which starts with uploading your logo. You will need a number of images in specific sizes and formats, and the wizard tells you what you need. Example images are on the right, and you can click to enlarge them.

Note

If you see errors when you upload SVG files, such as "Incorrect extension.File type image/svg+xml is not correct", "This SVG is invalid", or "Error uploading file: Incorrect size", try opening the file in Inkscape then save as "Plain SVG" and upload your SVG image again.

The wizard has two sections. The first section contains all the required elements: logos and other artwork, colors, naming, and your enterprise URL. The Suggested section contains optional items such as additional logo placements and custom URLs.

When you are finished, click the **Generate Web Server** button. If you want to change anything, go ahead and change it and click the **Generate Web Server** button. This will override your previous version, if it has not been

created yet.In 24-48 hours you'll find your new branded theme in the **Web** folder in your Customer.owncloud.com account.

Inside the **Web** folder you'll find a **themes** folder. Copy this to your `owncloud/themes` directory. You may name your **themes** folder anything you want, for example `myBrandedTheme`. Then configure your ownCloud server to use your branded theme by entering it in your `config.php` file:

```
"theme" => "myBrandedTheme"
```

If anything goes wrong with your new theme, comment out this line to re-enable the default theme until you fix your branded theme. The branded theme follows the same file structure as the default theme, and you may further customize it by editing the source files.

Note

Always edit only your custom theme files. Never edit the default theme files.

Stockages externes (Édition Entreprise)

Options d'authentification (Édition Entreprise)

In ownCloud 9.0+, there are five authentication backends for external storage mounts:

- Username and password
- Log-in credentials, save in session
- Log-in credentials, save in database
- User entered, store in database
- Global credentials

The first two are common to all editions of ownCloud, and the last three are only in the Enterprise edition. These are available to:

- FTP
- ownCloud
- SFTP
- SMB/CIFS
- WebDAV
- Windows Network Drive

Username and password

This is the default; a login entered by the admin when the external mount is created. The login is stored in the database, which allows sharing, and background jobs, such as file scanning, to operate.

Log-in credentials, save in session

Credentials are only stored in the session and not captured in the database. Files cannot be shared, as credentials are not stored.

Log-in credentials, save in database

Credentials are stored in the database, and files can be shared.

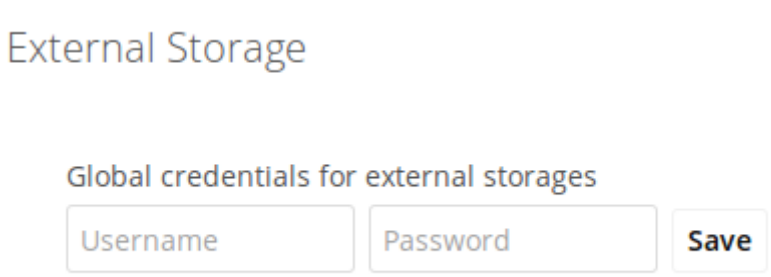
User entered, store in database

Users provide their own login credentials, rather than using admin-supplied credentials. User credentials are stored in the database, and files can be shared.

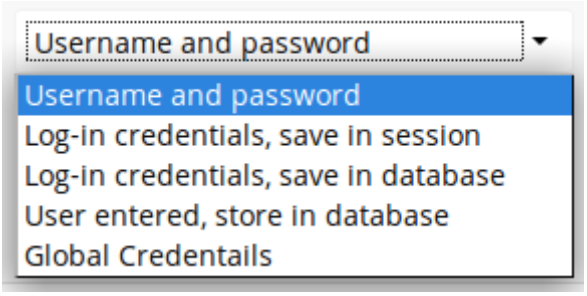
Global credentials

Re-usable credentials entered by the admin, files can be shared.

Global credentials are entered in a separate form.



Use the dropdown selector to choose the authentication backend when you create a new external mount.



Connecteur LDAP Home

The LDAP Home Connector App enables you to configure your ownCloud server to display your users' Windows home directories on their Files pages, just like any other folder. Typically, Windows home directories are stored on a network server in a root folder, such as Users, which then contains individual folders for each user.

You must already have the LDAP app enabled and a working LDAP/Active Directory configuration in ownCloud.

Next, configure the root Windows home directory to be mounted on your ownCloud server. Then use the LDAP Home Connector and LDAP app to connect it to ownCloud.

Mount Home Directory

Create an entry in /etc/fstab for the remote Windows root home directory mount. Store the credentials to access the home directory in a separate file, for example /etc/credentials, with the username and password on separate lines, like this:

```
username=winhomeuser
password=winhomepassword
```

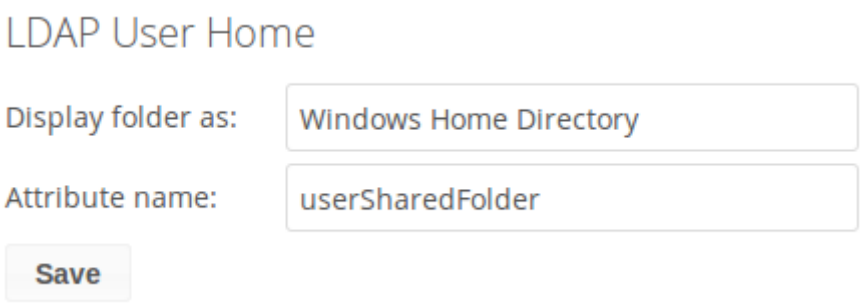
Then add a line like this to /etc/fstab, substituting your own server address and filenames:

```
//192.168.1.58/share /mnt/share cifs credentials=/etc/credentials,uid=33,gid=33
```

Configure the LDAP Home Connector

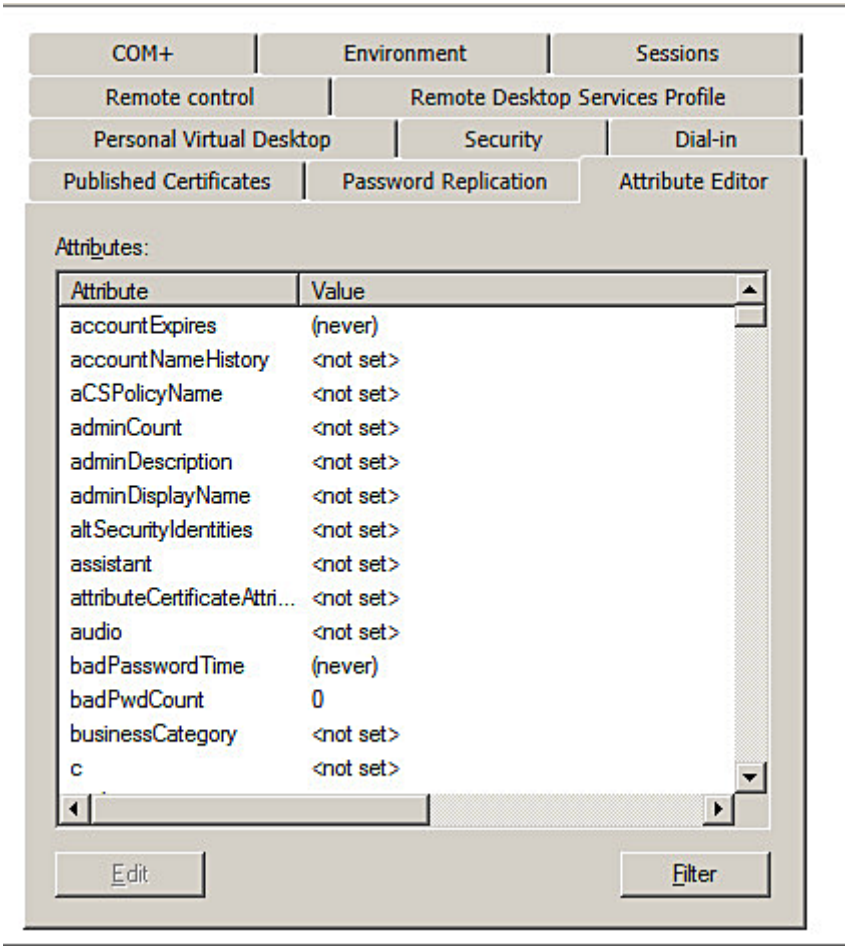
Enable the LDAP Home Connector app. Then go to the LDAP Home Connector form on your ownCloud admin page. In the **Display folder as:** field enter the name as you want it to appear on your users' File pages.

Then in the **Attribute name:** field enter the LDAP attribute name that will contain the home directory. Use any LDAP attribute that is not already in use, then save your changes.

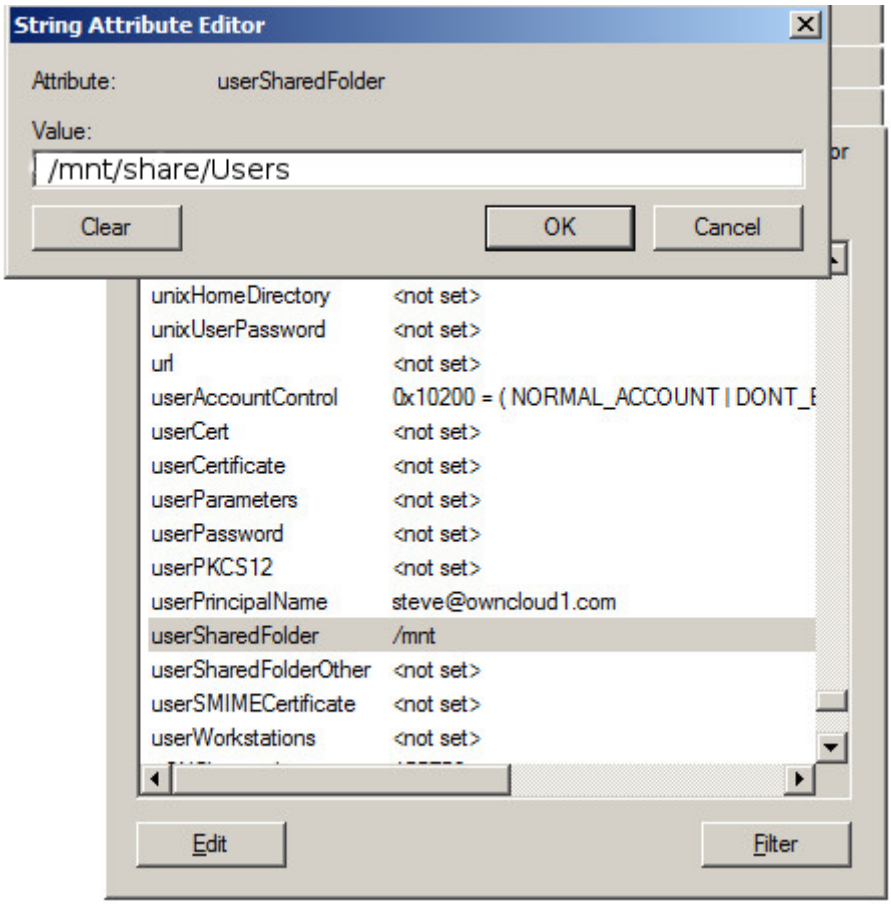


Configure the LDAP Server

In Active Directory, open the user profile. Scroll to the **Extensions** section and open the **Attribute Editor** tab



Scroll to the attribute being used (UserSharedFolder in this instance), and click **Edit**. Enter the users home directory.



Save your changes, and you are finished.

Configuration de l'intégration SharePoint

Native SharePoint support has been added to the ownCloud Enterprise edition as a secondary storage location for SharePoint 2007, 2010 and 2013. When this is enabled, users can access and sync all of their SharePoint content via ownCloud, whether in the desktop sync, mobile or Web interfaces. Updated files are bi-directionally synced automatically. SharePoint shares are created by the ownCloud admin, and optionally by any users who have SharePoint credentials.

The ownCloud SharePoint plugin uses SharePoint document lists as remote storage folders. ownCloud respects SharePoint access control lists (ACLs), so ownCloud sharing is intentionally disabled for SharePoint mountpoints. This is to preserve SharePoint ACLs and ensure content is properly accessed as per SharePoint rules.

The plugin uses the Simple Object Access Protocol (SOAP) and WebDAV for the uploads and downloads to talk to SharePoint servers. Your ownCloud server must have `php-soap` or `php5-soap` installed. Linux packages and ownCloud appliances will install `php5-soap` as a required dependency.

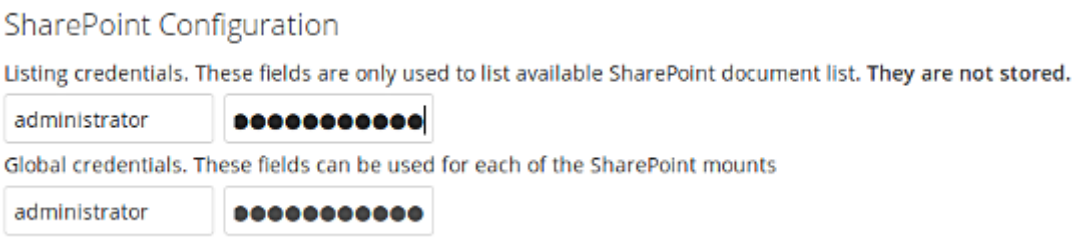
The supported authentication methods are:

- Basic Auth
- NTLM (Recommended)

Creating a Sharepoint Mount

Enable the Sharepoint app, and then enter the `Admin` panel to set up SharePoint connections in the `SharePoint Drive Configuration` section.

Enter your SharePoint Listing credentials. These credentials are not stored in the database, but are used only during plugin setup to list the Document Libraries available per SharePoint site.

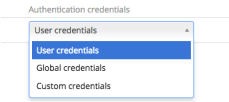


`Global credentials` is optional. If you fill in these fields, these credentials will be used on on all SharePoint mounts where you select: **Use global credentials** as the authentication credentials.

Enter your ownCloud mountpoint in the `Local Folder Name` column. This is the name of the folder that each user will see on the ownCloud filesystem. You may use an existing folder, or enter a name to create a new mount point

Select who will have access to this mountpoint, by default **All users**, or a user or a group.

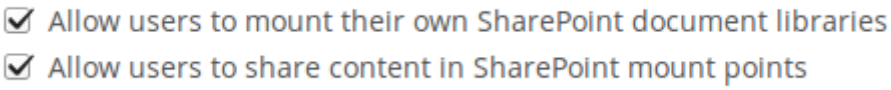
Enter your SharePoint server URL, then click the little refresh icon to the left of the `Document Library` field. If your credentials and URL are correct you'll get a dropdown list of available SharePoint libraries. Select the document library you want to mount.



Select which kind of Authentication credentials you want to use for this mountpoint. If you select **Custom credentials** you will have to enter the the credentials on this line. Otherwise, the global credentials or the user's own credentials will be used. Click Save, and you're done

Enabling Users

You may allow your users to create their own Sharepoint mounts on their Personal pages, and allow sharing on these mounts.



Note

Speed up load times by disabling file previews in `config.php`, because the previews are generated by downloading the remote files to a temp file. This means ownCloud will spend a lot of time creating previews for all of your SharePoint content. To disable file previews, add the following line to the ownCloud config file found in `/owncloud/config/config.php`:

```
'enable_previews' => false,
```

Troubleshooting

SharePoint unsharing is handled in the background via Cron. If you remove the sharing option from a Sharepoint mount, it will take a little time for the share to be removed, until the Cron job runs

Turn on Sharepoint app logging by modifying the following line in apps/sharepoint/lib/sharepoint.php to `TRUE`:

```
private static $enableLogs = TRUE;
```

Global mount points can't be accessed: You have to fill out your SharePoint credentials as User on the personal settings page, or in the popup menu. These credentials are used to mount all global mount points.

Personal mount points can't be accessed: You have to fill your SharePoint credentials as User on the personal settings page in case your personal mount point doesn't have its own credentials.

A user can't update the credentials: Verify that the correct credentials are configured, and the correct type, either global or custom.

Installing and Configuring the Windows Network Drive App

The Windows Network Drive app creates a control panel on your Admin page for seamless mounting of SMB/CIFS file shares on ownCloud servers.

Any Windows file share, and Samba servers on Linux and other Unix-type operating systems use the SMB/CIFS file-sharing protocol. The files and directories on the SMB/CIFS server will be visible on your Files page just like your other ownCloud files and folders. They are labeled with a little four-pane Windows-style icon, and the left pane of your Files page includes a Windows Network Drive filter. Figure 1 shows a new Windows Network Drive share marked with red warnings. These indicate that ownCloud cannot connect to the share because it requires the user to login, it is not available, or there is an error in the configuration.

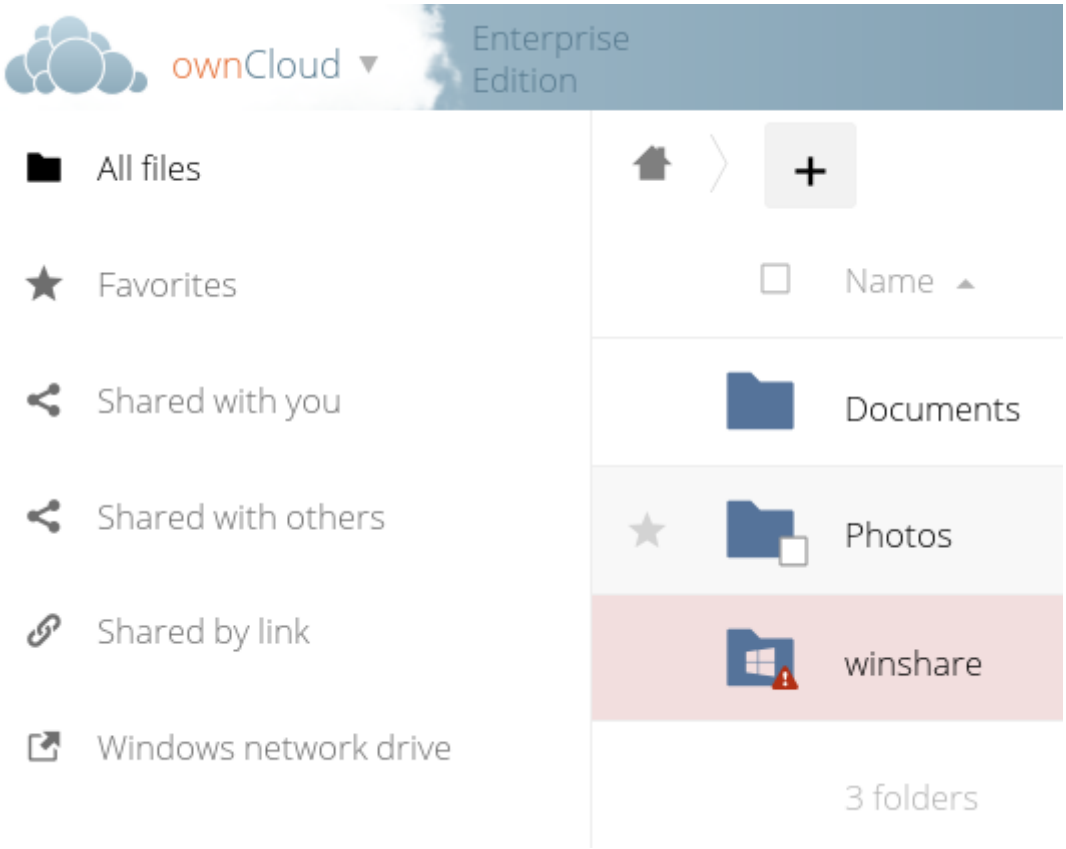


Figure 1: Windows Network Drive share on your Files page.

Files are synchronized bi-directionally, and you can create, upload, and delete files and folders. ownCloud server admins can create Windows Network Drive mounts, and optionally allow users to create their own personal Windows Network Drive mounts.

Depending on the authentication method, passwords for each mount are encrypted and stored in the ownCloud database, using a long random secret key stored in `config.php`, which allows ownCloud to access the shares when the users who own the mounts are not logged in. Or, passwords are not stored and available only for the current session, which adds security.

Installation

Enable the Windows Network Drive app on your ownCloud Apps page. Then there are a few dependencies to install.

You must install `php-smbclient`. This should be included in most Linux distributions. See eduardok/libsmbclient-php if your distribution does not include it; this provides source archives and instructions for installing binary packages.

You also need the Samba client installed on your Linux system. This is included in all Linux distributions; on Debian, Ubuntu, and other Debian derivatives this is `smbclient`. On SUSE, Red Hat, CentOS, and other Red Hat derivatives it is `samba-client`. You also need `which` and `stdbuf`, which should be included in most Linux distributions.

Creating a New Share

When you create a new WND share you need the login credentials for the share, the server address, the share name, and the folder you want to connect to.

1. First enter the ownCloud mountpoint for your new WND share. This must not be an existing folder.
2. Then select your authentication method; See *Options d'authentification (Édition Entreprise)* for complete information on the five available authentication methods.

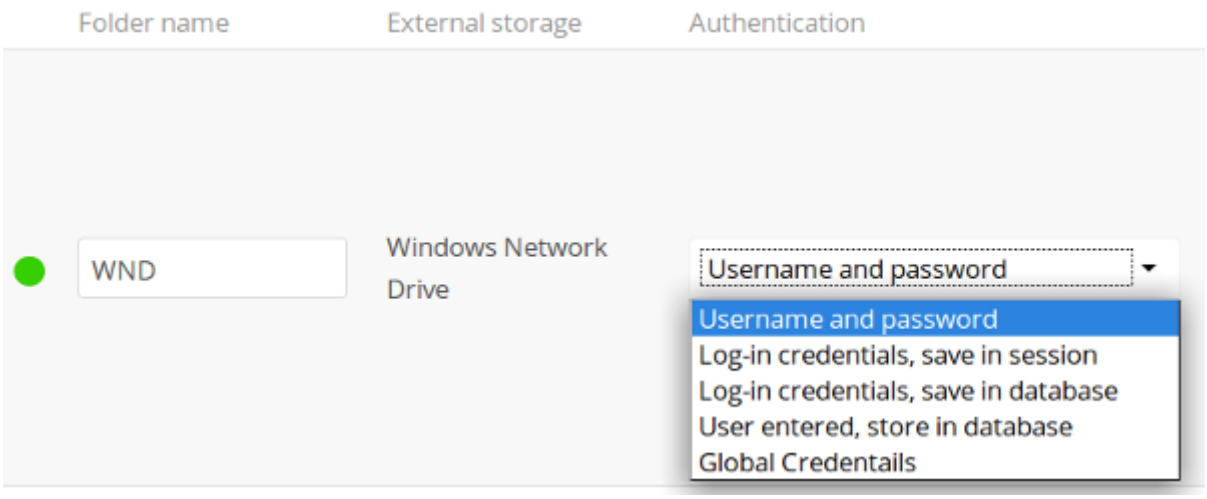


Figure 2: WND mountpoint and authorization credentials.

3. Enter the address of the server that contains the WND share.
4. The Windows share name.
5. The root folder of the share. This is the folder name, or the `$user` variable for user's home directories. Note that the LDAP `Internal Username Attribute` must be set to the `samaccountname` for either the share or the root to work, and the user's home directory needs to match the `samaccountname`. (See *Authentification utilisateur avec LDAP*.)
6. Login credentials.
7. Select users or groups with access to the share. The default is all users.
8. Click the gear icon for additional mount options. Note that encryption is enabled by default, while sharing is not. Sharing is not available for all authorization methods; see *Options d'authentification (Édition Entreprise)*.

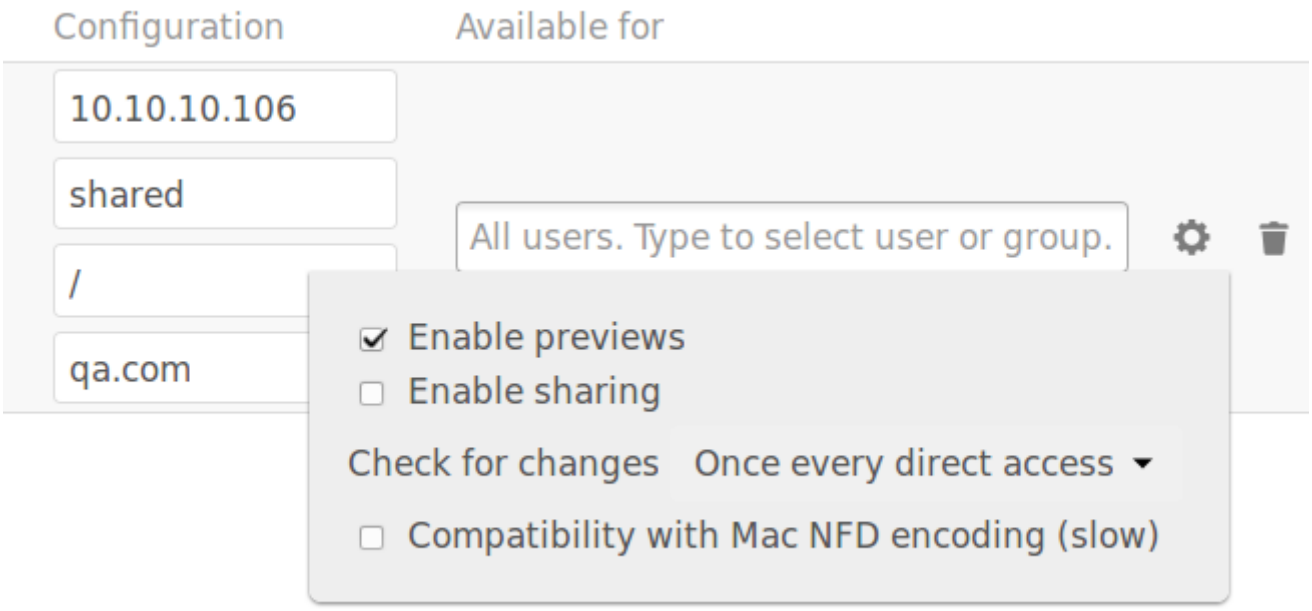


Figure 3: WND server, credentials, and additional mount options.

Your changes are saved automatically.

> ***Note***
>
> When you create a new mountpoint using Login credentials you must log out of ownCloud, and then log back in so you can access the share. You only have to do this the first time.

Personal WND Mounts

Users create their own personal WND mounts on their Personal pages. These are created the same way as Admin-created shares. Users have four options for login credentials:

- Username and password
- Log-in credentials, save in session
- Log-in credentials, save in database
- Global credentials

libsmclient Issues

If your Linux distribution ships with `libsmbclient 3.x`, which is included in the Samba client, you may need to set up the HOME variable in Apache to prevent a segmentation fault. If you have `libsmbclient 4.1.6` and higher it doesn't seem to be an issue, so you won't have to change your HOME variable.

To set up the HOME variable on Ubuntu, modify the `/etc/apache2/envvars` file:

```
unset HOME
export HOME=/var/www
```

In Red Hat/CentOS, modify the `/etc/sysconfig/httpd` file and add the following line to set the HOME variable in Apache:

```
export HOME=/usr/share/httpd
```

By default CentOS has activated SELinux, and the `httpd` process can not make outgoing network connections. This will cause problems with the `curl`, `ldap` and `samba` libraries. You'll need to get around this in order to make this work. First check the status:

```
getsebool -a | grep httpd
httpd_can_network_connect --> off
```

Then enable support for network connections:

```
setsebool -P httpd_can_network_connect 1
```

In openSUSE, modify the `/usr/sbin/start_apache2` file:

```
export HOME=/var/lib/apache2
```

Restart Apache, open your ownCloud Admin page and start creating SMB/CIFS mounts.

Windows Network Drive Listener

The SMB protocol supports registering for notifications of file changes on remote Windows SMB storage servers. Notifications are more efficient than polling for changes, as polling requires scanning the whole SMB storage. ownCloud supports SMB notifications with an `occ` command, `occ wnd:listen`.

Note

The notifier only works with remote storages on Windows servers. It does not work reliably with Linux servers due to technical limitations.

Your `smbclient` versions needs to be 4.x, as older versions do not support notifications.

The ownCloud server needs to know about changes of files on integrated storages so that the changed files will be synced to the ownCloud server, and to desktop sync clients. Files changed through the ownCloud Web interface or sync clients are automatically updated in the ownCloud filecache, but this is not possible when files are changed directly on remote SMB storage mounts.

To create a new SMB notification, start a listener on your ownCloud server with `occ wnd:listen`. The listener marks changed files, and a background job updates the file metadata.

Setup Notifications for an SMB Share

If you don't already have an SMB share, you must create one. Then start the listener with this command, like this example for Ubuntu Linux:

```
sudo -u www-data php occ wnd:listen <host> <share> <username> [password]
```

The `host` is your remote SMB server, which must be exactly the same as the server name in your WND configuration on your ownCloud Admin page.. `share` is the share name, and `username` and `password` are the login credentials for the share. By default there is no output. Enable verbosity to see the notifications:: `share` is the share name, and `username` and `password` are the login credentials for the share. By default there is no output. Enable verbosity to see the notifications:

```
$ sudo -u www-data php occ wnd:listen -v server share useraccount
Please enter the password to access the share:
File removed : Capirotes/New Text Document.txt
File modified : Capirotes
File added : Capirotes/New Text Document.txt
File modified : Capirotes
File renamed : old name : Capirotes/New Text Document.txt
File renamed : new name : Capirotes/New Document.txt
```

Enable increased verbosity to see debugging messages, including which storages are updated and timing:

```
$ sudo -u www-data php occ wnd:listen -vvv server share useraccount
Please enter the password to access the share:
notification received in 1471450242
File removed : Capirotes/New Document.txt
found 1 related storages from mount id 1
```

```
updated storage wnd::admin@server/share// from mount id 1 -> removed internal path : Capirot
found 1 related storages from mount id 3
updated storage wnd::administrador@server/share// from mount id 3 -> removed internal path :
found 1 related storages from mount id 2
```

See *Utilisation de la commande occ* for detailed help with `occ`.

One Listener for Many Shares

As the ownCloud server admin you can setup an SMB share for all of your users with a `$user` template variable in the root path. By using a ServiceUser you can listen to the common share path. The ServiceUser is any user with access to the share. You might create a special read-only user account to use in this case.

Example:

Share `/home` contains folders for every user, e.g. `/home/alice` and `/home/bob`. So the admin configures the Windows Network Drive external storage with these values:

- Folder name: home
- Storage Type: Windows Network Drive
- Authentication: Log-in credentials, save in database
- Configuration
  `host: "172.18.16.220", share: "home", remote subfolder: "$user", domain: ""`

Then starts the `wnd:listen` thread:

```
sudo -u www-data occ wnd:listen 172.18.16.220 home ServiceUser Password
```

Changes made by Bob or Alice made directly on the storage are now detected by the ownCloud server.

Configuration de S3 et OpenStack Swift Objects comme stockage primaire

In ownCloud Enterprise edition, you can configure S3 objects as primary storage. This replaces the default ownCloud `owncloud/data` directory. You may still need to keep the `owncloud/data` directory for these reasons:

- The ownCloud log file is saved in the data directory
- Legacy apps may not support using anything but the `owncloud/data` directory

You can move your logfile by changing its location in `config.php`. You may still need `owncloud/data` for backwards compatibility with some apps.

Implications

ownCloud in object store mode expects exclusive access to the object store container, because it only stores the binary data for each file. The metadata are kept in the local database for performance reasons.

The current implementation is incompatible with any app that uses direct file I/O and circumvents the ownCloud virtual filesystem. That includes Encryption and Gallery. Gallery stores thumbnails directly in the filesystem, and Encryption causes severe overhead because key files need to be fetched in addition to any requested file.

Configuration

Look in `config.sample.php` for a example configurations. Copy the relevant part to your `config.php` file. Any object store needs to implement `\\OCP\\Files\\ObjectStore\\IObjectStore` and can be passed parameters in the constructor with the `arguments` key:

```
'objectstore' => [
    'class' => 'Implementation\\Of\\OCP\\Files\\ObjectStore\\IObjectStore',
    'arguments' => [
        ...
    ],
],
```

Amazon S3

The S3 backend mounts a bucket of the Amazon S3 object store into the virtual filesystem. The class to be used is `OCA\ObjectStore\S3`:

```
'objectstore' => [
    'class' => 'OCA\ObjectStore\S3',
    'arguments' => [
        // replace with your bucket
        'bucket' => 'owncloud',
        'autocreate' => true,
        // uncomment to enable server side encryption
        //'serversideencryption' => 'AES256',
        'options' => [
            // version and region are required
            'version' => '2006-03-01',
            // change to your region
            'region'  => 'eu-central-1',
            'credentials' => [
                // replace key and secret with your credentials
                'key' => 'EJ39ITYZEUH5BGWDRUFY',
                'secret' => 'M5MrXTRjkyMaxXPe2FRXMTfTfbKEnZCu+7uRTVSj',
            ],
        ],
    ],
],
```

Ceph S3

The S3 backend can also be used to mount the bucket of a ceph object store via the s3 API into the virtual filesystem. The class to be used is `OCA\ObjectStore\S3`:

```
'objectstore' => [
    'class' => 'OCA\ObjectStore\S3',
    'arguments' => [
        // replace with your bucket
        'bucket' => 'owncloud',
        'autocreate' => true,
        'options' => [
            // version and region are required
            'version' => '2006-03-01',
            'region'  => '',
            // replace key, secret and bucket with your credentials
            'credentials' => [
                // replace key and secret with your credentials
                'key'    => 'EJ39ITYZEUH5BGWDRUFY',
                'secret' => 'M5MrXTRjkyMaxXPe2FRXMTfTfbKEnZCu+7uRTVSj',
            ],
            // replace the ceph endpoint with your rgw url
            'endpoint' => 'http://cephhost:8000/',
            // Use path style when talking to ceph
            'command.params' => [
                'PathStyle' => true,
            ],
        ],
    ],
],
```

S3 Multibucket Configuration

ownCloud 9.1+ has multibucket support for S3 object stores:

```
'objectstore_multibucket' => [
   'class' => 'OCA\ObjectStore\S3',
   'arguments' => [
       'autocreate' => true,
       'options' => [
           'version' => '2006-03-01',
           'region'  => 'eu-central-1',
           'credentials' => [
               'key' => 'EJ39ITYZEUH5BGWDRUFY',
               'secret' => 'M5MrXTRjkyMaxXPe2FRXMTfTfbKEnZCu+7uRTVSj',
           ],
       ],
   ],
],
```

OpenStack Swift

The Swift backend mounts a container on an OpenStack Object Storage server into the virtual filesystem. The class to be used is `\\OC\\Files\\ObjectStore\\Swift`:

```
'objectstore' => [
    'class' => 'OC\\Files\\ObjectStore\\Swift',
    'arguments' => [
        'username' => 'demo',
        'password' => 'password',
        'container' => 'owncloud',
        'autocreate' => true,
        'region' => 'RegionOne',
        'url' => 'http://devstack:5000/v2.0',
        'tenantName' => 'demo',
        'serviceName' => 'swift',
        // url Type, optional, public, internal or admin
        'urlType' => 'internal'
    ],
],
```

Intégration Jive

The Jive application allows Jive users to access files stored in Jive from a mobile device, tablet, or desktop client. Users have complete access through ownCloud Enterprise edition to upload, edit or download their files.

Jive can be configured as a data storage location for ownCloud, which means files saved in Jive appear in folders within ownCloud. Jive remains the system of record while ownCloud acts as a proxy, providing end-to-end file access for users at their desks and on the go.

Configuration

The Jive application is installed under the owncloud/apps directory on the server and enabled via the ownCloud admin screen. This app is only available for ownCloud EE v6 or higher. Go to the ownCloud admin screen section “Jive backend parameters” to configure the app to match your Jive server system parameters.

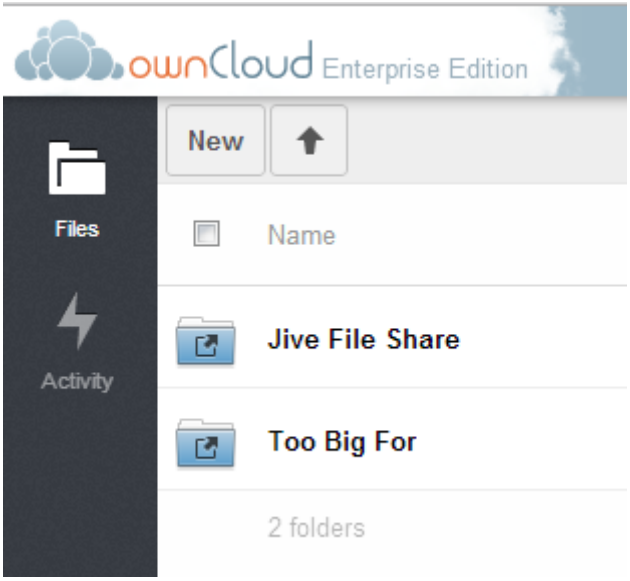
Parameter	Description	Values
Https	Verify the https server certificate. Certificate must be installed on the system.	Checkbox – enabled/disable
Authentication	Chose the Authentication mechanism to use against Jive	basic OR oAuth
Jive api url	URL string pointing to the Jive API	Example: https://mycompany.jiveon.com/api/core/v3/
Jive FS mount point	Folder where the Jive File share will be mounted	String value up to 10 characters max
Jive category filter	List of categories that files have to be shown	Jive categories list, or blank
Jive category separator	Separator for Jive catagories list	Comma by default or any single character
Jive tag filter	Tag to use for private stuff in jive	Jive tag or blank
Jive forbidden extensions	List of forbidden extensions These will not be allowed for upload or download with Jive.	Examples include: .exe,.zip

Jive forbidden extensions separator	Use this character to separate the list of extensions	Comma by default or any single character
Jive maximum upload filesize	Maximum file size allowed in MB. This includes upload and downloads.	Numeric value
Jive private folder	Folder name for private stuff in Jive	String value up to 250 characters max
Activate large file sharing for Jive	Enable the large file sharing subsystem. This allows storage of files that are too large for Jive to be stored on the ownCloud server and available via the ownCloud web, mobile and desktop interfaces.	Checkbox – enable/disable
Jive large file sharing FS mount point	Folder where the Jive large sharing File Share will be mounted	String value up to 10 characters max
Show groups of which you are a member	If this is not checked, the plugin will show all available groups for you matching the filter, even groups that you are not a member	Enable/disable

Use Cases

The ownCloud Jive plugin can be used in various ways to extend the access to the Jive content across multiple devices.

Web Client Use Cases



- Create a folder in the “Jive File Share” Web Client folder to create a new Jive Group.
    - Verify the Group is created in Jive.
- Create a new Group in Jive and upload a file to that Group.
    - Check the Web Client and download the file.
    - Verify that file is the same as the uploaded file.

- Upload a file in the “Too Big For” Jive folder, and create the link in a Jive document.
  - Verify that file link is in Jive.
  - Download the file via the link, and verify it is the same as the uploaded file.
- Upload a file to the private “My Jive” Web Client folder.
  - Check your Jive content and make sure the file has been uploaded.
  - Download the file and verify it is the same as the uploaded file.

Mobile Client Use Cases (iOs and Android)

Create a new folder in the Mobile Client to create a new Jive Group.

Upload a file in the Web Client folder, and see that file in the corresponding Jive Group.

Desktop Client Use Cases

Create a folder in the Desktop Client to create a new Jive Group.

Upload a file in the Desktop Client folder, and see that file in the corresponding Jive Group.

The ownCloud folder structure hierarchy matches the Jive Groups the user can access. Sub folders under the Jive Group folders that are created on the desktop will not sync to ownCloud or Jive because they will not match the Jive “Group” view. If a sub folder is created under the Jive Group desktop folder, the desktop client will display an error that this operation is not allowed. For example; if the folder structure is “JiveFileShare/GroupA”, any sub folder under GroupA will not be synced to ownCloud or Jive.

Configuring the Jive app

This section explains how each configuration parameter changes the behavior of the app.

Verify https certificate

If your Jive server is under https, it must provide a https certificate when a client connects to it. Curl (the client that ownCloud is using to connect to Jive) usually verify that certificate, but to do that you must somehow supply a CA cert so curl can verify against.

This feature is usually turn off to make the Jive app easier to use, because in this case curl won't verify the certificate, so you don't need to have installed the CA cert. However, turning this off could be a security issue: you could be connecting to a fake Jive server without notice.

If you want to turn on this feature, you must get the CA cert of the server (check “ http://curl.haxx.se/docs/sslcerts.html ” for more information about how you can get the file you need) and install it in your ownCloud server.

In order to know where you should install the CA cert, you can run

curl -v https://yourserver.com/

You should look the output for a line with the CA path:

- successfully set certificate verify locations:
- CAfile: none
- CApath: /etc/ssl/certs

That's the place where you should install the CA cert.

Once you have installed the CA cert, you should run again the same curl:

curl -v https://yourserver.com/

And look for:

- Server certificate:
- subject: *****

- start date: *******
- expire date: *****
- subjectAltName: *******
- issuer: ******
- SSL certificate verify ok.

If the SSL is verified correctly (“SSL certificate verify ok.”), you just need to activate the checkbox.

Curl usually comes installed with some CA certs by default, so all the previous steps might not be needed. Just check that curl can connect to your Jive server, and if so, activate this feature.

Authentication mechanism to use against Jive

To be able to access to Jive, the ownCloud plugin needs to use some kind of authentication. At this time, the plugin supports basic and oAuth authentication.

Basic authentication

In order to use basic authentication, you should take into account the following things:

- The credentials used to access to ownCloud must match the ones used to connect to Jive. This means that if you access to ownCloud with a user “PeterP” and password “PeterPassword”, the same user must exist in Jive with the same password. Otherwise, the user won't be able to access to Jive.
- If the credentials (typically the password) changes in one side, it must change in the other. You'll need to this manually.

The usage of basic authentication isn't recommended due to the following reasons:

- We need to store the password and be able to recover it. Although the password is stored encrypted, this is not strictly secure.
- Passwords are sent to the server in almost plain text. In fact it's a base64 encoded string of user and password, but that's all the security the authentication provides.

If you plan to use basic authentication, at least make sure you connect through HTTPS protocol and inside a local LAN if possible.

oAuth authentication

First of all, make sure Jive is prepared to support this authentication.

The usage of this authentication method solves the issue of having the same credentials in both ownCloud and Jive server. This means that the ownCloud user “PeterP” with password “PeterPassword” can access to the contents of the Jive user “John” with password “John007”. It's also possible that another ownCloud user “AliceK” access to the contents of the Jive user “John” too at the same time.

Keep in mind that this isn't insecure: any ownCloud user that wants to access to John's Jive content (following this little example) MUST know his Jive password.

If this authentication method is set, we don't store passwords BUT we still need to store some other things. These things are stored in plain text.

These are the steps to make it work (if your Jive server support this authentication):

1. Activate the oAuth authentication in the ownCloud admin settings (just the admin can do this)
2. Go to the ownCloud web interface, in the files app. A popup will appear.
3. Click on the link that appear in the popup
4. You'll get redirected to a Jive page asking for your Jive credentials. If this is not the case, it's recommended to clean the browser cache and start again (to point 2) because you might be accessing to Jive with another user.
5. After entering your Jive credentials, you get redirected a page with an activation code. If you entered the wrong credentials, you might not get redirected to that page. If this is the case click in the link again in the ownCloud popup (point 3) which will redirect you to the activation code page.

6. Copy the activation code into the ownCloud popup, and click in the “send code” button. If there is no error, you're done.

WARNING:

Not all the oAuth flows are covered by the plugin. The expiration of the access token is handled automatically by the plugin, so it will request a new access token if needed. HOWEVER, the expiration of the refresh token isn't covered, so the plugin will likely stop working for that user (we won't be able to get more access tokens)

[Ask for info to know how to solve this issue?]

It's very important that the user access to ownCloud through the web interface first, so the user goes through the oAuth flow for the first time (as described with the steps above) to get an access token. Otherwise, the plugin won't get an access token and the user won't be able to get the files from Jive.

Jive API URL

You'll need to enter the full URL of the Jive API. This includes the protocol (HTTP or HTTPS) and the port (if any).

An example of API URL could be: “ https://myjiveserver.com/api/core/v3/ ”

Notice the following things:

- You must specify a protocol that is understandable by curl. Under normal circumstances, the protocol is limited to HTTP or HTTPS.
- If your server is under a port different than the 80, you'll need to specify it. Take “ https://jserver.prv:9999/api/core/v3/ ” as an example
- If your server isn't under the root URL, you can also specify the correct path: “ https://myserver.prv:8888/path/to/jive/api/core/v3/ ”
- The API URL should end with “/api/core/v3/” (be careful with the slash at the end)

Filters

The Jive plugin comes with a set of filters that the admin can set to filter the content the users can access through ownCloud. The drawback of using filters is that there isn't any performance gain because the filtering is mainly done in the ownCloud side, and even can degrade performance in some cases. We'll explain the filters one by one, and tell you what consequences have each one.

Category filter and separator

You can filter files using one or several categories. This filter applies only to groups and files inside those groups. Your private files won't be affected by this filter.

In order to set this filter, you can provide a list of categories, all in one line. In order to separate the different categories, you must use the separator set in the “category separator” text box.

Jive category filter : syncWithMe,sync,syncMe

Jive category separator : ,

You can also achieve the same behavior with:

Jive category filter : syncWithMe#sync#syncMe

Jive category separator : #

The plugin will show all groups which have ALL those categories set. If there is a group with any of the categories missing, that group won't be shown. Anyway, you should only need to set one category.

It's important to notice that, although you can set only one category or leave the text box empty, the category separator MUST always be set. In fact, you shouldn't need to change the default value of the category separator.

Files shown inside those groups will be also affected by this filter. This means that all the files shown inside those groups must have all the categories too.

Files uploaded through ownCloud to those groups will have all the categories set in Jive automatically. If you want to add more categories to those files, you'll need to do it manually through Jive.

The usage of the category filter can degrade the performance a lot. We need to make extra calls to Jive to get the categories for each group, one extra call per group returned by Jive in the first place. There is also the limitation of not having more than 25 categories set per group. Use this filter with extreme caution.

You can “disable” this filter just by setting the category filter empty. This will prevent the extra call from being made, and will show all available groups.

Tag filter

This filter works only for private files. Files inside groups won't be affected by this filter.

You can only set one tag for the files that will be shown in ownCloud. Make sure you set one of the tags from Jive as they're shown there. It's highly recommended to use only lowercase letters to set the tag to prevent possible issues when the tag is set in Jive.

The usage of this filter won't alter significantly the performance

It's important to notice that the filter will be applied to all users. Users won't be able to set their own tag to sync their own files.

This filter can also be “disabled” by setting the filter empty.

Forbidden extensions filter and separator

This filter is set the same way as the category filter: you provide a list of extensions that are separated by the char set in the separator text box.

Jive forbidden extensions: .exe,.zip,.tar.gz

Jive forbidden extensions separator : ,

You can also achieve the same behavior with:

Jive forbidden extensions: .exe#.zip#.tar.gz

Jive forbidden extensions separator: #

Keep in mind that the filter is performed against the end of the filename, that's why the “.” is important. If

you set “exe” as a forbidden extension, a file named “texe” or “f1.lexe” will be affected by this filter.

You must also take into account that, by using only the filename, we avoid to download the file, so the performance isn't significantly degraded. On the other hand, we cannot verify that a “.png” file is what it claims to be.

This filter works for any file, and for uploads and downloads through ownCloud. This means that you won't be able to upload a file with any of those extensions from onwCloud and the Jive files which have those extensions won't be shown (and consequently they won't be downloaded). Of course, you can still upload the files from Jive (if Jive allows it) and have them there.

Maximum upload file size

This filter allows you to limit the size of the files that will go through ownCloud. All files uploads and downloads will be affected by this filter. You won't be able to upload files bigger than the file size limit and the Jive files bigger than the limit won't be shown in ownCloud (and consequently they won't be downloaded)

Under normal circumstances, you want to match the limit with the one Jive has. This way you can notify errors regarding the file size faster because the files won't reach the Jive server, and at the same time you allow the users to upload up to the maximum limit that Jive allows. (Note: we can't know this limit from ownCloud, so we can't provide a sensitive default value, plus the value can change among Jive instances. You might need to adjust the value manually).

You can also set the limit to a lower value than what it's in Jive, so only small files will be delivered from ownCloud.

Show groups of which you are member

Under normal circumstances, you can see all available groups in Jive, including open, member-only and private groups, only secrets groups are outside. Even if you're not a member of those groups, you can still see their contents.

For small Jive installations (less than 100 available groups per user) this is usually enough, and it has an acceptable performance. However, for larger installations, with more than 500 groups available per user, the performance is degraded a lot.

For these larger installations, this checkbox comes in handy.

Again, under normal circumstances, it's common that a user is member of just a few groups (let's say less than 25) even if there are thousand of groups available that the user can see. It usually makes sense to show the contents of only those 25 groups, not every group available.

By activating this checkbox, the user will see only those 25 groups instead of all available groups. This will increase the performance a lot, specially for larger installations, as long as each user isn't member of too many groups. Anyway, if there are user who are member of too many groups, the performance will still be degraded.

FS mount points

This Jive plugin mounts one (or two) virtual filesystems on the normal one in a transparent way.

From a user point of view, these virtual filesystems appear as new folders inside the root one.

From the settings page, you can change the mount points names. The folders will change accordingly.

Jive FS mount point

The name of the folder that will hold the Jive virtual FS. The name shouldn't collide with any existing name in the root folder to prevent possible issues. The virtual FS will be mounted inside the root folder of the ownCloud FS.

As said, the contents of the folder will be the groups that the user can access through ownCloud (recheck the “filters” section).

Jive private folder

The folder where your private Jive files will be stored. The name of the folder will be the same for all users, although the contents will likely differ.

This private folder will be inside the Jive mount point, as if it were another group.

Files inside this folder will be only visible to you, but they will be stored in Jive. They won't be visible neither for ownCloud users nor Jive users.

In order to prevent collisions with other groups, the folder name might be changed automatically by adding “(private)” to the end of the folder name if it's needed .

Large file sharing subsystem

The large file sharing allow you to share files over the Jive limits (typically size limits). You can enable or disable this subsystem by checking or un-checking the checkbox, and provide the corresponding mount point. Use a non-existent folder name to prevent issues.

Files inside that folder will be stored inside the ownCloud server. However those files can be shared by link to Jive.

The process is like the following:

1. Upload a file (or folder) inside the large file sharing folder (by default named as “Too Big For”)
2. Once the file is uploaded, click in the “share” action, and then click in the “Share link” checkbox
3. By default the share link will expire after 1 week. You can change the value and / or protect the link by password
4. Click the “Submit to Jive” button (the name can be changed depending on the actual Jive folder name)
5. A new browser tab should appear with the Jive draft ready to be edited (you might need to enter your Jive credentials first). The draft will have some predefined text, but you can edit it to your needs. Once you publish the document, it's done.

Notifications

This Jive plugin runs a connectivity check between ownCloud and Jive whenever the web page is loaded. This check allows you to know some potential issues between the ownCloud – Jive connection.

When a potential issue is detected, a notification will be shown, so you'll know what's happening.

You can control the time the notification is shown in the “notification time for the connectivity check” configuration. The time is in seconds.

Gestion des utilisateurs (Édition Entreprise)

Intégration Shibboleth (Édition Entreprise)

Introduction

The ownCloud Shibboleth user backend application integrates ownCloud with a Shibboleth Service Provider (SP) and allows operations in federated and single-sign-on (SSO) infrastructures. Setting up Shibboleth has two big steps:

1. Enable and configure the Apache Shibboleth module.
2. Enable and configure the ownCloud Shibboleth app.

The Apache Shibboleth module

Currently supported installations are based on the native Apache integration. The individual configuration of the service provider is highly dependent on the operating system, as well as on the integration with the Identity Providers (IdP), and require case-by-case analysis and installation.

A good starting point for the service provider installation can be found in the official Shibboleth Wiki.

A successful installation and configuration will populate Apache environment variables with at least a unique user id which is then used by the ownCloud Shibboleth app to login a user.

See the documentation Wiki for more configuration examples.

Apache Configuration

This is an example configuration as installed and operated on a Linux server running the Apache 2.4 Web server. These configurations are highly operating system specific and require a high degree of customization.

The ownCloud instance itself is installed in `/var/www/owncloud/`. The following aliases are defined in an Apache virtual host directive:

```
# non-Shibboleth access
Alias /owncloud /var/www/owncloud/
# for Shibboleth access
Alias /oc-shib /var/www/owncloud/
```

Further Shibboleth specific configuration as defined in `/etc/apache2/conf.d/shib.conf`:

```
#
# Load the Shibboleth module.
#
LoadModule mod_shib /usr/lib64/shibboleth/mod_shib_24.so

#
# Ensures handler will be accessible.
#
<Location /Shibboleth.sso>
  AuthType None
  Require all granted
</Location>

#
# Configure the module for content.
```

```
#

#
# Besides the exceptions below, this location is now under control of
# Shibboleth
#
<Location /oc-shib>
  AuthType shibboleth
  ShibRequireSession On
  ShibUseHeaders Off
  ShibExportAssertion On
  require valid-user
</Location>

#
# Shibboleth is disabled for the following location to allow non
# shibboleth webdav access
#
<Location ~ "/oc-shib/remote.php/nonshib-webdav">
  AuthType None
  Require all granted
</Location>

#
# Shibboleth is disabled for the following location to allow public link
# sharing
#
<Location ~ \
"/oc-shib/(status.php$\
|index.php/s/\
|public.php\
|cron.php$\
|core/img/\
|index.php/apps/files_sharing/ajax/publicpreview.php$\
|index.php/apps/files/ajax/upload.php$\
|apps/files/templates/fileexists.html$\
|index.php/apps/files/ajax/mimeicon.php$\
|index.php/apps/files_sharing/ajax/list.php$\
|themes/\
|index.php/apps/files_pdfviewer/\
|apps/files_pdfviewer/)">
  AuthType None
  Require all granted
</Location>

#
# Shibboleth is disabled for the following location to allow public gallery
# sharing
#
<Location ~ \
"/oc-shib/(index.php/apps/gallery/s/\
|index.php/apps/gallery/slideshow$\
|index.php/apps/gallery/.*\.public)">
  AuthType None
  Require all granted
</Location>

#
# Shibboleth is disabled for the following location to allow public link
# sharing
```

```
#
<Location ~ "/oc-shib/.*\.css">
  AuthType None
  Require all granted
</Location>

#
# Shibboleth is disabled for the following location to allow public link
# sharing
#
<Location ~ "/oc-shib/.*\.js">
  AuthType None
  Require all granted
</Location>

#
# Shibboleth is disabled for the following location to allow public link
# sharing
#
<Location ~ "/oc-shib/.*\.woff">
  AuthType None
  Require all granted
</Location>
```

Depending on the ownCloud Shibboleth app mode, you may need to revisit this configuration.

The ownCloud Shibboleth App

After enabling the Shibboleth app on your Apps page, you need to choose the app mode and map the necessary Shibboleth environment variables to ownCloud user attributes on your Admin page.

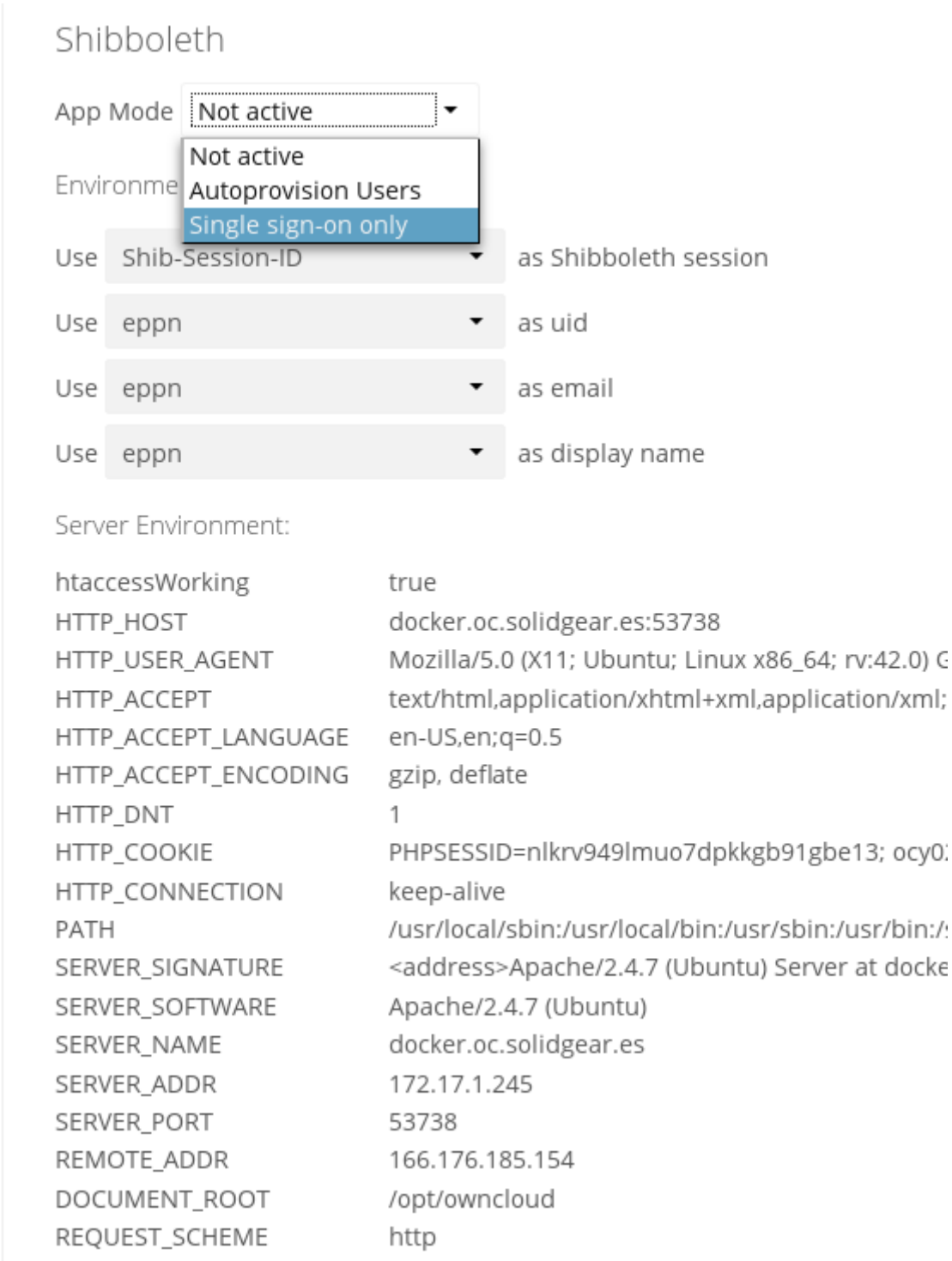


figure 1: Enabling Shibboleth on the ownCloud Admin page

Choosing the App Mode

After enabling the app it will be in **Not active** mode, which ignores a Shibboleth session and allows you to login as an administrator and inspect the currently available Apache environment variables. Use this mode to set up the environment mapping for the other modes, and in case you locked yourself out of the system. You can also change the app mode and environment mappings by using the `occ` command, like this example on Ubuntu Linux:

```
$ sudo -u www-data php occ shibboleth:mode notactive
$ sudo -u www-data php occ shibboleth:mapping --uid login
```

In **Single sign-on only** mode the app checks if the environment variable for the Shibboleth session, by default **Shib-Session-Id**, is set. If that is the case it will take the value of the environment variable as the `uid`, by default

eppn, and check if a user is known by that `uid`. In effect, this allows another user backend, eg. the LDAP app, to provide the `displayname`, `email` and `avatar`.

> ### *Note*
>
> As an example the IdP can send the **sAMAccountName** which the Apache Shibboleth module writes to a custom Apache environment variable called `login`. The ownCloud Shibboleth app reads that `login` environment variable and tries to find an LDAP user with that `uid`. For this to work the LDAP backend also needs to be configured to use the **sAMAccountName** as the **Internal Username Attribute** in the *LDAP expert settings*.

> ### *Note*
>
> In many scenarios Shibboleth is not intended to hide the user's password from the service provider, but only to implement SSO. If that is the case it is sufficient to protect the ownCloud base url with Shibboleth. This will send Web users to the IdP but allow desktop and mobile clients to continue using username and password, preventing popups due to an expired Shibboleth session lifetime.

In **Autoprovision Users** mode the app will not ask another user backend, but instead provision users on the fly by reading the two additional environment variables for display name and email address.

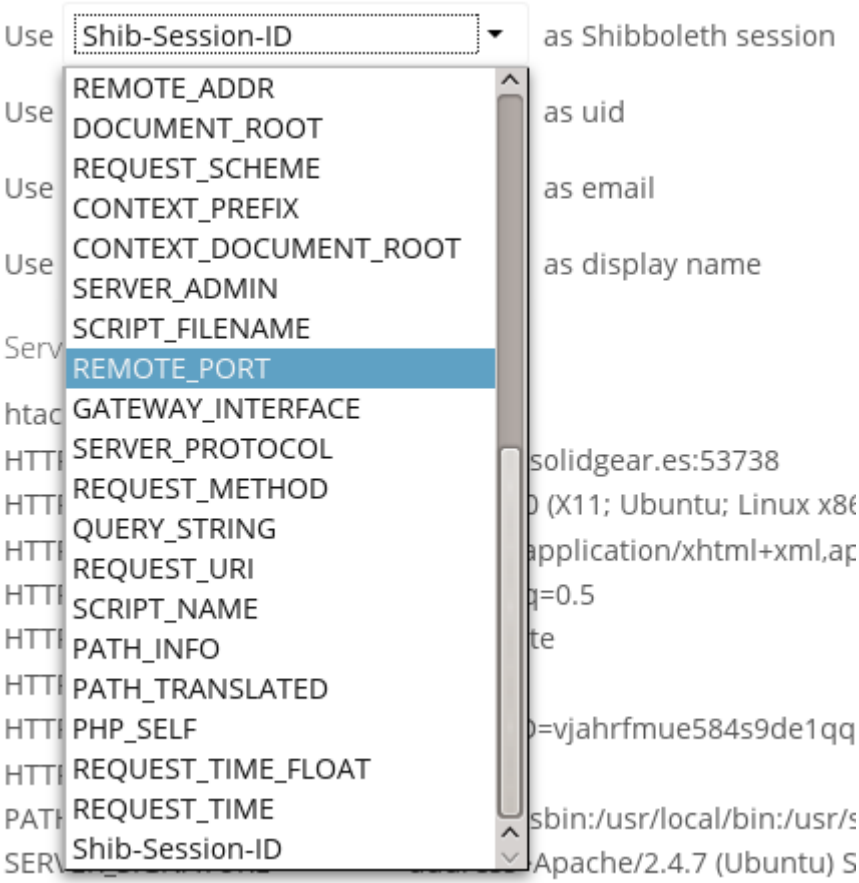


figure 2: Mapping Shibboleth environment configuration variables to ownCloud user attributes

In ownCloud 8.1 the Shibboleth environment variable mapping was stored in `apps/user_shibboleth/config.php`. This file was overwritten on upgrades, preventing a seamless upgrade procedure. In ownCloud 8.2+ the variables are stored in the ownCloud database, making Shibboleth automatically upgradeable.

Shibboleth with Desktop and Mobile Clients

The ownCloud Desktop Client can interact with an ownCloud instance running inside a Shibboleth Service Provider by using built-in browser components for authentication against the IdP.

The regular ownCloud Android and iOS mobile apps do not work with Shibboleth. However, customers who create *branded mobile apps with ownBrander* have the option to enable SAML authentication in ownBrander.

Enterprise customers also have the option to request a regular ownCloud mobile client built to use Shibboleth from their ownCloud account representatives.

The ownCloud desktop sync client and mobile apps store users' logins, so your users only need to enter their logins the first time they set up their accounts.

Note

The ownCloud clients may use only a single Shibboleth login per ownCloud server; multi-account is not supported with Shibboleth.

These screenshots show what the user sees at account setup. Figure 1 shows a test Shibboleth login screen from Testshib.org on the ownCloud desktop sync client.

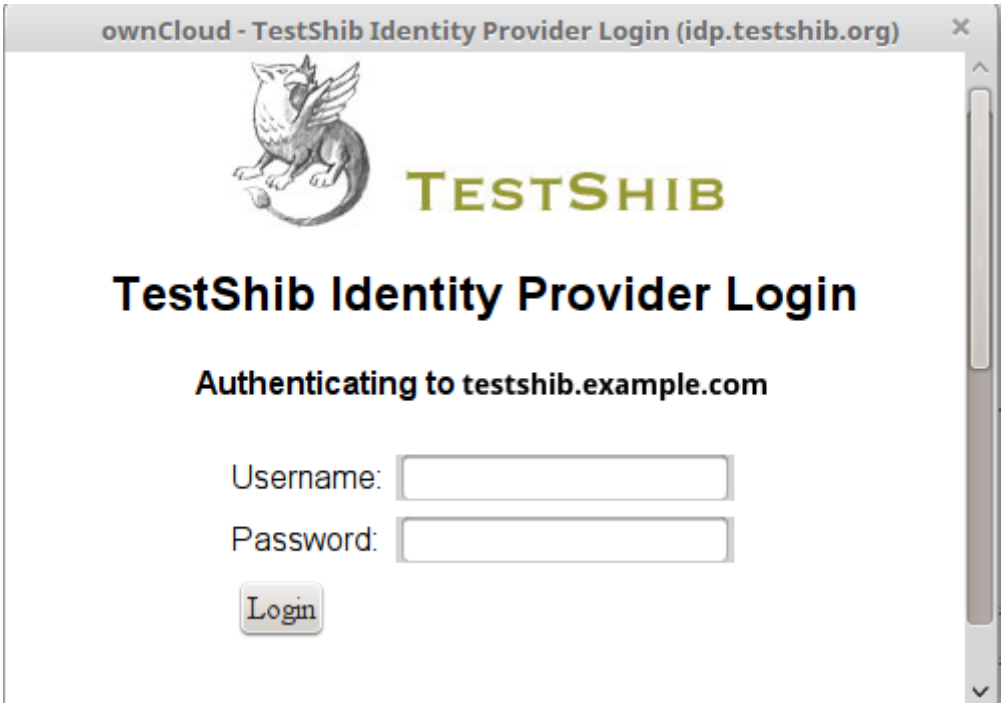


figure 3: First login screen

Then after going through the setup wizard, the desktop sync client displays the server and login information just like it does for any other ownCloud server connections.

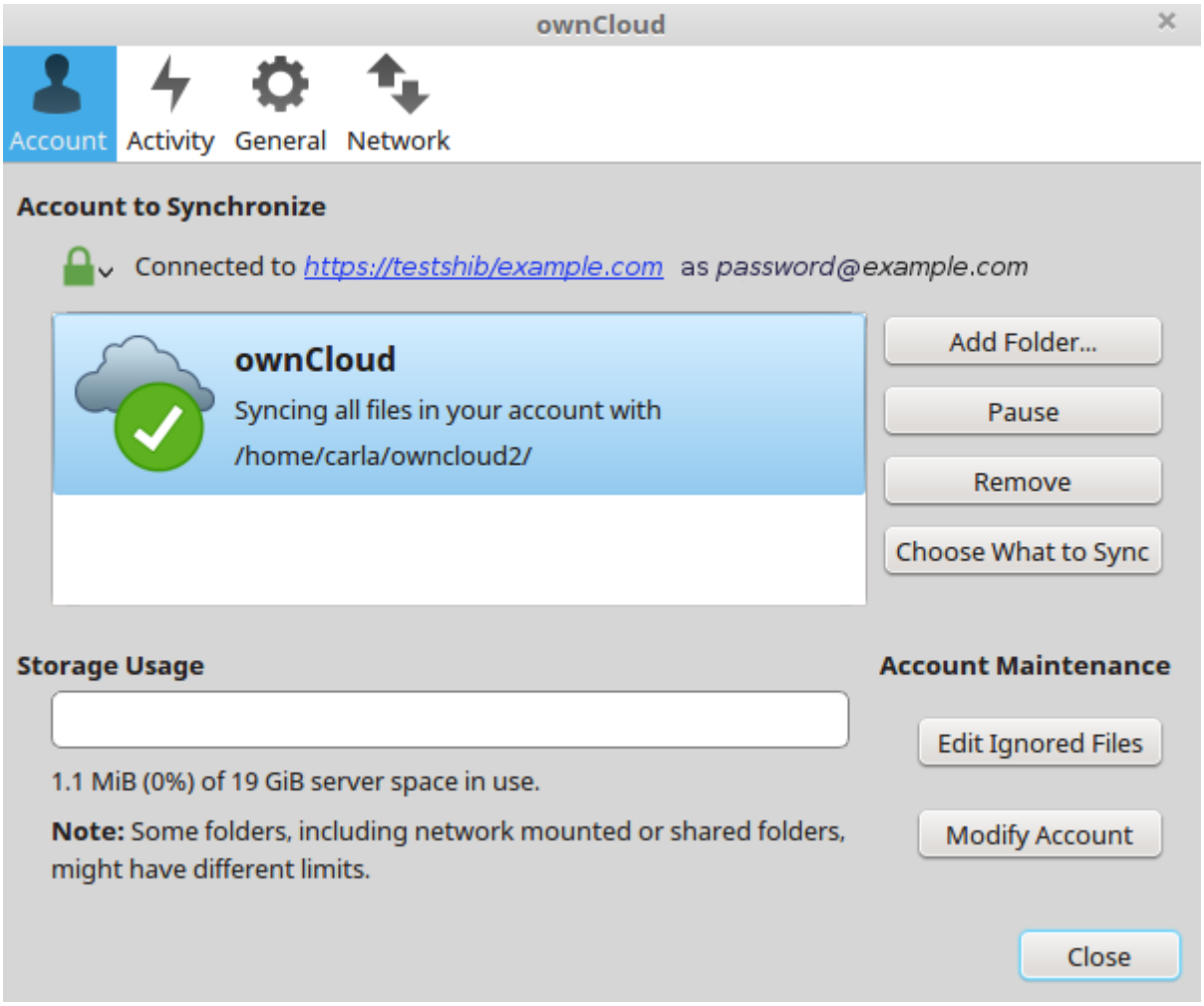


figure 4: ownCloud client displays server information

To your users, it doesn't look or behave differently on the desktop sync client, Android app, or iOS app from an ordinary ownCloud account setup. The only difference is the initial setup screen where they enter their account login.

WebDAV Support

Users of standard WebDAV clients can use an alternative WebDAV Url, for example `https://cloud.example.com/remote.php/nonshib-webdav/` to log in with their username and password. The password is generated on the Personal settings page.

Note

In **Single sign-on only** mode the alternative WebDAV Url feature will not work, as we have no way to store the WebDAV password. Instead the normal WebDAV endpoint can be omittet from the Shibboleth authentication, allowing WebDAV clients to use normal username and password based authentication. That includes the desktop and mobile clients.

For provisioning purpose an OCS API has been added to revoke a generated password for a user:

Syntax: `/v1/cloud/users/{userid}/non_shib_password`

- HTTP method: DELETE

Status codes:

- 100 - successful
- 998 - user unknown

Example:

```
$ curl -X DELETE "https://cloud.example.com/ocs/v1.php/cloud/users/myself@testshib.org/non_s
<?xml version="1.0"?>
<ocs>
 <meta>
  <status>ok</status>
  <statuscode>100</statuscode>
  <message/>
 </meta>
 <data/>
</ocs>
```

Known Limitations

Encryption

File encryption can only be used together with Shibboleth when the *master key-based encryption* is used because the per- user encryption requires the user's password to unlock the private encryption key. Due to the nature of Shibboleth the user's password is not known to the service provider.

Other Login Mechanisms

You can allow other login mechanisms (e.g. LDAP or ownCloud native) by creating a second Apache virtual host configuration. This second location is not protected by Shibboleth, and you can use your other ownCloud login mechanisms.

Session Timeout

Session timeout on Shibboleth is controlled by the IdP. It is not possible to have a session length longer than the length controlled by the IdP. In extreme cases this could result in re-login on mobile clients and desktop clients every hour.

UID Considerations and Windows Network Drive compatability

When using `user_shibboleth` in **Single sign-on only** mode, together with `user_ldap`, both apps need to resolve to the same `uid`. `user_shibboleth` will do the authentication, and `user_ldap` will provide user details such as `email` and `displayname`. In the case of Active Directory, multiple attributes can be used as the `uid`. But they all have different implications to take into account:

sAMAccountName

- *Example:* jfd
- *Uniqueness:* Domain local, might change e.g. marriage
- *Other implications:* Works with `windows_network_drive` app

userPrincipalName

- *Example:* jfd@owncloud.com
- *Uniqueness:* Forest local, might change on eg. marriage
- *Other implications:* TODO check WND compatability

objectSid

- *Example:* S-1-5-21-2611707862-2219215769-354220275-1137
- *Uniqueness:* Domain local, changes when the user is moved to a new domain
- *Other implications:* Incompatible with `windows_network_drive` app

sIDHistory

- *Example:* Multi-value
- *Uniqueness:* Contains previous objectSIDs
- *Other implications:* Incompatible with `windows_network_drive` app

objectGUID

- *Example:* 47AB881D-0655-414D-982F-02998C905A28
- *Uniqueness:* Globally unique
- *Other implications:* Incompatible with `windows_network_drive` app

Keep in mind that ownCloud will derive the home folder from the `uid`, unless a home folder naming rule is in place. The only truly stable attribute is the `objectGUID`, so that should be used. If not for the `uid` then at least as the home folder naming rule. The tradeoff here is that if you want to use `windows_network_drive` you are bound to the `sAMAccountName`, as that is used as the login.

Also be aware that using `user_shibboleth` in **Autoprovision Users** mode will not allow you to use SSO for additional `user_ldap` users, because `uid` collisions will be detected by `user_ldap`.

Gestion de fichiers d'entreprise (Édition Entreprise)

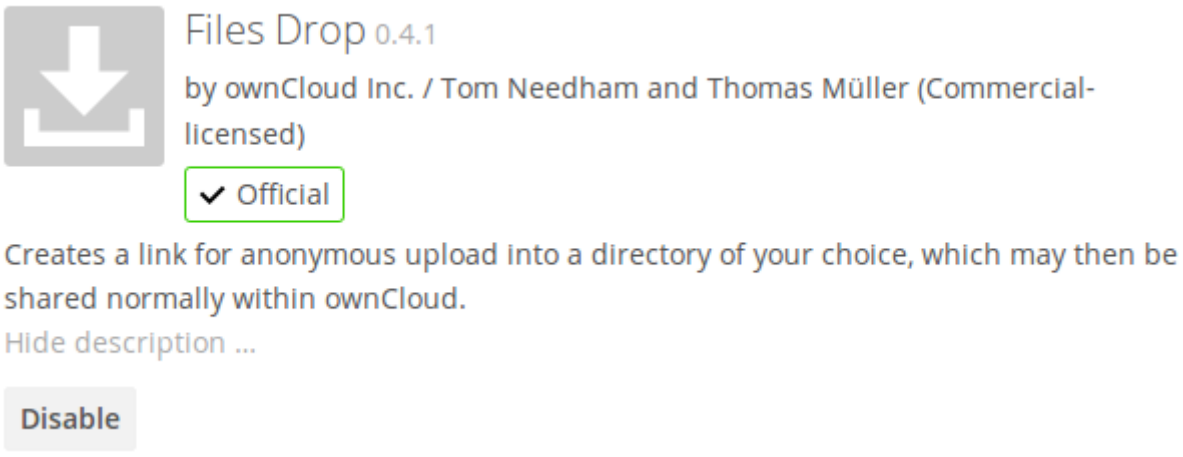
Activation des téléversements anonymes avec Files Drop (Édition Entreprise)

The Files Drop application, introduced in ownCloud 8.0.3 Enterprise Subscription, allows anyone to upload files with the click of a button to the directory of your choosing, without needing a login, and they cannot see or change the contents of the directory. It is the perfect replacement for attaching large files to email, maintaining an FTP server, and commercial file-sharing services.

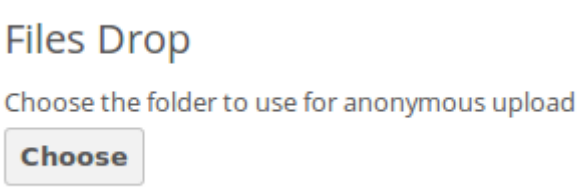
When files are uploaded to your Files Drop directory, you can manage them just like any other ownCloud share: you may share them, restrict access, edit, and delete them.

Setting Up the Files Drop App

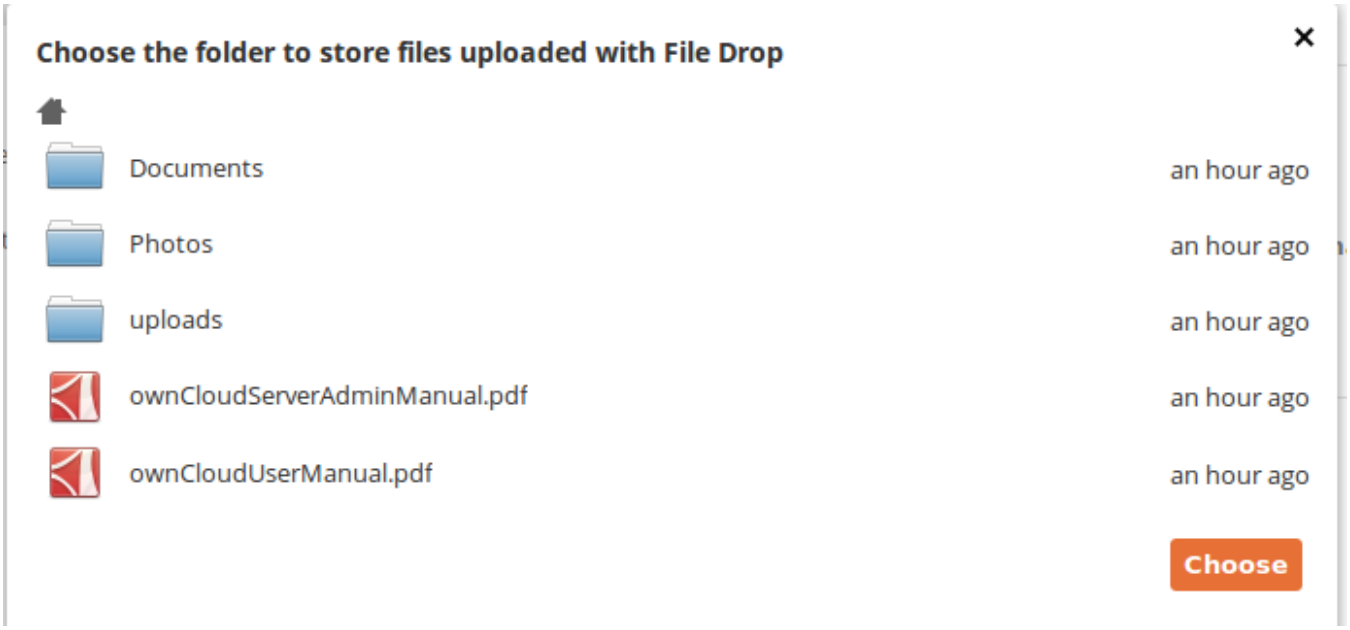
Setting up Files Drop is a matter of a few clicks. First go to your Apps page and enable it.



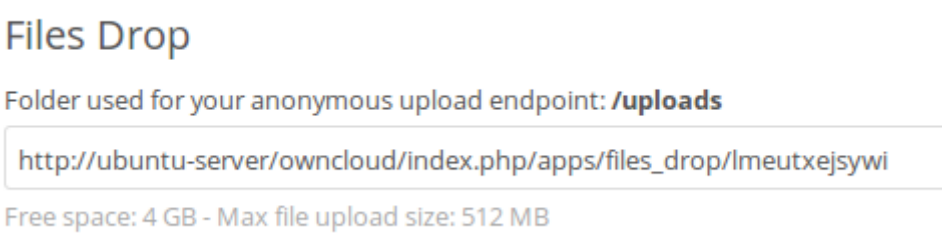
Now your users will see a configuration section on their Personal pages.



Click the **Choose** button to open a dialog to select your upload directory. You may wish to first create a special upload directory (on your Files page), which in the following example is name **upload**.

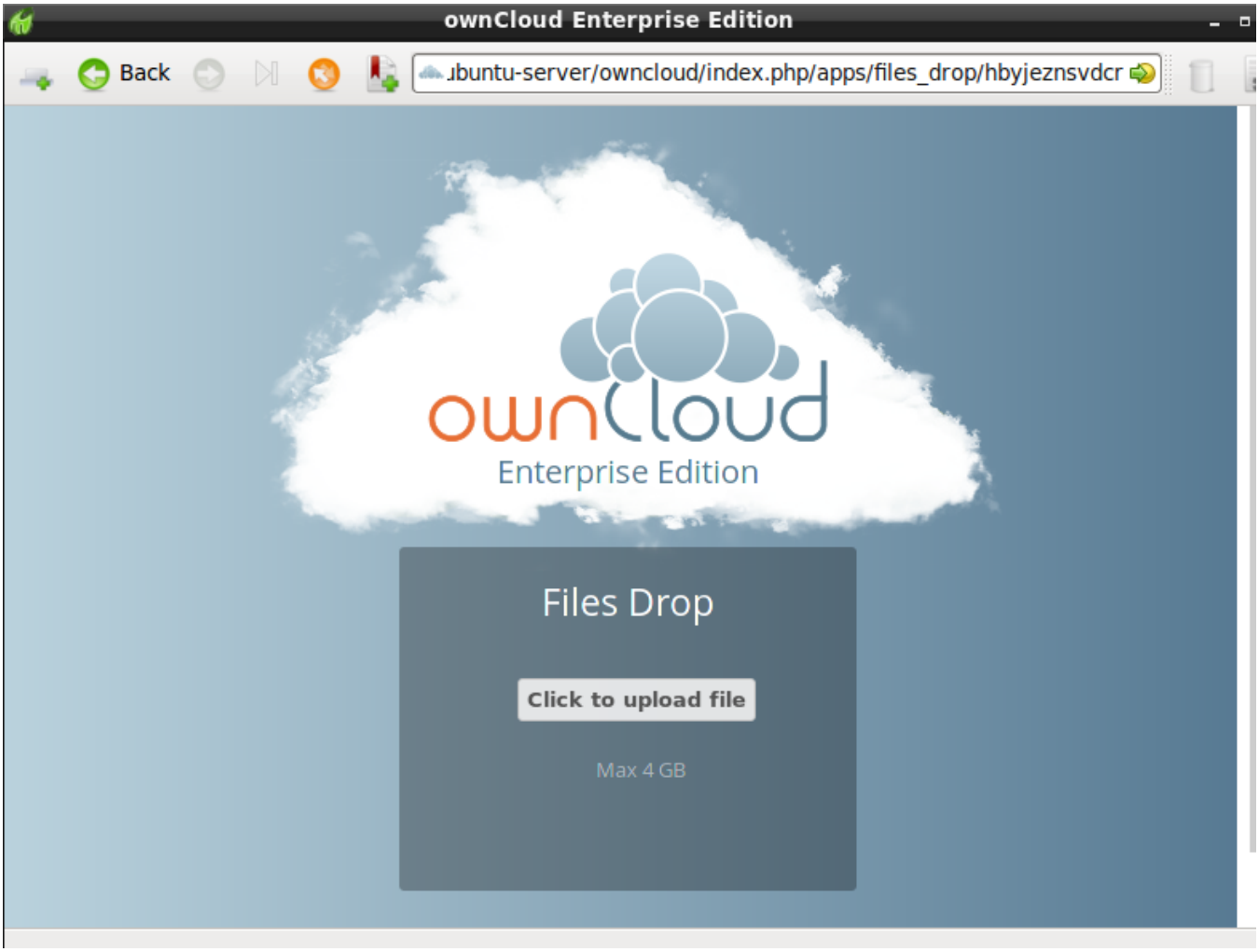


On your Personal page you should now see a URL for your upload directory. Share this URL with anyone you want to allow uploads to your File Drop folder. Note that the maximum upload size in this example is 512MB. (The default ownCloud upload file size limit is 512MB. See *Téléversement de gros fichiers > 512 Mo* to learn how to customize this.)

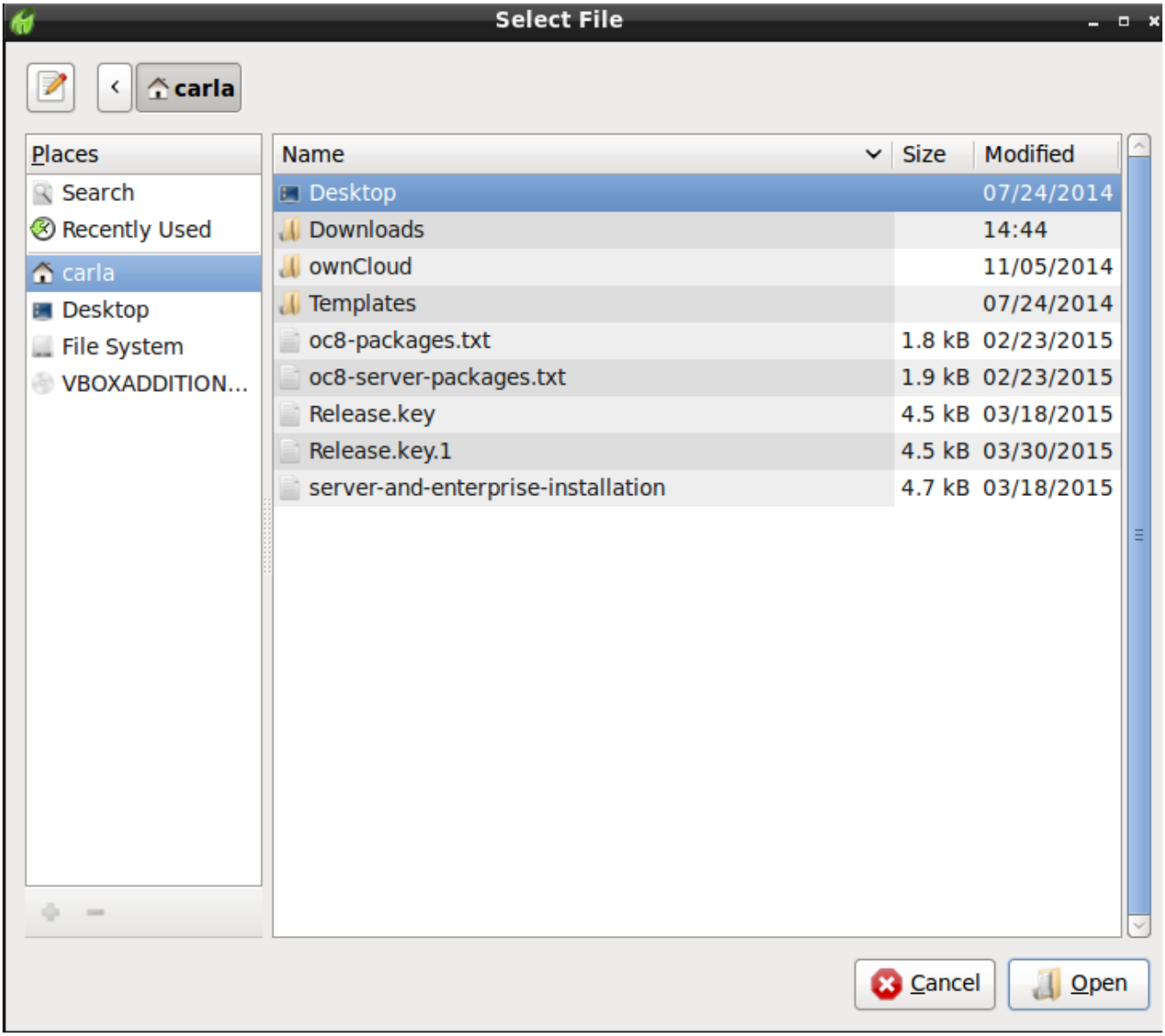


Using the Files Drop App

Uploading files via the Files Drop app is simple. Open your Web browser to the share URL created by ownCloud:



Click the **Click to upload file** button. This opens a file picker, and you select the file or directory you want to upload.



When your upload is completed, you'll see a confirmation message with the filenames.

Advanced File Tagging With the Workflow App (Enterprise only)

New in ownCloud 9.0, the Workflow App provides advanced management of file tagging. The app has three parts: Tag Manager, Automatic Tagging, and Retention.

The Workflow App should be enabled by default (Apps page), and the three configuration modules visible on your ownCloud Admin page.

See Tagging Files in the ownCloud User manual to learn how to apply and filter tags on files.

Tag Manager

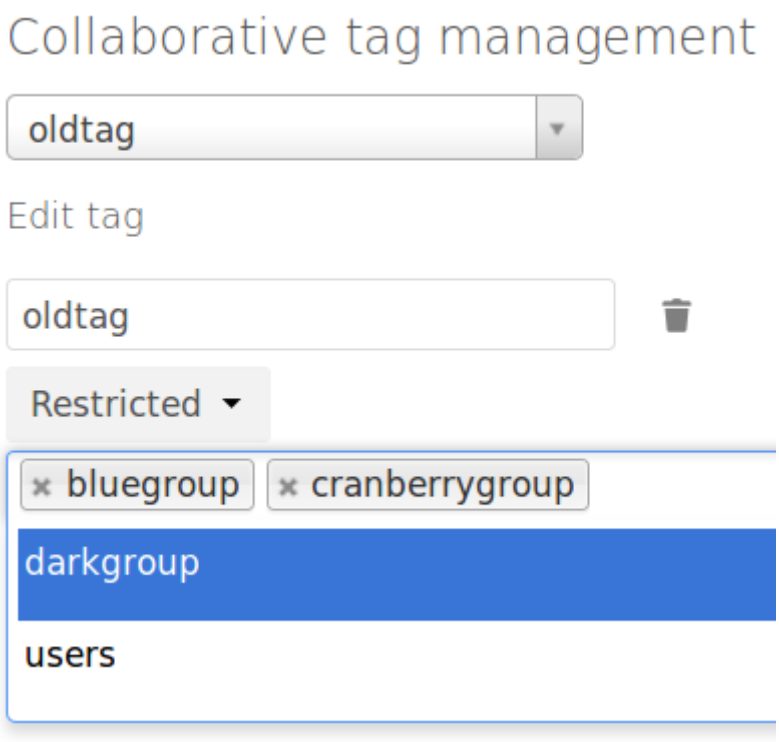
The Tag Manager is for creating new tags, editing existing tags, and deleting tags. Tags may be marked as **Visible**, **Restricted**, or **Invisible**.

Visible means that all users may see, rename, and apply these tags to files and folders.

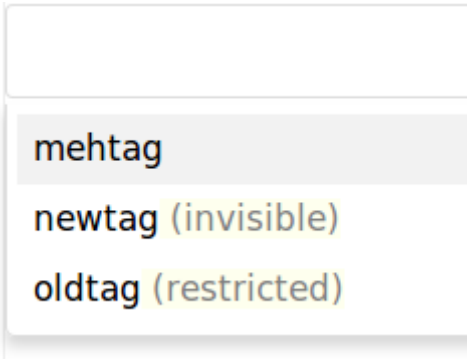
Restricted means tags are assignable and editable only to the user groups that you select. Other users can filter files by restricted tags, but cannot tag files with them or rename them. The tags are marked (restricted).

Invisible means visible only to ownCloud admins.

Use the **Collaborative tag management** module on your ownCloud admin page to edit and create tags.



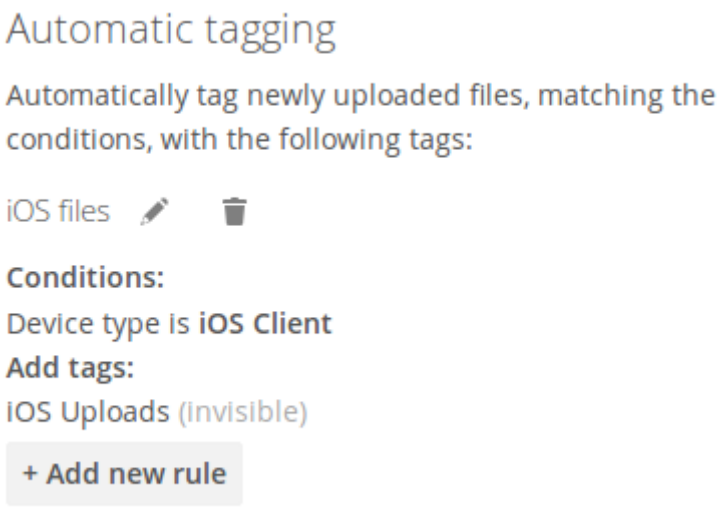
This is what your tags look like in the **Tags** view on your files page. Non-admin users will not see invisible tags, but they will see visible and restricted tags.



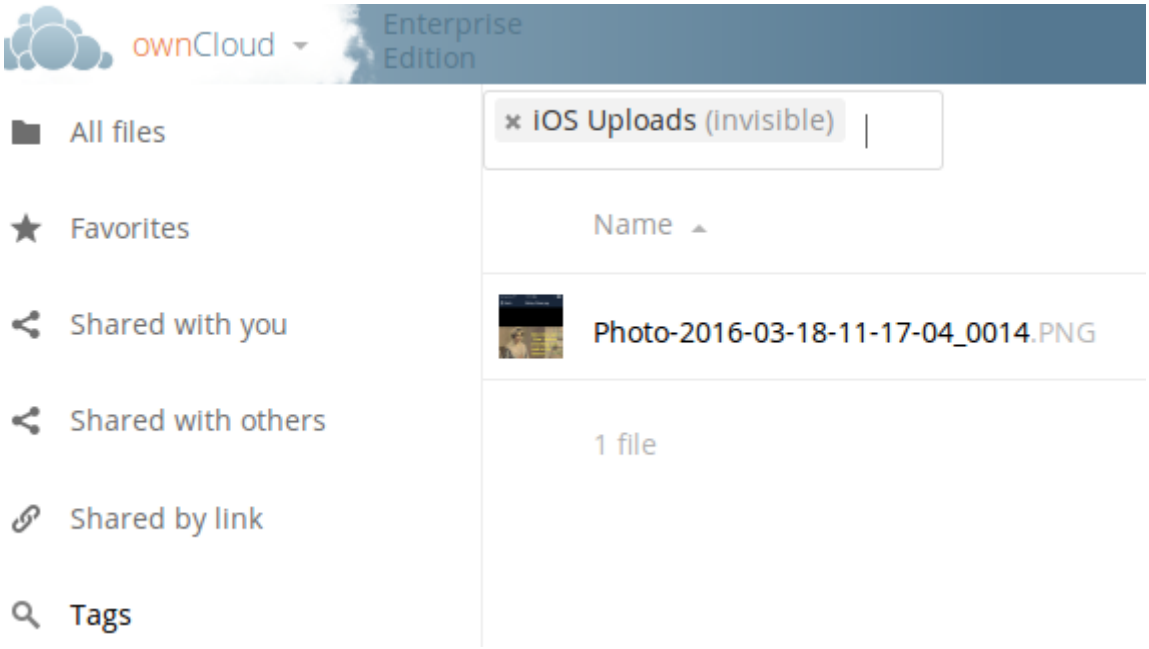
Automatic Tagging

The Automatic Tagging module operates on newly-uploaded files. Create a set of conditions, and then when a file or folder matches those conditions it is automatically tagged. The tag must already have been created with the Tag Manager.

For example, you can assign the invisible tag **iOS Uploads** to all files uploaded from iOS devices. This tag is visible only to admins.



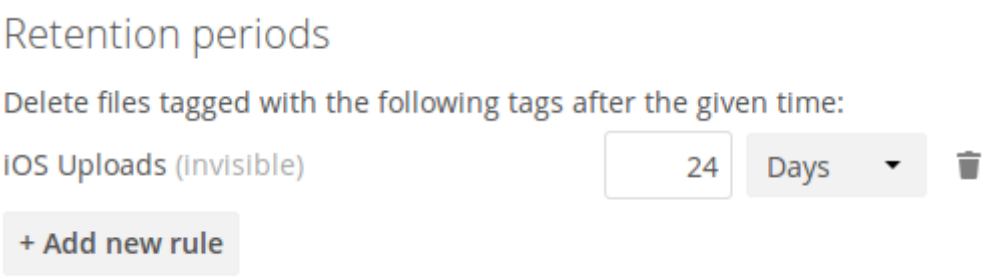
When files with this tag are shared with you, you can view them with the Tags filter on the Files page.



Automatic Tagging is especially useful with the Retention module.

Retention

The Retention module is your housecleaning power tool, because it automatically deletes files after a time period that you specify. Select which tag to set a time limit on, and then set your time limit. File age is calculated from the file mtime (modification time).



For best performance, retention tags should be applied high in your file hierarchy. If subfolders have the same tags as their parent folders, their tags must also be processed, so it will take a little longer.

Retention Engines

There are two retention engines that further allow you to fine-tune your retention settings: **TagBasedRetention** and **UserBasedRetention**. **TagBasedRetention** is the default.

TagBasedRetention: This checks files that have a particular tag assigned. Then it checks (depth-first) the children of the tagged item, before continuing with the other tagged items. Children that have already been checked will not be checked a second time.

This is optimised for processing smaller numbers of files that have multiple retention tags.

UserBasedRetention: Examines files per user. It first iterates over all files and folders (siblings first), then examines the tags for those items and checks their respective retention periods. This is optimised for many files with few retention tags.

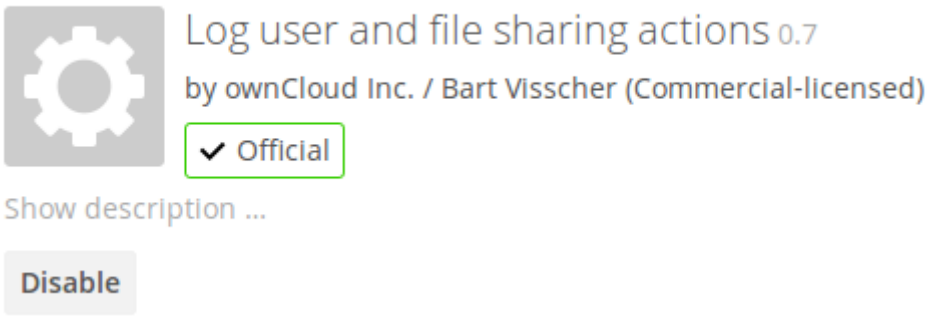
To select UserBasedRetention, add this line to your ee.config.php:

```
'workflow.retention_engine' => userbased,
```

Applications de journalisation (Édition Entreprise)

Applications de journalisation en entreprise

The **Log user and file sharing actions** app (`apps/admin_audit`) records the file sharing activity of your users, file tagging, and user logins and logouts.



Your logging level must be set to at least **Info, warnings, errors, and fatal issues** on your ownCloud admin page, or `'loglevel' => 1` in `config.php`.

View your logfiles on your admin page. Click the **Download logfile** button to dump the plain text log, or open the logfile directly in a text editor. The default location is `owncloud/data/owncloud.log`.

See *Configuration des fichiers journaux* and *Advanced File Tagging With the Workflow App (Enterprise only)* for more information on logging and tagging.

Pare-feu en entreprise (Édition Entreprise)

File Firewall (Enterprise only)

The File Firewall GUI enables you to create and manage firewall rule sets from your ownCloud admin page. The File Firewall gives you finer-grained control of access and sharing, with rules for allowing or denying access, and restrictions per group, upload size, client devices, IP address, time of day, and many more criteria. For additional flexibility the File Firewall also supports regular expressions.

Each rule consists of one or more conditions. A request matches a rule if all conditions evaluate to true. If a request matches at least one of the defined rules, the request is blocked and the file content can not be read or written.

Note

As of ownCloud 9.0, the File Firewall app cannot lock out administrators from the Web interface when rules are misconfigured.

Figure 1 shows an empty firewall configuration panel. Set your logging level to **Failures Only** for debugging, and create a new ruleset by clicking the **Add Group** button. After setting up your rules you must click the **Save Rules** button.

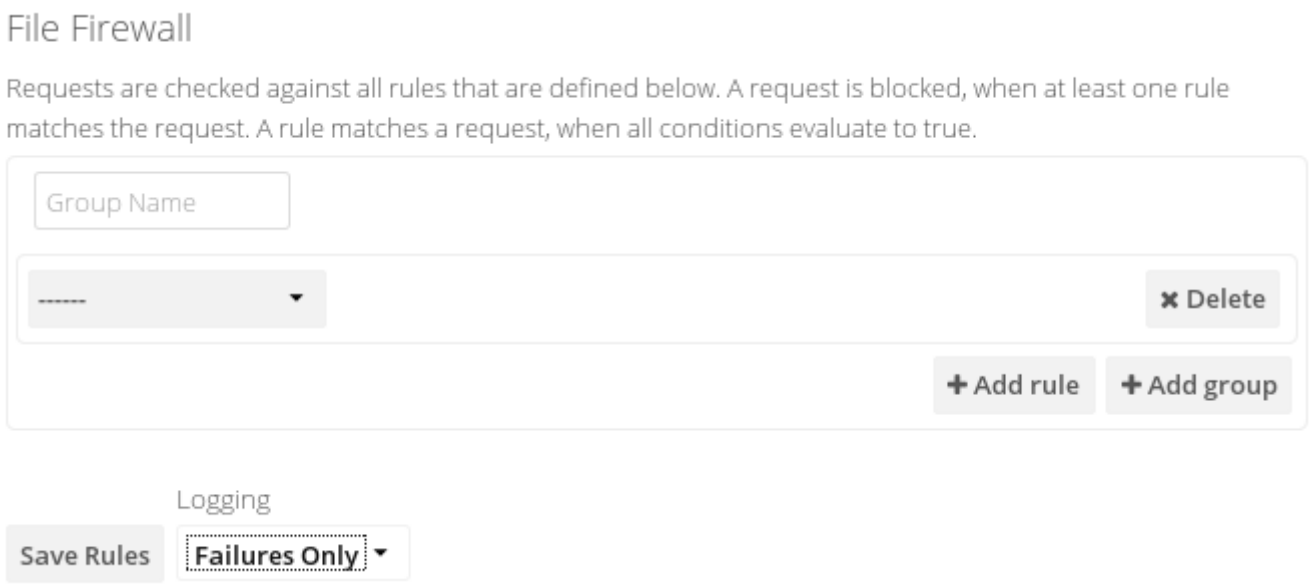


Figure 1: Empty File Firewall configuration panel

Figure 2 shows two rules. The first rule, **No Support outside office hours**, prevents members of the support group from logging into the ownCloud Web interface from 5pm-9am, and also blocks client syncing.

The second rule prevents members of the qa-team group from accessing the Web UI from IP addresses that are outside of the local network.

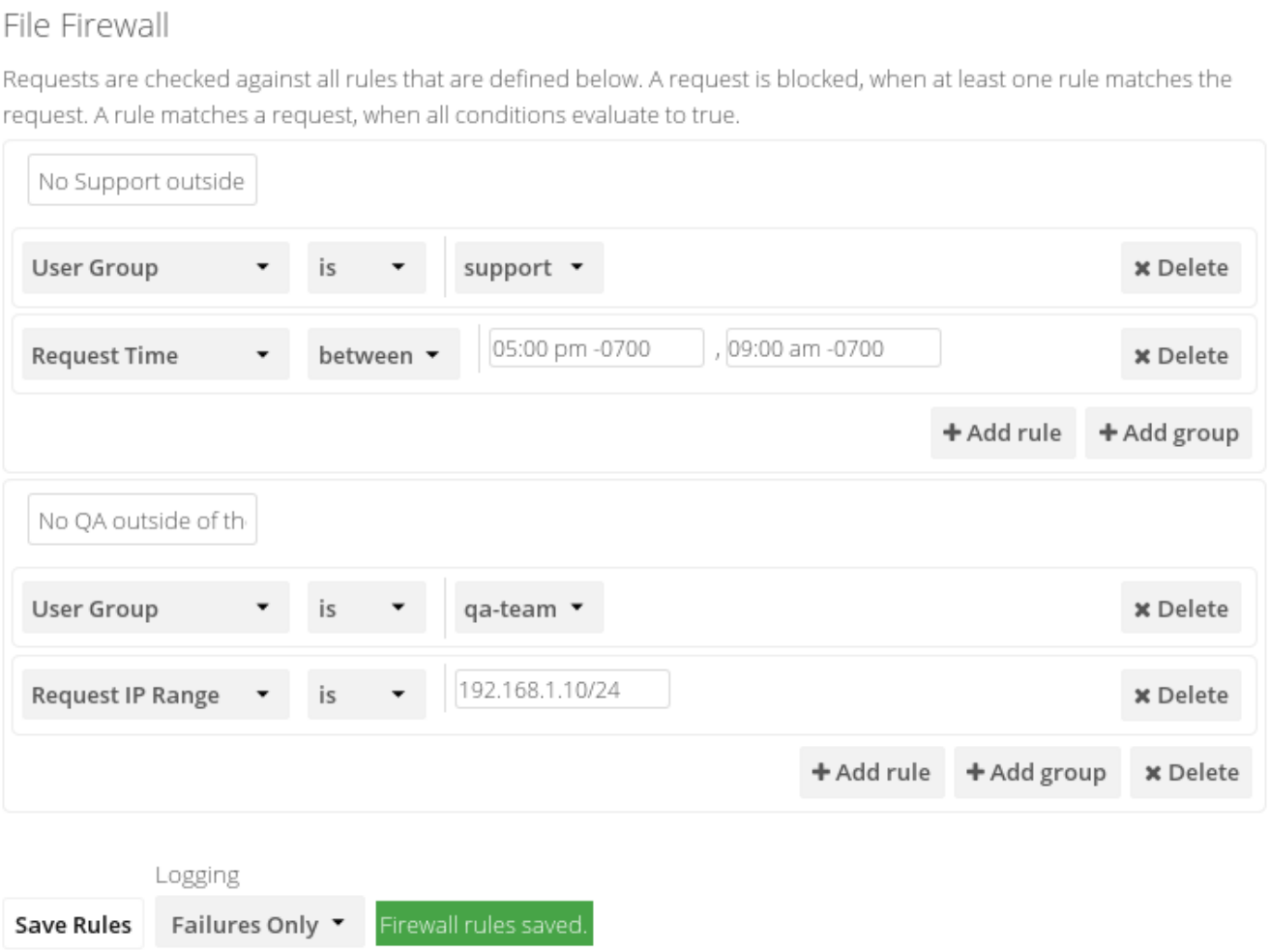


Figure 2: Two example rules that restrict logins per user group

All other users are not affected, and can log in anytime from anywhere.

Available Conditions

User Group
: The user (is|is not) a member of the selected group.

User Agent
: The User-Agent of the request (matches|does not match) the given string.

User Device
: A shortcut for matching all known (`android` | `ios` | `desktop`) sync clients by their User Agent string.

Request Time
: The time of the request (has to|must not) be in a single range from beginning time to end time.

Request URL
: The **full page URL** (has to|must not) (match|contain|begin with|end) with a given string.

Request Type
: The request (is|is not) a (WebDAV|public share link|other) request.

Request IP Range (IPv4) and IP Range (IPv6)
: The request's `REMOTE_ADDR` header (is|is not) matching the given IP range.

Subnet (IPv4) and Subnet (IPv6)
: The request's `SERVER_ADDR` header (is|is not) matching the given IP range.

File Size Upload
: When a file is uploaded the size has to be (less|less or equal|greater|greater or equal) to the given size.

File Mimetype Upload

When a file is uploaded the mimetype (is|is not|begins with|does not begin with|ends with|does not end with) the given string.

System File Tag

One of the parent folders or the file itself (is|is not) tagged with a System tag.

Regular Expression

The File Firewall supports regular expressions, allowing you to create custom rules using the following conditions:

- IP Range (IPv4)
- IP Range (IPv6)
- Subnet (IPv4)
- Subnet (IPv6)
- User agent
- User group
- Request URL

You can combine multiple rules into one rule. E.g., if a rule applies to both the support and the qa-team you could write your rule like this:

```
Regular Expression > ^(support|qa-team)$ > is > User group
```

No Manual Editing

We do not recommend modifying the configuration values directly in your `config.php`. These use JSON encoding, so the values are difficult to read and a single typo will break all of your rules.

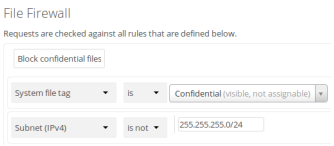
Controlling Access to Folders

The easiest way to block access to a folder, starting with ownCloud 9.0, is to use a system tag. A new rule type was added which allows you to block access to files and folders, where at least one of the parents has a given tag. Now you just need to add the tag to the folder or file, and then block the tag with the File Firewall.

This example blocks access to any folder with the tag "Confidential".

Block by System Tag:

```
System file tag:   is       "Confidential"
Subnet IPv4:       is not   "255.255.255.0/24"
```



Custom Configuration for Branded Clients

If you are using *branded ownCloud clients*, you may define `firewall.branded_clients` in your `config.php` to identify your branded clients in the firewall **"User Device"** rule.

The configuration is a `User-Agent => Device` map. `Device` must be one of the following:

- android
- android_branded
- ios
- ios_branded
- desktop

- desktop_branded

The `User-Agent` is always compared all lowercase. By default the agent is compared with `equals`. When a trailing or leading asterisk, `*`, is found, the agent is compared with `starts with` or `ends with`. If the agent has both a leading and a trailing `*`, the string must appear anywhere. For technical reasons the `User-Agent` string must be at least 4 characters (including wildcards). (When you build your branded client you have the option to create a custom User Agent.)

In this example configuration you need to replace the example User Agent strings, for example `'android_branded'`, with your own User Agent strings:

```
// config.php

 'firewall.branded_clients' => array(
   'my ownbrander android user agent string' => 'android_branded',
   'my ownbrander second android user agent string' => 'android_branded',
   'my ownbrander ios user agent string' => 'ios_branded',
   'my ownbrander second ios user agent string' => 'ios_branded',
   'my ownbrander desktop user agent string' => 'desktop_branded',
   'my ownbrander second desktop user agent string' => 'desktop_branded',
 ),
```

The Web UI dropdown then expands to the following options:

- Android Client - always visible
- iOS Client - always visible
- Desktop Client - always visible
- Android Client (Branded) - visible when at least one `android_branded` is defined
- iOS Client (Branded) - visible when at least one `ios_branded` is defined
- Desktop Client (Branded) - visible when at least one `desktop_branded` is defined
- All branded clients - visible when at least one of `android_branded`, `ios_branded` or `desktop_branded` is defined
- All non-branded clients - visible when at least one of `android_branded`, `ios_branded` or `desktop_branded` is defined
- Others (Browsers, etc.) - always visible

Then these options operate this way:

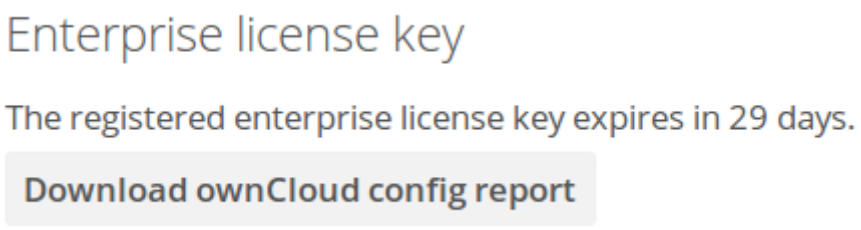
- The `* Client` options only match `android`, `ios` and `desktop` respectively.
- The `* Client (Branded)` options match the `*_branded` agents equivalent.
- `All branded clients` matches: `android_branded`, `ios_branded` and `desktop_branded`
- `All non-branded clients` matches: `android`, `ios` and `desktop`

Dépannage en entreprise

When you have problems with your ownCloud Enterprise installation, refer to *Dépannage général* to see if you can resolve your issue without opening a support ticket. If you need to open a support ticket, use the Open Ticket button in your account on https://customer.owncloud.com/owncloud/.

Bug reports and trouble tickets usually need a copy of your ownCloud server configuration report. You have two ways to generate a configuration report.

1. Use the *occ config command*.
2. Use the **Enterprise license key** app on your ownCloud Admin page to generate the report with the click of a button.



Both methods automatically obscure passwords and secrets.