

Proyecto Fin de Carrera
Ingeniería Industrial

Despliegue y configuración de una plataforma privada de almacenamiento en la nube

Autor:

José María Solís Toro

Tutor:

David Muñoz de la Peña Sequedo

Profesor titular

Departamento. Ingeniería de Sistemas y Automática

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2018

Índice de contenidos

Índice de contenidos.....	1
1. Introducción y motivación.	3
1.1. Breve historia de los sistemas para compartir archivos por Internet.....	3
1.2. Razones para optar por almacenamiento en la nube autogestionado.	5
1.3. Presentación de <i>Nextcloud</i>	6
1.4. Objetivo de esta memoria.	7
2. Preparación del servidor.....	8
2.1. Instalación del sistema Ubuntu 18.04 LTS.	8
2.2. Instalación de prerequisites de <i>Nextcloud 13</i>	12
2.3. Configuración preliminar del servidor web.	13
3. Instalación de <i>Nextcloud</i>	15
3.1. Instalación del núcleo de <i>Nextcloud</i>	15
3.2. Configuración de <i>Nextcloud</i> y el servidor web para la conexión cifrada.....	17
3.3. Auditoría de seguridad interna de <i>Nextcloud</i>	21
3.4. Optimización del rendimiento.	24
4. Clientes de sincronización.....	27
4.1. Cliente de sincronización de escritorio.	27
4.2. Clientes de sincronización para dispositivos móviles.	30
5. Diseño modular de <i>Nextcloud</i> . Aplicaciones oficiales adicionales y de terceros.....	31
6. Administración de usuarios en <i>Nextcloud</i>	32
6.1. Usuarios y grupos. Cuotas de espacio.....	32
6.2. Control de acceso a archivos.	32
7. Seguridad y cifrado en <i>Nextcloud</i>	34
7.1. Cifrado en la comunicación entre cliente y servidor.	34
7.2. Cifrado del almacenamiento en el servidor.	34

7.3.	Cifrado de extremo a extremo.....	35
7.4.	Auditoría de seguridad externa para servidores <i>Nextcloud</i>	36
8.	Copias de seguridad.....	37
8.1.	Sistema de revisiones y papelera de <i>Nextcloud</i>	37
8.2.	Copia de seguridad completa en un servidor externo.....	37
8.3.	Configuración adicional de <i>backupninja</i>	51
8.4.	Protocolo de restauración del <i>backup</i> externo.	53
9.	Administración del servidor.....	55
9.1.	Renovación automática de certificados.....	55
9.2.	Envío automático de información del servidor por correo electrónico.....	56
10.	Conclusiones.	58

1. Introducción y motivación.

1.1. Breve historia de los sistemas para compartir archivos por Internet.

Antes de la llegada de los sistemas de compartición de archivos y trabajo en grupo con sincronización de carpetas en la nube, había básicamente dos sistemas que permitían el trabajo en grupo con archivos y carpetas:

1. **Correo electrónico con archivos adjuntos.** Al popularizarse el servicio de correo electrónico entre usuarios, se usaba (y todavía se usa) éste para enviar y recibir contenido. Aunque es una solución de bajo coste económico, no es sencillo llevar un control de los archivos compartidos, su almacenamiento y su organización.
2. **Sistemas de carpetas en red local extendido mediante VPN** (red privada virtual). El despliegue de estos sistemas podía ser complejo y caro (redes *Active Directory* de Microsoft, por ejemplo), aunque satisfacían la mayoría de las necesidades corporativas de seguridad y permisos granulados por departamentos y roles. Para el acceso a través de Internet se implementaba una capa adicional de enrutado mediante una *Red Privada Virtual*, VPN en sus siglas en inglés, que precisa configuraciones adicionales (más o menos complejas) en servidores y clientes, mantenimiento adicional, etc. Todo este sistema completo podía llegar a ser bastante costoso de desplegar, operar y mantener en términos de licencias y mano de obra cualificada.

Los fundadores de [Dropbox](#), servicio pionero en la sincronización y almacenamiento de contenido en la nube, vieron una oportunidad de negocio, al constatar que la mayoría de los usuarios continuaban usando el correo electrónico como alternativa de bajo coste para compartir contenido, siendo esto muy ineficiente y engorroso. Dropbox nació en junio de 2007, con enorme éxito desde entonces.

La novedad del servicio de Dropbox fue que los usuarios podían compartir contenido entre sus diferentes dispositivos y con otras personas (incluso aunque no fueran usuarios de Dropbox), mediante un sistema de carpetas sincronizadas a través de la *nube*. Conviene aclarar aquí que *nube*, en mi opinión, no es más que un eufemismo usado para referirse a un ordenador que se encuentra físicamente en un lugar que no sabemos cuál es, operado por personas que no conocemos, y propiedad de una empresa o persona que podemos saber cuál es (la propietaria del servicio) o no. La mayoría de la gente común no son conscientes de lo que implica eso.

Hoy día, además de Dropbox, existen otros servicios competidores, ya que es un mercado en alza debido principalmente a dos factores:

1. La creciente demanda de almacenamiento, por parte de usuarios y empresas, para todo el contenido digital que generan, de forma masiva, gracias a herramientas informáticas de todo tipo: fotos, vídeos, documentación...
2. La importancia estratégica del control de la moneda de cambio con más valor en este siglo XXI: **la información.**

Basta con mirar la lista de las 10 mayores empresas por cotización bursátil de la actualidad (2018) para constatar que las 5 primeras tienen como ámbito principal de negocio la información:

Empresa	Capitalización (Millones de \$)	País	Sector
Apple Inc.	886	Estados Unidos	Tecnologías de la Información
Alphabet Inc A	726	Estados Unidos	Tecnologías de la Información
Microsoft Corp	657	Estados Unidos	Tecnologías de la Información
Amazon.com Inc	563	Estados Unidos	Consumo y tecnologías de la información
Facebook Inc A	522	Estados Unidos	Tecnologías de la Información
Berkshire Hathaway B	485	Estados Unidos	Finanzas
Johnson & Johnson	379	Estados Unidos	Salud
JP Morgan Chase & Co	366	Estados Unidos	Finanzas
Exxon Mobil Corp	351	Estados Unidos	Energía
Bank of America	301	Estados Unidos	Finanzas

Las cuatro primeras poseen ya un servicio de almacenamiento y compartición de archivos en la nube, estilo Dropbox, propio:

1. Apple: [iCloud](#).
2. Alphabet (Google): [Google Drive](#).
3. Microsoft: [OneDrive](#).
4. Amazon: [Amazon Drive](#).

Y la quinta en lista, Facebook, es de facto un sistema de almacenamiento y compartición del contenido multimedia generado por los usuarios de sus redes sociales (entre otras muchas cosas también).

1.2. Razones para optar por almacenamiento en la nube autogestionado.

Ante la popularidad y el uso masivo que alcanzan estos servicios de almacenamiento de contenido en la *nube* (entendida ésta como metáfora para referirse a un ordenador / servidor en manos de un tercero), cabe preguntarse si las condiciones de uso y privacidad que imponen las empresas proveedoras del servicio son deseables, adecuadas o incluso legales, para usuarios y empresas que las usan.

Dar respuesta a esa pregunta es complejo, pero a poco que profundizamos en las condiciones de uso y privacidad de los servicios de estas empresas (Google, Microsoft, Apple, Facebook, etc.), podemos llegar a la conclusión de que todo nuestro contenido será **examinado, indexado y utilizado comercialmente** sin que apenas tengamos control ni conocimiento del tratamiento digital de dicho contenido.

A la mayoría de los usuarios a título personal de este tipo de servicios no parece importarles (todavía) las consecuencias que para su privacidad, libertad y seguridad, tienen la acumulación y tratamiento informático de su información privada. La mayoría, probablemente, por desconocimiento de las impresionantes capacidades de análisis y síntesis en el tratamiento digital masivo (técnicas de *Big Data* e Inteligencia Artificial) de los datos que pone en manos de estas empresas de manera voluntaria pero inconsciente.

Como ejemplo de estas técnicas, podemos citar un informe publicado por la Universidad Carlos III de Madrid, escrito por los investigadores José González Cabañas, Ángel Cuevas y Rubén Cuevas, en el que se afirma que Facebook “atribuye a más del 73 por ciento de los usuarios en la Unión Europea (**el 40 por ciento de la población de la UE**), al menos una de entre 500 preferencias publicitarias consideradas sensibles, tras un sofisticado proceso de filtrado y análisis de los datos con técnicas de procesado del lenguaje natural y clasificación manual para preferencias complejas de determinar con software.” Ese 40% de población europea está identificado con nombres y apellidos, y la información sensible a la que se hace alusión incluye al menos una etiqueta (muchas más en la mayoría de los casos) que permiten definir con precisión sus intereses, inclinaciones, o vínculos de pertenencia, en política, religión, sexualidad, salud, etnia y otros datos. Todos estos datos están considerados como muy sensibles, existiendo leyes muy estrictas que los protegen pero que se aplicaban tradicionalmente en los ámbitos en los que esa información se usaba y almacenaba: la salud en el sistema sanitario, por ejemplo.

En estos momentos gran parte de esa información que había estado repartida en diferentes organismos (públicos y privados), protegida por leyes específicas de cada ámbito, y que solo se usaba para consulta de datos de personas específicas (como expedientes médicos en una consulta), es propiedad de una empresa privada extranjera (y de fuera de la Unión Europea) como es Facebook, y tiene la capacidad de indexarlo, procesarlo **y venderlo al mejor postor** de una manera tan rápida y eficiente como nunca antes en la historia.

A través del salto a la luz pública de casos como el de [Cambridge Analytica](#), se empieza a ser consciente de las enormes consecuencias que tiene la capacidad de manipular a millones de personas usando la información privada de sus perfiles en redes sociales. Resultados de elecciones

como la que aupó a Donal Trump a la Casa Blanca, o dieron la victoria al Brexit en Reino Unido están en entredicho.

Aunque el tratamiento de datos en redes sociales no es objeto de esta memoria, sí sirve para ilustrar las capacidades y consecuencias del proceso de grandes cantidades de datos personales que también se dan en los servicios de almacenamiento en la nube ofrecidos por grandes compañías como Google, Microsoft, Amazon o Dropbox.

En la cultura corporativa sí se tiene claro que la información de las empresas no debe tratarse sin un control estricto, pero de nuevo muchas veces no se es consciente de que los servicios usados (probablemente de manera gratuita) por departamentos o trabajadores individuales tienen los graves inconvenientes que acabamos de reseñar.

Incluso aunque haya un contrato de servicio con proveedores de almacenamiento en la nube en el que no se contemple cesión de datos alguna, las empresas (y particulares) deben preguntarse cuán seguro está su contenido. Éstos seguramente incluyan información muy sensible como claves bancarias, contabilidad financiera, datos personales de clientes protegidos por ley, información confidencial de proyectos y operaciones de la empresa, etc.

También es reseñable el hecho de que todos los gigantes tecnológicos de los que estamos hablando son estadounidenses, y de alguna manera u otra tienen servidumbres que pagar para con su gobierno. En concreto, Dropbox fue señalada, entre otras compañías de tecnologías de la información, como una de las involucradas dentro del programa de vigilancia electrónica a cargo de la Agencia de Seguridad Nacional (NSA) de los Estados Unidos, según los informes y documentos filtrados por el exempleado de la Agencia Central de Inteligencia (CIA), Edward Snowden, en junio de 2013. El 9 de abril de 2014, Condoleezza Rice, quien entre 2005 y 2009 fue secretaria de Estado de Estados Unidos durante la presidencia de George W. Bush, entró en el consejo de administración de Dropbox. Su nombramiento causó una gran polémica, generando el rechazo de muchos de sus usuarios y campañas en contra como la de [Drop-Dropbox](#). Ha sido acusada por dicha plataforma de “apoyar y haber autorizado muchas escuchas telefónicas ilegales” en su propio país.

1.3. Presentación de *Nextcloud*.

[Nextcloud](#) es una aplicación web, basada en **PHP** y bases de datos (**MySQL** o **MariaDB**). Sus principales características, y razones por las que yo elegí dicho desarrollo para montar un sistema de intercambio de archivos en mi ámbito empresarial, son:

1. Permite el intercambio de contenido, la colaboración y comunicación entre usuarios, dentro de la empresa y fuera de ella, hacia y desde clientes y proveedores.
2. Protege, controla y monitoriza el contenido y su intercambio. Garantiza el cumplimiento de los requerimientos empresariales y legales europeos (la última versión, que es la que tratamos aquí, está diseñada para poder cumplir la GDPR, **General Data Protection Regulation** o regulación europea para la protección de datos).
3. Está impulsado por una comunidad transparente basada en el modelo de código abierto, libre de servidumbres y que desarrolla y mejora de manera constante la aplicación web,

los clientes de escritorio y las aplicaciones móviles. Esta comunidad tiene su sede en Alemania.

En resumen: Nextcloud permite un **aumento de la productividad** teniendo disponible el contenido de la empresa en todas las plataformas y dispositivos, **sin perder el control** de dicho contenido, y **cumpliendo los requerimientos legales europeos**.

Por ello, [Nextcloud ha sido elegido por el Gobierno Federal Alemán](#), a través de su organismo oficial “Centro para las Tecnologías de la Información” (ITZBund), como solución eficiente y segura para el intercambio de contenido entre las administraciones públicas. No en vano, el equipo de desarrollo de *Nextcloud* está en Alemania, y esa es una de las causas por las que tiene en cuenta (y se esfuerzan por cumplir) las regulaciones europeas, mucho más avanzadas que las norteamericanas en lo concerniente a la protección de la privacidad de sus ciudadanos.

1.4. Objetivo de esta memoria.

El objetivo de esta memoria es que un técnico con conocimientos medios / bajos de servidores Linux, partiendo de una máquina (alquilada como es en el caso de esta guía o en propiedad) a la que se le pueda instalar la distribución *Ubuntu 18.04 LTS* para servidores (basada en Debian) o similar, siguiendo los pasos que aquí se detallan, pueda completar el despliegue de un servicio de compartición de contenido y trabajo en grupo basado en *Nextcloud*, alcanzando los siguientes objetivos:

1. Configuración del servidor para un rendimiento optimizado con *Nextcloud* y las tecnologías en las que se apoya: servidor web *Apache*, *PHP* y *MySQL*.
2. Imposición de conexiones cifradas entre cliente y servidor, respaldadas por los certificados emitidos por la autoridad certificadora independiente [Let's Encrypt](#).
3. Implementación de una política de copias de seguridad completas en un servidor remoto, junto con el protocolo de restauración de los datos y el servicio completo.
4. Implementación de tareas de mantenimiento del servidor sencillas: control del espacio disponible, renovación automática de certificados, etc.
5. Modelo de seguridad opcional con cifrado de datos en el servidor y almacenamientos externos y, adicionalmente, cifrado de extremo a extremo. Este último todavía en fase de pruebas por el equipo de desarrollo de *Nextcloud*.

2. Preparación del servidor.

Para montar el servidor de almacenamiento en la nube he elegido una máquina “modesta” y de bajo coste, alquilada a la empresa de servidores [Kimsufi](#) (filial de bajo coste de [OVH](#)):

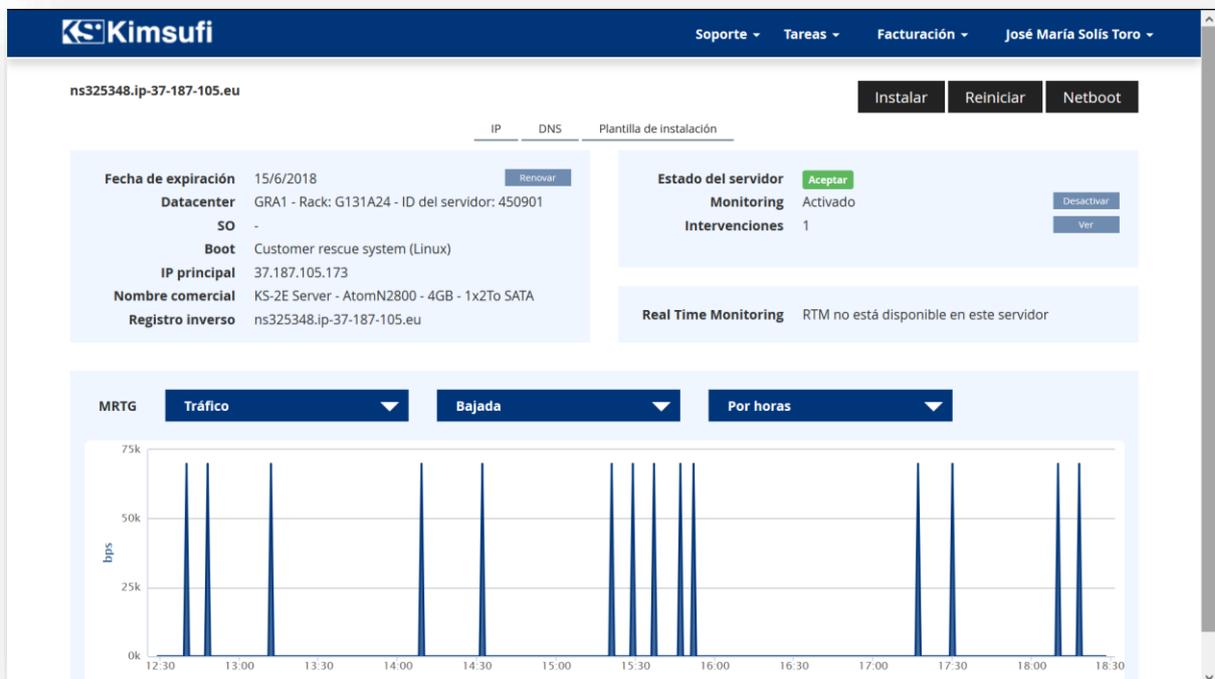


Ilustración 2.1

Las características principales de la máquina son:

- Procesador Atom N2800 (1860MHz, 2 núcleos y 4 hilos).
- 4GBytes de RAM.
- 2TBytes de espacio en disco duro (no SSD).

Dichas características son suficientes para desplegar un servidor que dé servicio a un número de usuarios modesto (alrededor de 10 para que funcione de manera holgada) con mucho espacio.

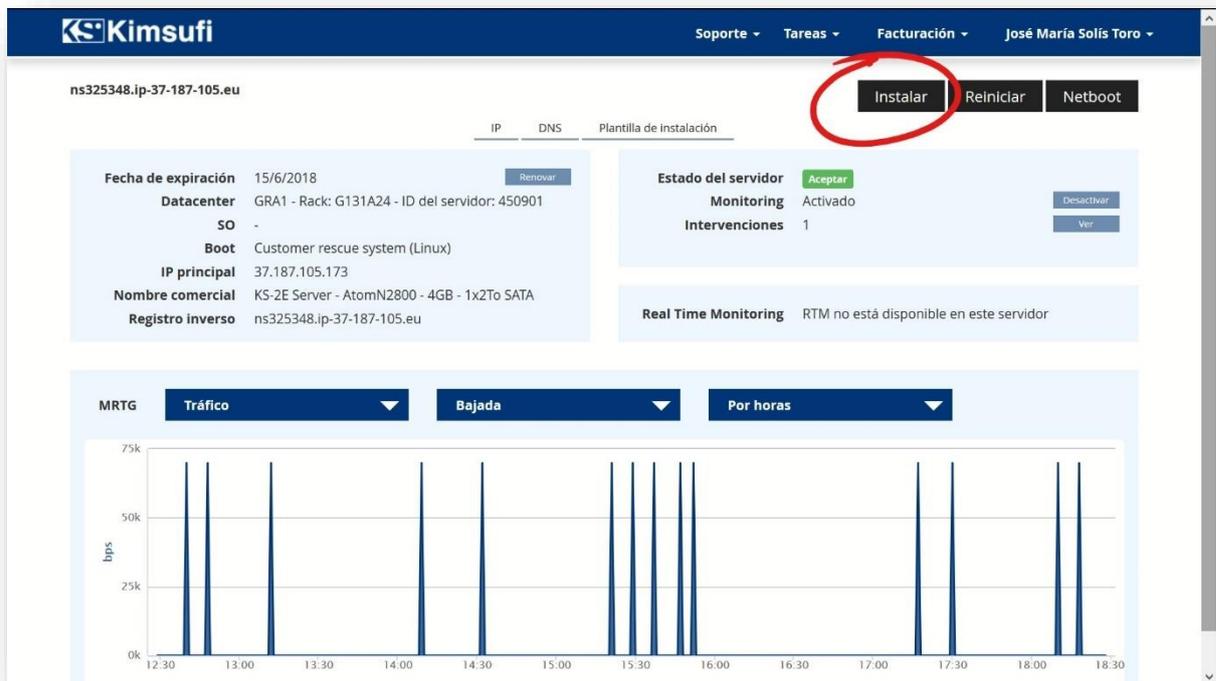
Destacar que se trata de una máquina física, no una instancia virtual dentro de un servidor más grande.

2.1. Instalación del sistema Ubuntu 18.04 LTS.

Para comenzar se debe instalar un sistema operativo en la máquina. He optado por un sistema operativo Linux, requisito de *Nextcloud*, y en concreto por un [Ubuntu 18.04 LTS](#) para servidores, última versión estable y con soporte extendido de 5 años, ideal para despliegues empresariales a largo plazo.

2. Preparación del servidor.

La instalación se realiza desde el panel de control de la cuenta de cliente de Kimsufi, pulsando en el botón de **Instalar**:



The screenshot shows the Kimsufi control panel for server ns325348.ip-37-187-105.eu. The 'Instalar' button is circled in red. The server details include:

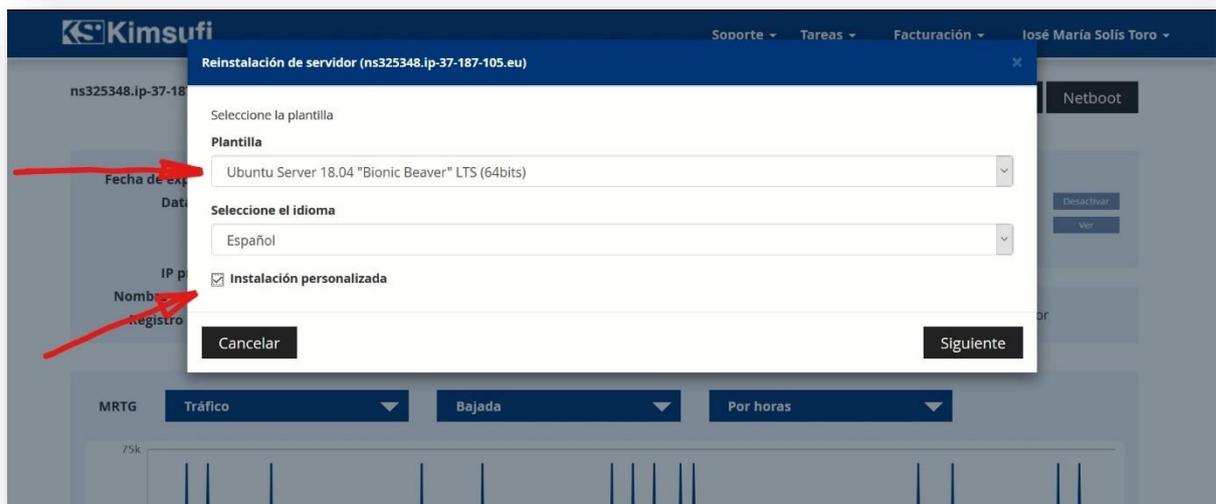
Fecha de expiración	15/6/2018	Renovar
Datacenter	GRA1 - Rack: G131A24 - ID del servidor: 450901	
SO	-	
Boot	Customer rescue system (Linux)	
IP principal	37.187.105.173	
Nombre comercial	KS-2E Server - AtomN2800 - 4GB - 1x2To SATA	
Registro inverso	ns325348.ip-37-187-105.eu	

Server status: Estado del servidor (Aceptar), Monitoring (Activado), Intervenciones (1). Real Time Monitoring: RTM no está disponible en este servidor.

MRTG Traffic graph showing data for Bajada (Down) over time (12:30 to 18:30).

Ilustración 2.2

Tras lo cual seleccionamos el sistema operativo que queremos y tenemos que asegurarnos que marcamos la casilla de **Instalación personalizada**:



The screenshot shows the 'Reinstalación de servidor (ns325348.ip-37-187-105.eu)' dialog box. The 'Plantilla' dropdown is set to 'Ubuntu Server 18.04 "Bionic Beaver" LTS (64bits)'. The 'Idioma' dropdown is set to 'Español'. The 'Instalación personalizada' checkbox is checked. The 'Cancelar' and 'Siguiete' buttons are visible at the bottom of the dialog.

Ilustración 2.3

Debemos seleccionar **Instalación personalizada** porque la tabla de particiones que Ubuntu crea por defecto no es adecuada para el uso que le vamos a dar al servidor. En la siguiente pantalla se nos presenta la tabla de particiones por defecto, que no nos sirve, y debemos eliminar todas las particiones (excepto *Swap*, que es obligatoria y le asignaremos 4100Mbytes) dejando que la partición raíz (Root ó /) ocupe el espacio restante de los 2TBytes, tal y como muestra la siguiente imagen:

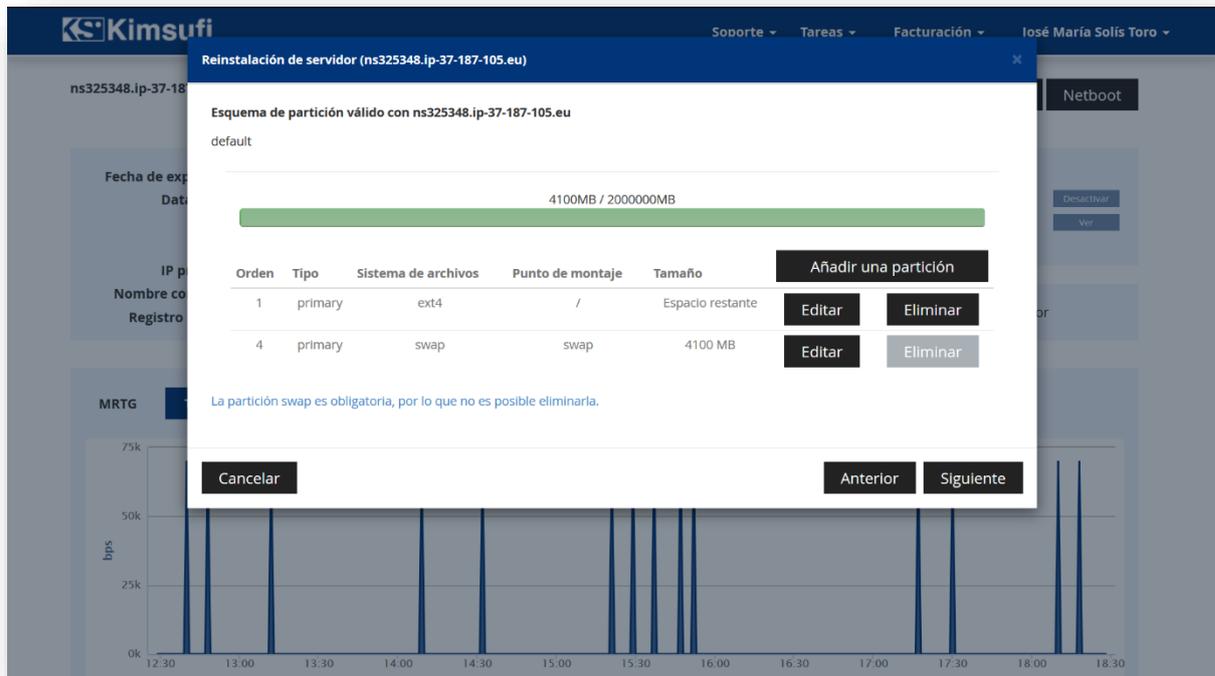


Ilustración 2.4

Esto es muy importante porque el particionado por defecto asigna la mayor parte del espacio a la partición **/home** y sólo 20Gbytes a la raíz **/**. Al ser *Nextcloud* es una aplicación web, se instala en la ruta **/var/www/html/**, y sólo tendríamos disponible de almacenamiento 20GBytes menos lo que ocupe el sistema operativo y los archivos temporales, desaprovechando el resto de los 2TBytes que estarían asignados a **/home** y sin uso alguno.

En el siguiente paso debemos configurar el nombre del servidor o **Hostname**, que será el nombre o URL que escribamos en los navegadores para acceder al servidor. He optado por asignar un subdominio dentro de un nombre de dominio de mi propiedad:

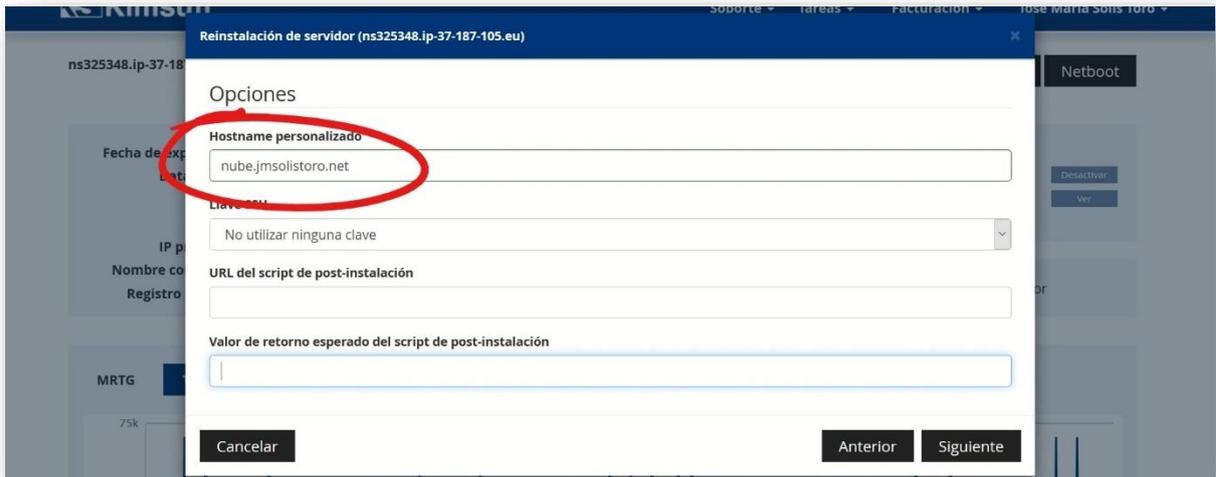


Ilustración 2.5

Así, desde la sección de DNS de mi dominio puedo asignar un registro para que **nube.jmsolistoro.net** coincida con la dirección IP del servidor con *Nextcloud*:

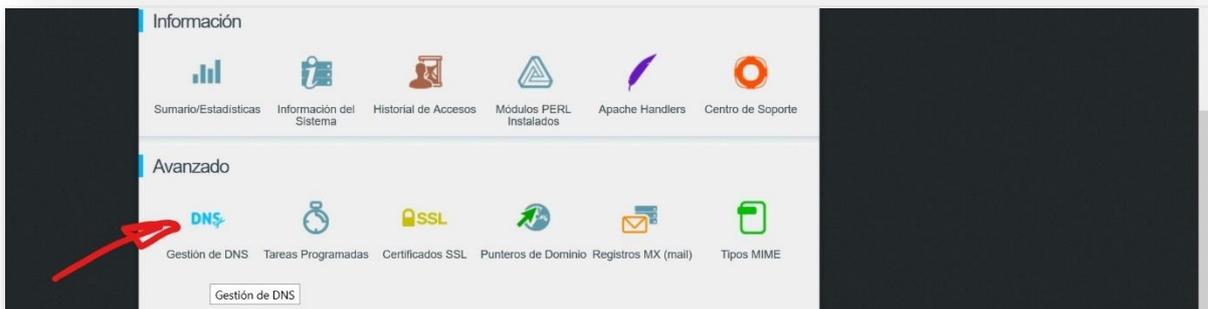


Ilustración 2.6

Hay que añadir un **registro tipo A** en la configuración de las DNS para que la URL **nube.jmsolistoro.net** apunte a la IP de la máquina recién configurada (se puede ver en el panel de control, *Ilustración 2.1*):



Ilustración 2.7

En el caso de un despliegue para una pequeña o mediana empresa, se elegiría el dominio propiedad de esa empresa para hacer esta configuración.

Con el sistema operativo ya instalado y las DNS configuradas ya podemos hacer *login* por SSH para continuar con la configuración (usando la URL si hemos configurado las DNS o directamente la dirección IP del servidor).

2.2. Instalación de prerequisites de *Nextcloud* 13.

Para configurar el servidor a partir de aquí usaremos una conexión de consola remota SSH, haciendo *login* con el usuario **root** y usando la clave que Kimsufi nos envía por correo electrónico tras completarse la instalación del sistema operativo.

Una vez hecho *login*, podemos empezar por comprobar que el nombre del servidor coincide con el que hemos asignado en las DNS:

```
root@nube:~# hostnamectl
  Static hostname: nube
    Icon name: computer-desktop
    Chassis: desktop
  Machine ID: c0484fa3e772e3ab78fcc01d5b044d48
  Boot ID: c473b416d48840178a6f10282034df38
  Operating System: Ubuntu 18.04 LTS
    Kernel: Linux 4.15.0-22-generic
  Architecture: x86-64
```

Vemos que el Static hostname corresponde al subdominio y si vemos el fichero de configuración `/etc/hosts`, debería aparecer el FQDN (*Fully Qualified Domain Name*, **nube.jmsolistoro.net**) y la dirección IP del servidor:

```
root@nube:~# cat /etc/hosts
127.0.0.1    localhost

# The following lines are desirable for IPv6 capable hosts
::1        ip6-localhost ip6-loopback
fe00::0    ip6-localnet
ff00::0    ip6-mcastprefix
ff02::1    ip6-allnodes
ff02::2    ip6-allrouters
ff02::3    ip6-allhosts
37.187.105.173  nube.jmsolistoro.net  nube
```

Después de esta comprobación podemos pasar a instalar los prerequisites para que *Nextcloud* funcione:

- Servidor web: Apache o Nginx.
- PHP: 7.0, 7.1 ó 7.2.
- Motor de base de datos: SQLite, MySQL, MariaDB o PostgreSQL.

Por sencillez he optado por la terna clásica Apache2, MySQL y PHP (versión 7.2). Para entornos empresariales más exigentes podríamos decantarnos por el servidor web Nginx y PostgreSQL, ya que ofrecen mayor rendimiento, aunque a costa de incrementar la complejidad de la instalación.

Para empezar instalaremos **tasksel**, que nos permitirá simplificar la instalación de Apache2, MySQL y PHP:

```
root@nube:~# apt-get install tasksetl
```

Ahora usamos **tasksetl** para instalar los paquetes requeridos para Apache2, MySQL y PHP:

```
root@nube:~# tasksetl install lamp-server
```

Durante la instalación se nos preguntará por la clave de superusuario (**root**) de MySQL. Es importante que escojamos una clave segura y la apuntemos, porque nos hará falta para configurar la base de datos que usará *Nextcloud*.

Debemos asegurarnos que están instalados todos los paquetes de PHP 7.2 que *Nextcloud* requiere, no solo los básicos, por lo que procederemos a instalar explícitamente todas las librerías requeridas:

```
root@nube:~# apt-get install libapache2-mod-php7.2 bzip2 php7.2-gd php7.2-json
php7.2-mysql php7.2-curl php7.2-mbstring php7.2-intl php-imagick php7.2-xml php7.2-
zip
```

El último prerrequisito que nos queda ya es crear la base de datos que usará *Nextcloud*:

```
root@nube:~# mysql -u root -p
```

Nos pedirá la clave de **root** de MySQL (en este caso, debido a la plantilla de instalación de *Kimsufi*, la clave **root** de MySQL es la misma clave que tiene el usuario **root** del servidor) y entraremos en una consola interactiva del motor del base de datos, desde la que podemos crear y configurar la base de datos que necesitamos:

```
mysql> CREATE DATABASE nextcloud;
Query OK, 1 row affected (0.00 sec)
mysql> CREATE USER 'nextcloud'@'localhost' IDENTIFIED BY 'LaClave22';
Query OK, 0 rows affected (0.00 sec)
mysql> GRANT ALL PRIVILEGES ON nextcloud.* TO 'nextcloud'@'localhost';
Query OK, 0 rows affected (0.00 sec)
mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)
mysql> quit
```

El resultado es que hemos creado una base de datos llamada **nextcloud** y un usuario llamado también **nextcloud**, con clave **LaClave22**, que tiene todos los permisos para operar en su base de datos. Nos hará falta cuando lancemos *Nextcloud* por primera vez y configuremos la conexión con la base de datos.

2.3. Configuración preliminar del servidor web.

Vamos a instalar *Nextcloud* en la siguiente carpeta del servidor:

- **/var/www/html/nube**

Para configurar adecuadamente el servidor web Apache, crearemos el archivo:

```
root@nube:~# nano /etc/apache2/sites-available/nube.conf
```

Que albergará la configuración necesaria para crear el subdominio `nube.jmsolistoro.net` con *Nextcloud*:

```
<VirtualHost *:80>
  ServerName nube.jmsolistoro.net
  ServerAdmin nube@jmsolistoro.net

  DocumentRoot /var/www/html/nube/

  <Directory /var/www/html/nube/>
    Options +FollowSymLinks
    AllowOverride All

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

    SetEnv HOME /var/www/html/nube
    SetEnv HTTP_HOME /var/www/html/nube

  </Directory>

  ErrorLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log

  # Possible values include: debug, info, notice, warn, error, crit,
  # alert, emerg.
  LogLevel warn

  CustomLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log combined
</VirtualHost>
```

Después de guardar el archivo, instalamos los módulos de Apache que *Nextcloud* necesita:

```
root@nube:~# a2enmod rewrite headers env dir mime
```

Activamos el *VirtualHost* *nube* y eliminamos el que está por defecto:

```
root@nube:~# a2ensite nube
root@nube:~# rm /etc/apache2/sites-enabled/000-default.conf
```

Y finalmente reiniciamos el servicio de Apache para que se apliquen los cambios:

```
root@nube:~# systemctl restart apache2
```

Antes de instalar y configurar *Nextcloud* solo nos quedaría configurar el cortafuegos para permitir la conexión al puerto 80 (http) y el 443 (https) del servidor web Apache (**y al puerto 22 de SSH para que no se corte la conexión remota y tengamos que empezar de nuevo instalando el sistema operativo**):

```
root@nube:~# ufw allow http
root@nube:~# ufw allow https
root@nube:~# ufw allow OpenSSH
```

Tras lo cual podemos activar el cortafuegos y comprobar los puertos abiertos:

```
root@nube:~# ufw enable
root@nube:~# ufw status numbered
Status: active

      To Action      From
      --  -
[ 1] 80/tcp    ALLOW IN    Anywhere
[ 2] 443/tcp    ALLOW IN    Anywhere
[ 3] OpenSSH   ALLOW IN    Anywhere
[ 4] 80/tcp (v6) ALLOW IN    Anywhere (v6)
[ 5] 443/tcp (v6) ALLOW IN    Anywhere (v6)
[ 6] OpenSSH (v6) ALLOW IN    Anywhere (v6)
```

El siguiente paso es la instalación y configuración de *Nextcloud*.

3. Instalación de Nextcloud.

3.1. Instalación del núcleo de Nextcloud.

Vamos a descargar y descomprimir la última versión de *Nextcloud* (la versión número 13 en el momento de escribir esta memoria) en la carpeta ya configurada del servidor web Apache:

```
root@nube:~# cd /var/www/html/
root@nube:/var/www/html# wget https://download.Nextcloud.com/server/releases/latest-13.tar.bz2 -O Nextcloud-13-latest.tar.bz2
root@nube:/var/www/html# tar -xvzf Nextcloud-13-latest.tar.bz2
root@nube:/var/www/html# mv Nextcloud nube
root@nube:/var/www/html# chown -R www-data:www-data nube
root@nube:/var/www/html# rm Nextcloud-13-latest.tar.bz2
```

A partir de aquí la configuración se realiza desde el navegador web. Abriendo la url:

- <http://nube.jmsolistoro.net>

Llegaremos a esta pantalla:

Crear una cuenta de administrador

Nombre de usuario 1

Contraseña 2

Almacenamiento y base de datos

Directorio de datos

/var/www/html/nube/data 3

Configurar la base de datos

Solo MySQL/MariaDB está disponible. Instalar y activar módulos PHP adicionales para elegir otros formatos de base de datos. Para más detalles revisar la documentación. ↗

Usuario de la base de datos

Contraseña de la base de datos

Nombre de la base de datos: localhost

Por favor especifique el número del puerto junto al nombre del anfitrión (p.e., localhost:5432).

Completar la instalación

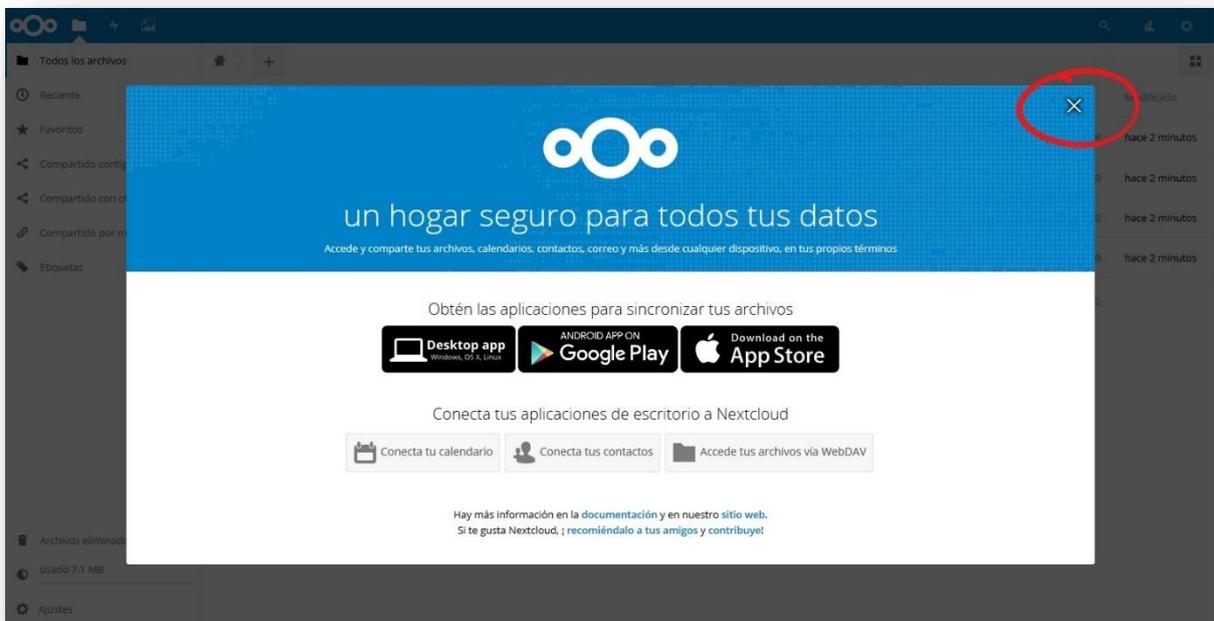
¿Necesita ayuda? Vea la documentación.

Nextcloud – un hogar seguro para todos tus datos

Los campos a rellenar, de arriba abajo, son:

1. Nombre de usuario administrador: **admin** (como sugerencia para el usuario administrador).
2. Clave de usuario administrador: **Admin343** (por ejemplo).
3. Carpeta de datos: **/var/www/html/nube/data** (debería estar puesto, dejar como está).
4. Usuario de la base de datos: **nextcloud**.
5. Clave del usuario de la base de datos: **LaClave22** (la que se puso al crear el usuario de la base de datos).
6. Nombre de la base de datos: **nextcloud**.
7. Anfitrión del motor de base de datos: **localhost** (debería estar puesto, dejar como está).

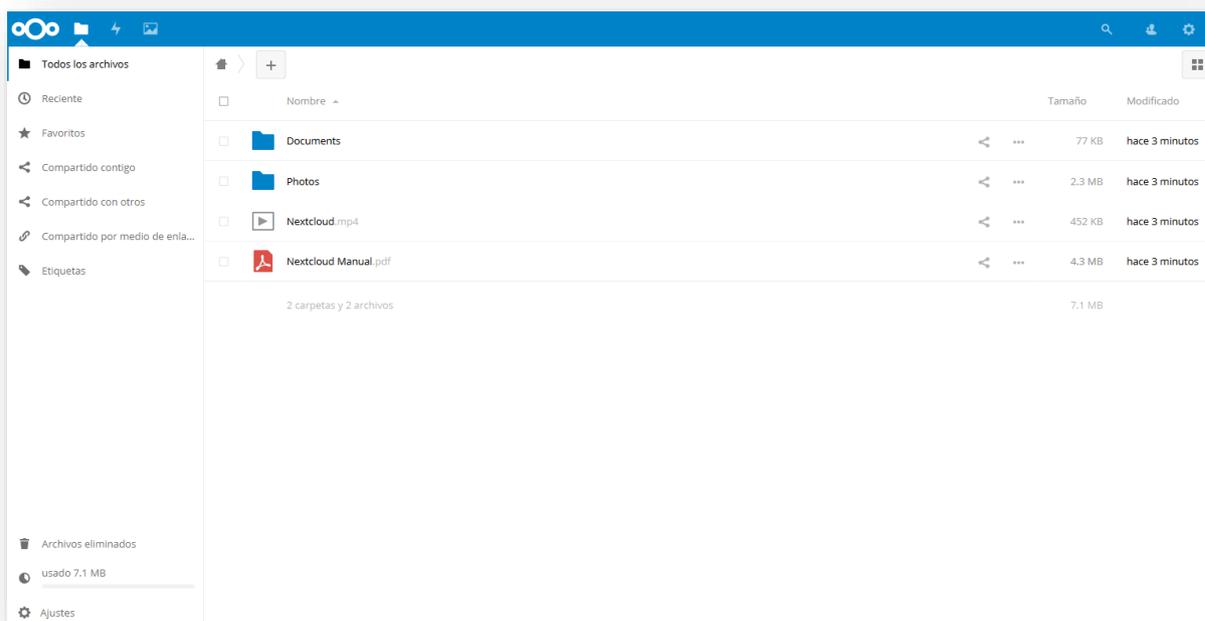
Tras rellenar todos los campos correctamente y pulsar el botón **Completar la instalación**, el servidor tardará unos momentos en completar la instalación, tras la cual seremos redireccionados a la pantalla de bienvenida de *Nextcloud*:



En esta pantalla se nos invita a descargar el cliente de escritorio de *Nextcloud* (que veremos más adelante en esta misma memoria), a conectar nuestros calendarios y contactos con nuestra nueva cuenta, y a acceder a nuestros archivos mediante el protocolo *WebDav*.

La sincronización de calendario y contactos no funcionará en estos momentos porque las aplicaciones correspondientes de *Nextcloud* no están instaladas. Trataremos la instalación y configuración de aplicaciones de *Nextcloud* más adelante.

Aquí simplemente cerraremos ventana en primer plano pulsando en la X de arriba a la derecha para acceder a la gestión de ficheros de nuestra recién creada cuenta:



No es el momento todavía de usar nuestra recién creada nube de intercambio de ficheros (y potencialmente mucho más), pues todas las conexiones que realicemos con el servidor están sin cifrar, con el grave riesgo que ello supone para la seguridad, privacidad e integridad de nuestros datos.

3.2. Configuración de *Nextcloud* y el servidor web para la conexión cifrada.

Es muy importante que en un sistema corporativo de intercambio de ficheros y trabajo en grupo todas las comunicaciones estén cifradas, para así asegurar la privacidad de las comunicaciones entre el servidor y los clientes en todas las actividades: usuarios, claves, ficheros, y todo tipo de información que pueda pertenecer a la empresa o a la esfera personal de los usuarios.

Ello se consigue configurando el servidor para que:

1. Se identifique mediante un certificado adecuado expedido por una autoridad certificadora autorizada y reconocida por los clientes. En nuestro caso usaremos los certificados gratuitos de la fundación sin ánimo de lucro [Let's Encrypt](#).
2. Obligue a que todas las consultas se realicen mediante conexiones cifradas TLS, redireccionando todas aquellas peticiones sin cifrar (**http**) al canal cifrado (**https**).

Para la obtención e instalación de los certificados usaremos software desarrollado por [Certbot](#):

1. Añadimos los repositorios de **Certbot** para Ubuntu 18.04, con lo que obtendremos la última versión, y lo instalamos:

```
root@nube:~# add-apt-repository ppa:certbot/certbot
root@nube:~# apt update
root@nube:~# apt upgrade
```

```
root@nube:~# apt install python-certbot-apache
```

- Para obtener el certificado ejecutaremos **Certbot** con la URL de nuestro servidor como argumento, siendo bueno que escribamos un correo electrónico para que nos avisen de las renovaciones cuando nos lo pida, y elegir que nos haga la redirección automática de **http a https** (opción 2):

```
root@nube:~# certbot --apache -d nube.jmsolistoro.net
Saving debug log to /var/log/letsencrypt/letsencrypt.log
Plugins selected: Authenticator apache, Installer apache
Enter email address (used for urgent renewal and security notices) (Enter 'c' to
cancel): nube@jmsolistoro.net

-----
Please read the Terms of Service at
https://letsencrypt.org/documents/LE-SA-v1.2-November-15-2017.pdf. You must
agree in order to register with the ACME server at
https://acme-v01.api.letsencrypt.org/directory
-----
(A)gree/(C)ancel: A

-----
Would you be willing to share your email address with the Electronic Frontier
Foundation, a founding partner of the Let's Encrypt project and the non-profit
organization that develops Certbot? We'd like to send you email about EFF and
our work to encrypt the web, protect its users and defend digital rights.
-----
(Y)es/(N)o: Y
Obtaining a new certificate
Performing the following challenges:
http-01 challenge for nube.jmsolistoro.net
Waiting for verification...
Cleaning up challenges
Created an SSL vhost at /etc/apache2/sites-available/nube-le-ssl.conf
Enabled Apache socache_shmcb module
Enabled Apache ssl module
Deploying Certificate to VirtualHost /etc/apache2/sites-available/nube-le-ssl.conf
Enabling available site: /etc/apache2/sites-available/nube-le-ssl.conf

Please choose whether or not to redirect http traffic to HTTPS, removing http
access.
-----
1: No redirect - Make no further changes to the webserver configuration.
2: Redirect - Make all requests redirect to secure https access. Choose this for
new sites, or if you're confident your site works on HTTPS. You can undo this
change by editing your web server's configuration.
-----
Select the appropriate number [1-2] then [enter] (press 'c' to cancel): 2
Redirecting vhost in /etc/apache2/sites-enabled/nube.conf to ssl vhost in
/etc/apache2/sites-available/nube-le-ssl.conf

-----
Congratulations! You have successfully enabled https://nube.jmsolistoro.net

You should test your configuration at:
https://www.ssllabs.com/ssltest/analyze.html?d=nube.jmsolistoro.net
-----

IMPORTANT NOTES:
- Congratulations! Your certificate and chain have been saved at:
  /etc/letsencrypt/live/nube.jmsolistoro.net/fullchain.pem
  Your key file has been saved at:
  /etc/letsencrypt/live/nube.jmsolistoro.net/privkey.pem
  Your cert will expire on 2018-08-23. To obtain a new or tweaked
  version of this certificate in the future, simply run certbot again
  with the "certonly" option. To non-interactively renew *all* of
  your certificates, run "certbot renew"
- Your account credentials have been saved in your Certbot
  configuration directory at /etc/letsencrypt. You should make a
  secure backup of this folder now. This configuration directory will
  also contain certificates and private keys obtained by Certbot so
  making regular backups of this folder is ideal.
- If you like Certbot, please consider supporting our work by:

  Donating to ISRG / Let's Encrypt: https://letsencrypt.org/donate
  Donating to EFF: https://eff.org/donate-le
```

Con un servidor limpio, recién instalado y configurado según los pasos de esta memoria, la configuración automática de ha funcionado perfectamente. Como nos sugiere **Certbot**, podemos

comprobar la configuración de la conexión **https** a nuestro servidor usando el servicio de [SSL Labs](https://www.ssllabs.com/sslltest/analyze.html?d=nube.jmsolistoro.net), poniendo <https://www.ssllabs.com/sslltest/analyze.html?d=nube.jmsolistoro.net> en el navegador:

SSL Report: nube.jmsolistoro.net (37.187.105.173)

Assessed on: Fri, 25 May 2018 10:52:00 UTC | [Hide](#) | [Clear cache](#)

[Scan Another »](#)

Summary

Overall Rating



Category	Score
Certificate	100
Protocol Support	95
Key Exchange	90
Cipher Strength	90

Visit our [documentation page](#) for more information, configuration guides, and books. Known issues are documented [here](#).

Certificate #1: RSA 2048 bits (SHA256withRSA)

Server Key and Certificate #1 ↓

Subject	nube.jmsolistoro.net Fingerprint: SHA256: 5c16df945469c128b1a30d270cf9a380e60b6056c4292d299e55293f1f4330f Pin: SHA256: LKglkXrF SaP 8gDMs6C4Q6lYj Rv8e2JRYbOzRE4QJv58=
Common names	nube.jmsolistoro.net
Alternative names	nube.jmsolistoro.net
Serial Number	04825275723ba6eca6381c1ca4a6e77d9fa7
Valid from	Fri, 25 May 2018 09:49:00 UTC
Valid until	Thu, 23 Aug 2018 09:49:00 UTC (expires in 2 months and 28 days)
Key	RSA 2048 bits (e 65537)
Weak Key (Debian)	No
Issuer	Let's Encrypt Authority X3 AIA: http://cert.int-x3.letsencrypt.org/
Signature algorithm	SHA256withRSA
Extended Validation	No
Certificate Transparency	Yes (certificate)
OC SP Must Staple	No
Revocation information	OCSP OCSP: http://ocsp.int-x3.letsencrypt.org
Revocation status	Good (not revoked)
DN S CAA	No (more info)
Trusted	Yes Mozilla Apple Android Java Windows

Additional Certificates (if supplied) ↓

Certificates provided	2 (2733 bytes)
Chain issues	None
#2	Let's Encrypt Authority X3 Fingerprint: SHA256: 25847d668eb4f04fd40b12b6b0740c567da7d024309eb6c2c90fe41d9de218d Pin: SHA256: YLh1dJRy6Kja30RiAn7JKnbQGiuEtLMkBgFF2FuHg=
Valid until	Wed, 17 Mar 2021 16:40:46 UTC (expires in 2 years and 9 months)
Key	RSA 2048 bits (e 65537)
Issuer	DST Root CA X3
Signature algorithm	SHA256withRSA

Certification Paths +

[Click here to expand](#)

Hemos obtenido una buena nota (A), señal de que la configuración SSL es correcta. Además, **Certbot** nos ha dejado configurada la redirección **obligatoria** de *http* a *https* en todas las peticiones dirigidas al dominio **nube.jmsolistoro.net**, pues no queremos que nuestro servidor establezca ninguna conexión que no sea cifrada.

Aunque todo haya funcionado a la primera y el resultado obtenido sea satisfactorio, es conveniente que sepamos qué es lo que ha hecho **Certbot** con la configuración de nuestro servidor web:

1. Ha pedido a los servidores de **Let's Encrypt** que comprueben que la URL <http://nube.jmsolistoro.net/> realmente corresponde a nuestro servidor. Se hace mediante la comprobación de *tokens* que pone en la carpeta raíz del dominio para el que estamos pidiendo los certificados. Cuando la comprobación (satisfactoria o no) se realiza, los *tokens* son borrados del servidor.
2. Si la comprobación es satisfactoria, se guardan los certificados proporcionados por **Let's Encrypt** en la ruta **/etc/letsencrypt/live/nube.jmsolistoro.net/** y crea un nuevo Virtual Host llamado **/etc/apache2/sites-available/nube-le-ssl.conf** que escucha en el puerto 443 y que solo admite conexiones cifradas. El contenido de ese archivo de configuración es el mismo que el de nuestro **Virtual Host** original con algunos cambios importantes (marcados en **rojo**):

```
<IfModule mod_ssl.c>
<VirtualHost *:443>

    ServerName nube.jmsolistoro.net
    ServerAdmin nube@jmsolistoro.net

    DocumentRoot /var/www/html/nube/

    <Directory /var/www/html/nube/>
        Options +FollowSymLinks
        AllowOverride All

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

    SetEnv HOME /var/www/html/nube
    SetEnv HTTP_HOME /var/www/html/nube

    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log combined

    SSLCertificateFile /etc/letsencrypt/live/nube.jmsolistoro.net/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/nube.jmsolistoro.net/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf

</VirtualHost>
</IfModule>
```

Vemos que además de cambiar el puerto 80 por el 443 (TLS), se ha añadido la configuración relativa a los certificados que acabamos de obtener.

Si hubiésemos elegido configurar la redirección obligatoria por nosotros mismos, lo que habría que hacer es editar el archivo **/etc/apache2/sites-available/nube.conf** (que corresponde al **Virtual Host** que escucha en el puerto 80 -estándar *http*-) y añadir unas líneas (destacadas en color **rojo**):

```

<VirtualHost *:80>

    ServerName nube.jmsolistoro.net
    ServerAdmin nube@jmsolistoro.net

    DocumentRoot /var/www/html/nube/

    <Directory /var/www/html/nube/>
        Options +FollowSymLinks
        AllowOverride All

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

    SetEnv HOME /var/www/html/nube
    SetEnv HTTP_HOME /var/www/html/nube

    </Directory>

    ErrorLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log combined

    RewriteEngine on
    RewriteCond %{SERVER_NAME} =nube.jmsolistoro.net
    RewriteRule ^ https://%{SERVER_NAME}%{REQUEST_URI} [END,NE,R=permanent]

</VirtualHost>

```

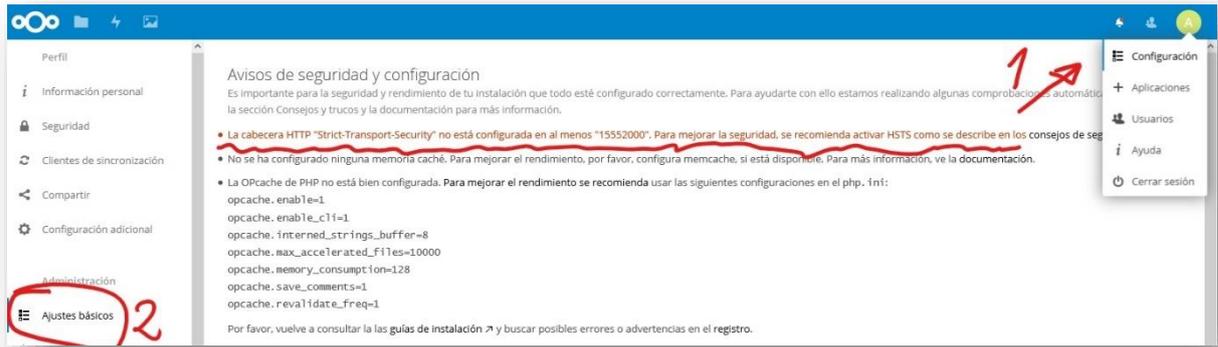
La redirección solo afecta al (sub)dominio ***http://nube.jmsolistoro.net/***, lo que asegura que todas las conexiones que hagamos con *Nextcloud* serán cifradas. Además, como no hay especificado ningún otro (sub)dominio en la configuración de *Apache*, de facto todas las conexiones web que se hagan con nuestro servidor serán cifradas.

3.3. Auditoría de seguridad interna de *Nextcloud*.

Dentro del interfaz de configuración de *Nextcloud* como administrador, en la sección de ajustes básicos, hay un apartado llamado ***Avisos de seguridad y configuración***, en el que muestra el resultado de algunas comprobaciones automáticas sobre seguridad y rendimiento.

Llegados a este paso, la única recomendación de seguridad que nos hace *Nextcloud* está relacionada con la configuración del servidor web Apache (todos los demás requisitos los hemos satisfecho ya):

- *La cabecera HTTP "Strict-Transport-Security" no está configurada en al menos "15552000". Para mejorar la seguridad, se recomienda activar HTTP Strict-Transport-Security.*



Para cumplir con ese requisito debemos volver a editar la configuración del Virtual Host https, que está en `/etc/apache2/sites-available/nube-le-ssl.conf` y añadir 3 líneas más (destacadas en rojo):

```
<IfModule mod_ssl.c>
<VirtualHost *:443>

    ServerName nube.jmsolistoro.net
    ServerAdmin nube@jmsolistoro.net

    DocumentRoot /var/www/html/nube/

    <Directory /var/www/html/nube/>
        Options +FollowSymLinks
        AllowOverride All

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

    SetEnv HOME /var/www/html/nube
    SetEnv HTTP_HOME /var/www/html/nube

</Directory>

    ErrorLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log

    # Possible values include: debug, info, notice, warn, error, crit,
    # alert, emerg.
    LogLevel warn

    CustomLog ${APACHE_LOG_DIR}/nube.jmsolistoro.net.log combined

    SSLCertificateFile /etc/letsencrypt/live/nube.jmsolistoro.net/fullchain.pem
    SSLCertificateKeyFile /etc/letsencrypt/live/nube.jmsolistoro.net/privkey.pem
    Include /etc/letsencrypt/options-ssl-apache.conf

    <IfModule mod_headers.c>
        Header always set Strict-Transport-Security "max-age=15768000;
        includeSubDomains; preload"
    </IfModule>

</VirtualHost>
</IfModule>
```

Otra medida de seguridad importante, y que no viene en los **Avisos de seguridad y configuración** de Nextcloud es el bloqueo de ataques de fuerza bruta mediante el servicio **fail2ban**.

1. Vamos a instalarlo para después configurarlo:

```
root@nube:~# apt update -y
root@nube:~# apt install fail2ban
```

2. **Fail2ban** no tiene soporte para *Nextcloud* por defecto, pero podemos crear un filtro para que reconozca las *IPs* que fallan en la autenticación que aparecen en los logs de *Nextcloud*. Para ello debemos crear el fichero `/etc/fail2ban/filter.d/nextcloud.conf` y añadirle las siguientes líneas:

```
[Definition]
failregex=^{"reqId":".*","remoteAddr":".*","app":"core","message":"Login failed:
'.*' \(\(Remote IP: '<HOST>'\)\)","level":2,"time":".*"}$
^{"reqId":".*","level":2,"time":".*","remoteAddr":".*","app":"core.*","message":"Lo
gin failed: '.*' \(\(Remote IP: '<HOST>'\)\)".*}$
^.*"remoteAddr":\ "<HOST>"\.*Trusted domain error.*$
```

3. Crearemos un fichero de configuración adicional `/etc/fail2ban/jail.d/jail.local` al que añadiremos las siguientes líneas (que incluyen reglas para proteger a **Apache** y al **servidor de SSH**, además de al propio *Nextcloud*):

```
[apache]
enabled = true
port = http,https
filter = apache-auth
logpath = /var/log/apache2/*error.log
maxretry = 3
bantime = 1200

[apache-overflows]
enabled = true
port = http,https
filter = apache-overflows
logpath = /var/log/apache2/*error.log
maxretry = 3
bantime = 1200

[apache-noscript]
enabled = true
port = http,https
filter = apache-noscript
logpath = /var/log/apache2/*error.log
maxretry = 3
bantime = 1200

[apache-badbots]
enabled = true
port = http,https
filter = apache-badbots
logpath = /var/log/apache2/*error.log
maxretry = 3
bantime = 1200

[ssh]
enabled = true
port = ssh
filter = sshd
logpath = /var/log/auth.log
maxretry = 3
bantime = 1200

[nextcloud]
backend = auto
enabled = true
port = 80,443
protocol = tcp
filter = nextcloud
maxretry = 4
bantime = 36000
findtime = 36000
logpath = /var/www/html/nube/data/nextcloud.log
```

4. Reiniciamos el servicio **fail2ban** para que se cargue la nueva configuración:

```
root@nube:~# systemctl restart fail2ban
```

Nos quedan algunas configuraciones de rendimiento, relacionadas con cachés de PHP y el CRON para realizar tareas de mantenimiento automáticas, que trataremos en el siguiente punto.

3.4. Optimización del rendimiento.

Para optimizar el rendimiento en una instalación para pequeñas y medianas empresas se recomienda usar **Redis file locking cache** y **APCu memory cache** (además de configurar correctamente el **Opcache** de **PHP** para las necesidades de **Nextcloud**). Para dejar **Opcache**, **Redis** y **APCu** configurados y funcionando en nuestro servidor debemos proceder como sigue:

1. Instalación de paquetes adicionales desde consola (por si no estaban instalados ya):

```
root@nube:~# apt install php-apcu redis-server php-redis php7.2-bz2 php7.2-ldap
imagemagick php-smbclient
```

2. Editar el fichero **/etc/php/7.2/apache2/php.ini** para hacer los siguientes cambios que recomienda **Nextcloud** (principalmente consiste en *descomentar* las opciones correspondientes eliminando el **;** al principio de las líneas en el fichero de configuración, asegurándonos de que los valores de dichas opciones son los requeridos):

```
opcache.enable=1
opcache.enable_cli=1
opcache.memory_consumption=128
opcache.interned_strings_buffer=8
opcache.max_accelerated_files=10000
opcache.revalidate_freq=1
opcache.save_comments=1
```

3. Nos aseguramos de que el caché **apcu** está activado también para **cli** editando el fichero de configuración **/etc/php/7.2/mods-available/apcu.ini** y asegurándonos que contiene las siguientes dos líneas:

```
extension=apcu.so
apc.enable_cli=1
```

4. Modificar la configuración por defecto de **Redis**, cambiando las siguientes líneas del fichero **/etc/redis/redis.conf** (cambios en **rojo**):

```
.
.
.
# Accept connections on the specified port, default is 6379 (IANA #815344).
# If port 0 is specified Redis will not listen on a TCP socket.
port 0
.
.
.
# Unix socket.
#
# Specify the path for the Unix socket that will be used to listen for
# incoming connections. There is no default, so Redis will not listen
# on a unix socket when not specified.
#
unixsocket /var/run/redis/redis-server.sock
unixsocketperm 770
.
.
.
```

5. Añadir el usuario de **Apache**, **www-data**, al grupo **redis**:

```
root@nube:~# usermod -a -G redis www-data
```

6. Reiniciamos los servicios de **Apache** y **Redis**:

```
root@nube:~# systemctl restart apache2
root@nube:~# systemctl restart redis-server
```

7. Modificamos la configuración de *Nextcloud* en `/var/www/html/nube/config/config.php` añadiendo lo que aparece en **rojo**:

```
<?php
$CONFIG = array (
  'instanceid' => 'oc8tkrb5i1mq',
  'passwordsalt' => 'ZboE18GEMAF12hvMo9K+jVc110vU9s',
  'secret' => 'I9a30YkwuKoSp3FpHIIAqvBHQGuGK7vKRp0Zxm9oc0Eg+FWO',
  'trusted_domains' =>
  array (
    0 => 'nube.jmsolistoro.net',
  ),
  'datadirectory' => '/var/www/html/nube/data',
  'overwrite.cli.url' => 'http://nube.jmsolistoro.net',
  'dbtype' => 'mysql',
  'version' => '13.0.2.1',
  'dbname' => 'nextcloud',
  'dbhost' => 'localhost',
  'dbport' => '',
  'dbtableprefix' => 'oc_',
  'mysql.utf8mb4' => true,
  'dbuser' => 'nextcloud',
  'dbpassword' => 'LaClave22',
  'installed' => true,

  'memcache.local' => '\OC\Memcache\APCu',
  'memcache.locking' => '\OC\Memcache\Redis',
  'filelocking.enabled' => 'true',
  'redis' =>
  array (
    'host' => '/var/run/redis/redis-server.sock',
    'port' => 0,
    'timeout' => 0.0,
  ),
);
```

8. Nos aseguramos de que el servidor **Redis** vuelva a ejecutarse si la máquina se reinicia:

```
root@nube:~# systemctl enable redis-server
```

Con esto ya tenemos configurada la optimización del servidor web (Apache + PHP) para un despliegue de un solo servidor para una pequeña o mediana empresa.

Como último detalle vamos a cambiar la configuración de los **Trabajos en segundo plano** para usar el **Cron** del sistema en vez **AJAX**, que solo ejecuta peticiones cada vez que alguien carga la web. Para ello vamos a crear una tarea en el **cron** del sistema mediante el usuario **root**, y que cada 15 minutos lance las tareas de mantenimiento de *Nextcloud*:

1. Creamos una *script* de *bash*, de nombre `nextcloud-cron.sh` por ejemplo, en la carpeta del usuario **root**:

```
root@nube:~# mkdir scripts
root@nube:~# nano /root/scripts/Nextcloud-cron.sh
```

2. El contenido del *script* será:

```
#!/bin/bash
sudo -H -u www-data php -d memory_limit=512M -f /var/www/html/nube/cron.php
```

3. Después de grabar el *script* debemos hacerlo ejecutable:

```
root@nube:~# chmod +x /root/scripts/Nextcloud-cron.sh
```

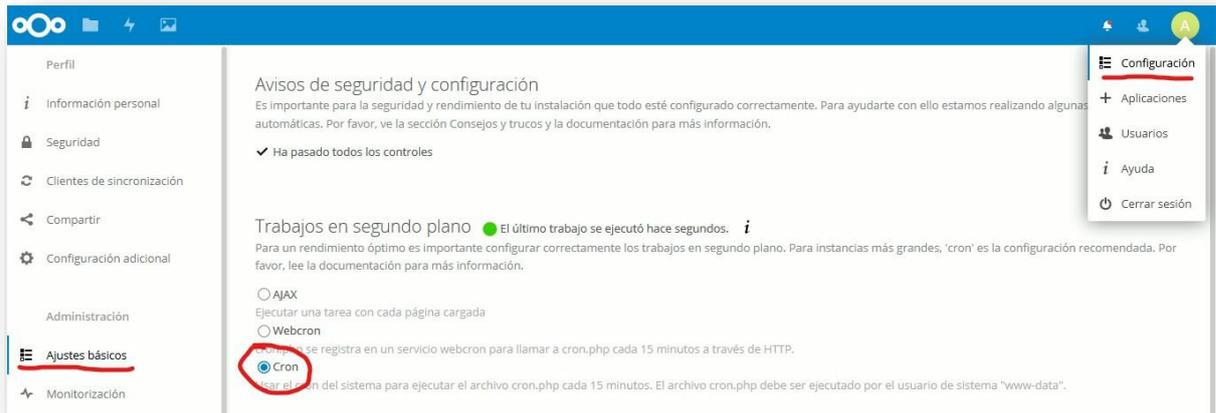
4. Debemos añadir la ejecución del *script* al *cron*:

```
root@nube:~# crontab -e
```

5. Y añadiremos estas dos líneas al final:

```
# NEXTCLOUD cronjob cada 15 minutos  
*/15 * * * * /root/scripts/nextcloud-cron.sh
```

Por último, cambiamos la configuración en *Nextcloud* para seleccionar **Cron**:



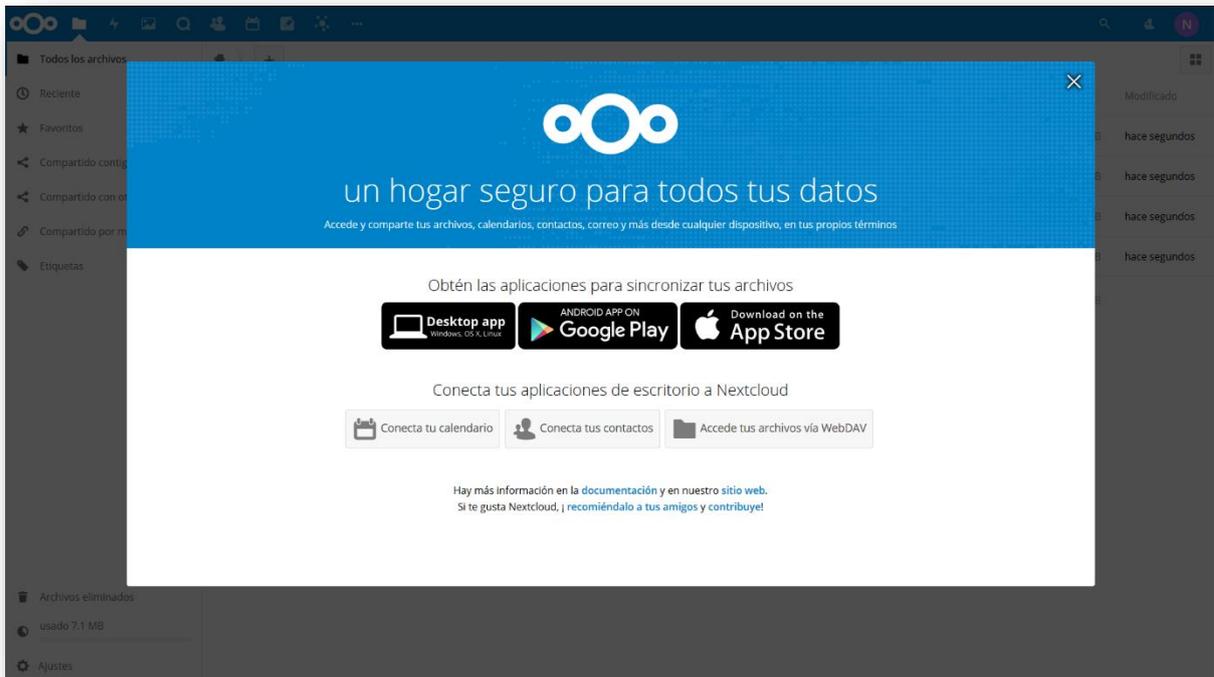
Con todos los cambios que hemos hecho en la configuración de Apache y PHP no está de más que nos aseguremos que dichos cambios se aplican haciendo una última recarga del servicio **Apache**:

```
root@nube:~# systemctl restart apache2
```

4. Clientes de sincronización.

4.1. Cliente de sincronización de escritorio.

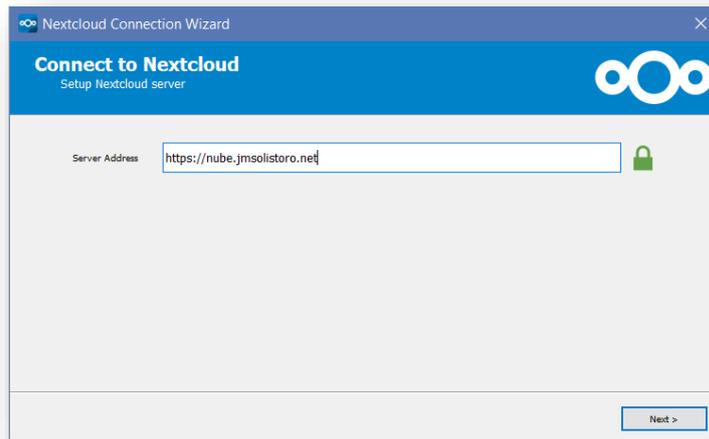
La primera vez que entremos en una cuenta recién creada de *Nextcloud*, el sistema nos recomendará descargarnos el cliente de escritorio:



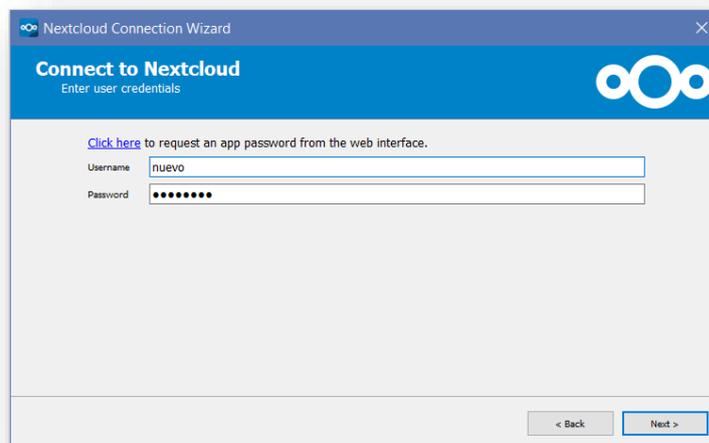
Es muy recomendable instalar el cliente de escritorio de *Nextcloud*, [que puede descargarse de su página web oficial](#). Ello nos permitirá sincronizar, de manera automática, los archivos que tengamos en nuestra cuenta de *Nextcloud* en una carpeta de nuestro ordenador portátil o sobremesa (Windows, Mac y Linux) que seleccionaremos durante la instalación. Aunque existe también un cliente de sincronización para móviles, su uso será tratado en la siguiente sección.

Una vez instalado el cliente de sincronización, al ejecutarlo por primera vez nos pedirá configurar una cuenta, dando los siguientes pasos:

1. Dirección del servidor: <https://nube.jmsolistoro.net> en nuestro caso. Nótese que el protocolo seleccionado es **https**, proporcionando una conexión cifrada y segura entre el servidor y cliente.

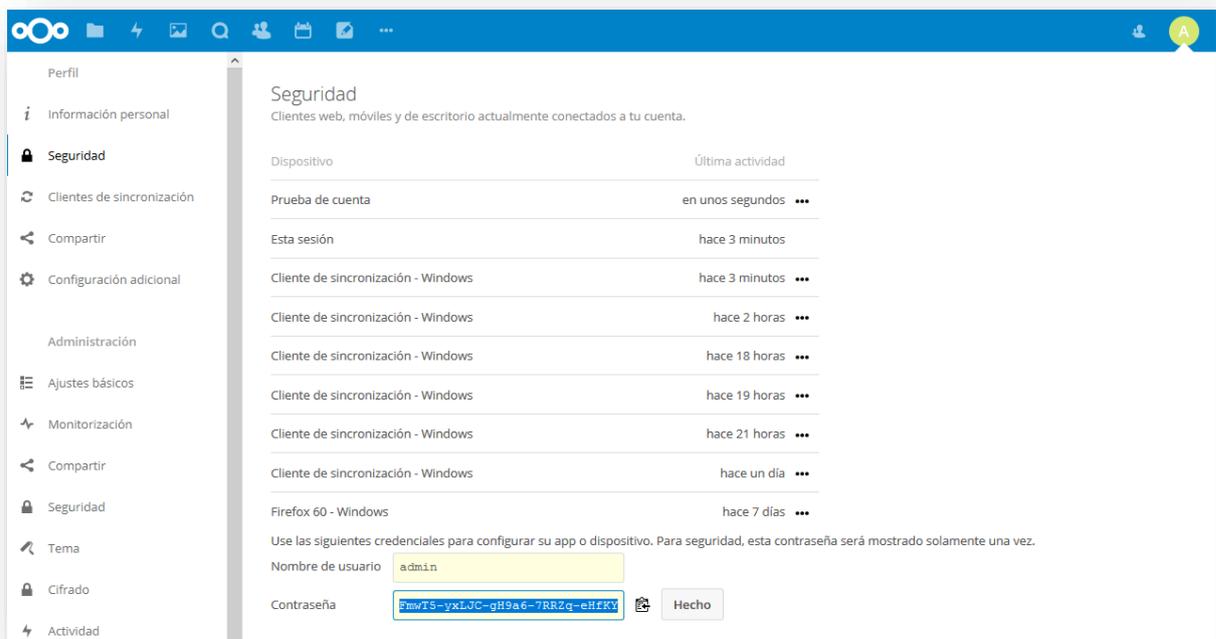
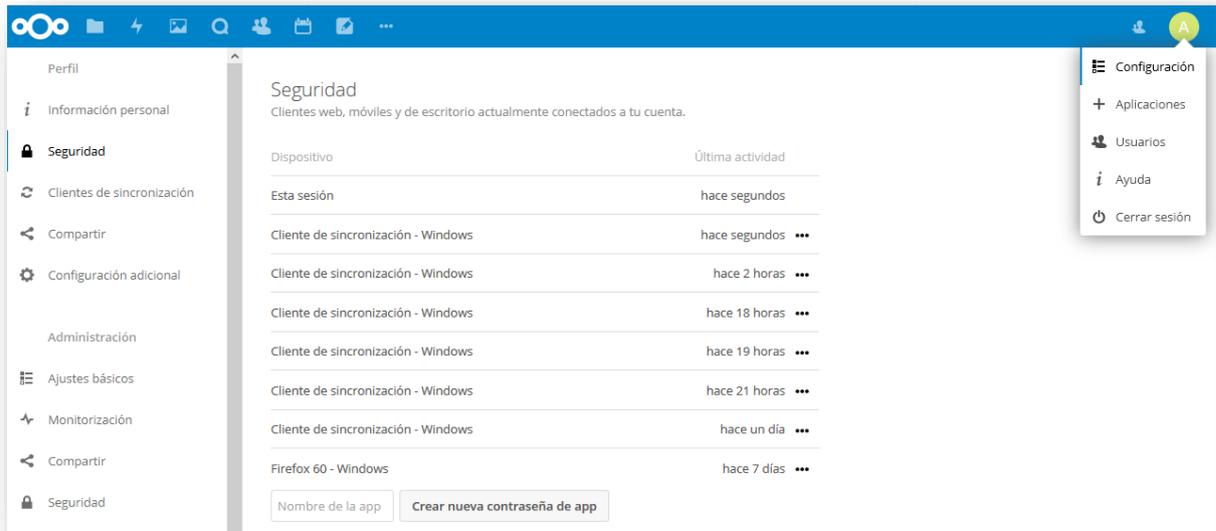


2. En el siguiente paso deberemos introducir el usuario y su clave. También es posible definir una clave de aplicación desde la interfaz web de Nextcloud (el cliente nos proporciona un enlace en este paso para hacerlo). Esto significa que para esta instalación usamos una clave distinta, generada desde el propio *Nextcloud*, y no la clave original de nuestro usuario. Así, en futuro, tendremos la posibilidad de revocar el acceso al cliente de esta instalación sin tener que cambiar nuestra clave de usuario “maestra” y volver a configurar todos los clientes en los demás dispositivos en los que tengamos nuestra cuenta sincronizada.



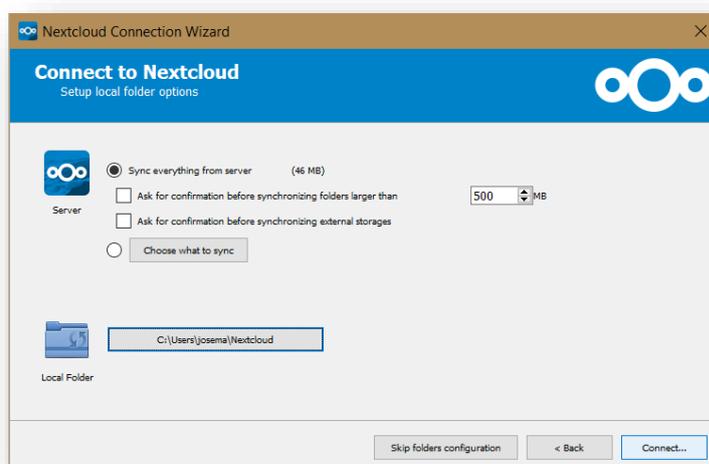
3. Si hemos optado por crear una clave de aplicación específica para esta instalación, que es lo más recomendable, el cliente de instalación nos abrirá en el navegador una página de acceso al servidor de *Nextcloud* que hemos puesto en el primer paso, en el que pondremos nuestros datos de acceso y nos redirigirá al menú de **Configuración de Seguridad** de nuestra cuenta, donde podremos crear la clave de aplicación específica y revocar las que tengamos creadas. Pondremos un nombre descriptivo de la instalación que estamos realizando y pulsamos el botón **Crear nueva contraseña de app**. Después de generar la clave de aplicación podremos copiarla e introducirla en la instalación.

4. Clientes de sincronización.



- Después de conectar al servidor con el usuario y contraseña proporcionados, en esta pantalla podremos configurar los últimos detalles de la sincronización, como sincronizar selectivamente solo algunas carpetas del servidor en vez de hacerlo con la cuenta completa, y seleccionar la carpeta local donde se guardarán los datos sincronizados. La sincronización selectiva es posible cambiarla en cualquier momento desde la configuración del cliente, pero no la carpeta local de sincronización. Si quisiéramos cambiar de sitio dicha carpeta no tendríamos más remedio que eliminar la cuenta (esto no elimina la carpeta local) y volver a repetir el proceso de añadir una cuenta al cliente de sincronización seleccionando al final la nueva ubicación. Si movemos los archivos de la carpeta local a la nueva ubicación antes de configurar la nueva cuenta, el cliente de

sincronización de Nextcloud lo detectará y no descargará los archivos que ya estén sincronizados con el servidor.



4.2. Clientes de sincronización para dispositivos móviles.

Nextcloud dispone de clientes de sincronización en dispositivos móviles para Android e iOS, disponibles a través de sus respectivas tiendas de aplicaciones. Aunque la configuración de estos clientes es igual que en los clientes de sobremesa (conexión al servidor, usuario, clave, etc.), su funcionamiento es ligeramente distinto: por defecto no se realiza sincronización alguna, sino que simplemente nos muestra los archivos que tenemos en nuestra cuenta, necesitando wifi u otra conexión de datos para que funcione.

Entre las funciones que incluye el cliente móvil, podemos destacar:

1. Operaciones con archivos: copiar, mover, borrar, abrir previa descarga al dispositivo móvil, etc.
2. Subir archivos a nuestra cuenta a través de la acción estándar de *Compartir* de nuestro dispositivo móvil.
3. Configurar una carpeta de nuestro dispositivo móvil para que automáticamente se suba su contenido a nuestra cuenta de *Nextcloud*.
4. Marcar archivos y carpetas de nuestra cuenta de *Nextcloud* como *Disponible sin conexión*, lo que hará que sean descargados a nuestro móvil para que estén accesibles aunque no tengamos conexión a Internet en ese momento.
5. Sincronizar calendarios entre nuestro móvil y nuestra cuenta de *Nextcloud*.
6. Sincronizar y hacer copia de seguridad de los contactos de nuestro dispositivo móvil.
7. Si nuestro móvil lo soporta, configurar el uso de la huella dactilar para acceder a la aplicación móvil de *Nextcloud*.

5. Diseño modular de Nextcloud. Aplicaciones oficiales adicionales y de terceros.

Nextcloud se instala con un núcleo base de funcionalidades que pueden expandirse instalando nuevas aplicaciones, tanto oficiales como de terceros. Por conveniencia y claridad, *Nextcloud*, organiza algunas de las más importantes en lotes temáticos:

1. **Para empresas:** Auditoría y bitácora, *Backend* de usuario y grupos por *LDAP*, Políticas de retención de archivos, Etiquetado automático de archivos, Inicio de sesión único (*SSO*) mediante *SAML*, Control de acceso a ficheros.
2. **Trabajo en grupo y productividad:** Calendario, Contactos, Videoconferencia *Talk*, Tareas y Notas.
3. **Compartición en redes sociales:** Compartir por Twitter, Google+, Facebook, correo electrónico y Diaspora.
4. **Educación:** Círculos, Carpetas de grupos, Centro de avisos, Notificaciones del Administrador y Aviso de límites de cuota.

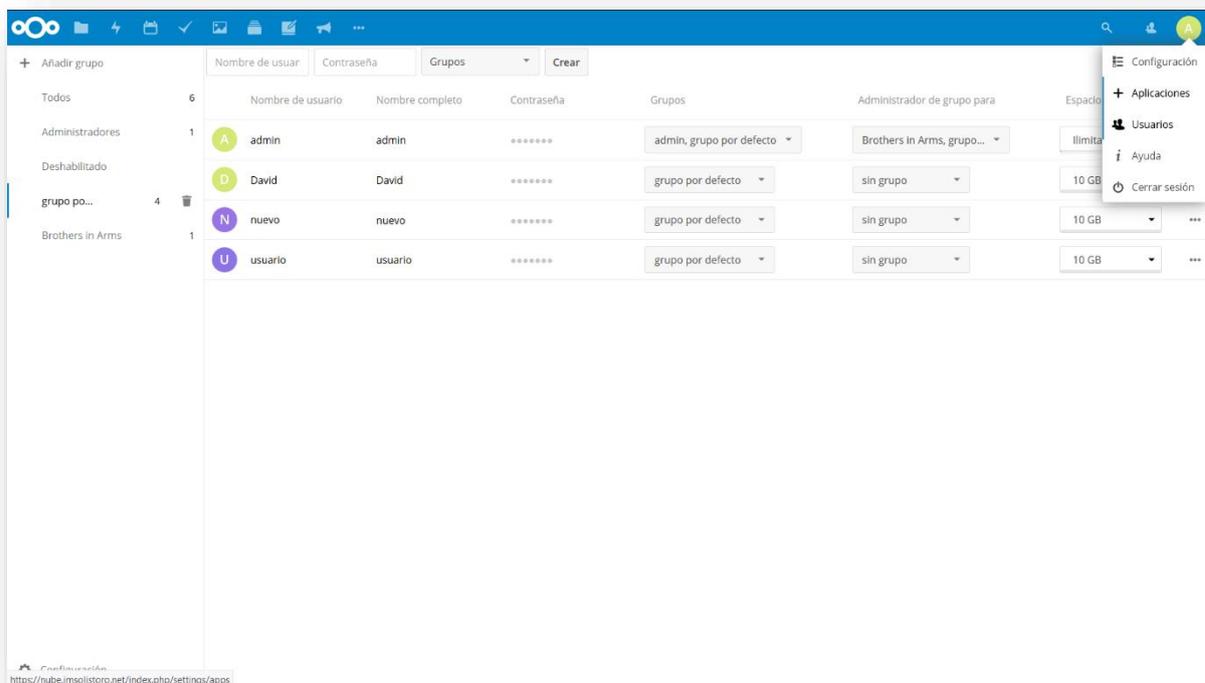
A continuación se listan algunas aplicaciones útiles que pueden añadirse a una instalación estándar de *Nextcloud*:

1. Actividad.
2. Marcadores.
3. Etiquetas colaborativas.
4. Requerimiento de datos.
5. Módulo de cifrado por defecto.
6. Cifrado de extremo a extremo.
7. Sitios externos.
8. Suplantar.
9. Click derecho.
10. Adjuntos de correo electrónico.
11. Suplantar usuarios para el administrador.
12. Integración de correo electrónico.
13. Votaciones.
14. Registro de usuarios.
15. Reproductor de vídeo y audio.
16. Edición online de documentos.

6. Administración de usuarios en Nextcloud.

6.1. Usuarios y grupos. Cuotas de espacio.

El administrador de Nextcloud es el que controla la creación de usuarios y grupos desde **Configuración, Usuarios:**



Desde esa pantalla, a la que tiene acceso cualquier administrador de *Nextcloud*, podemos:

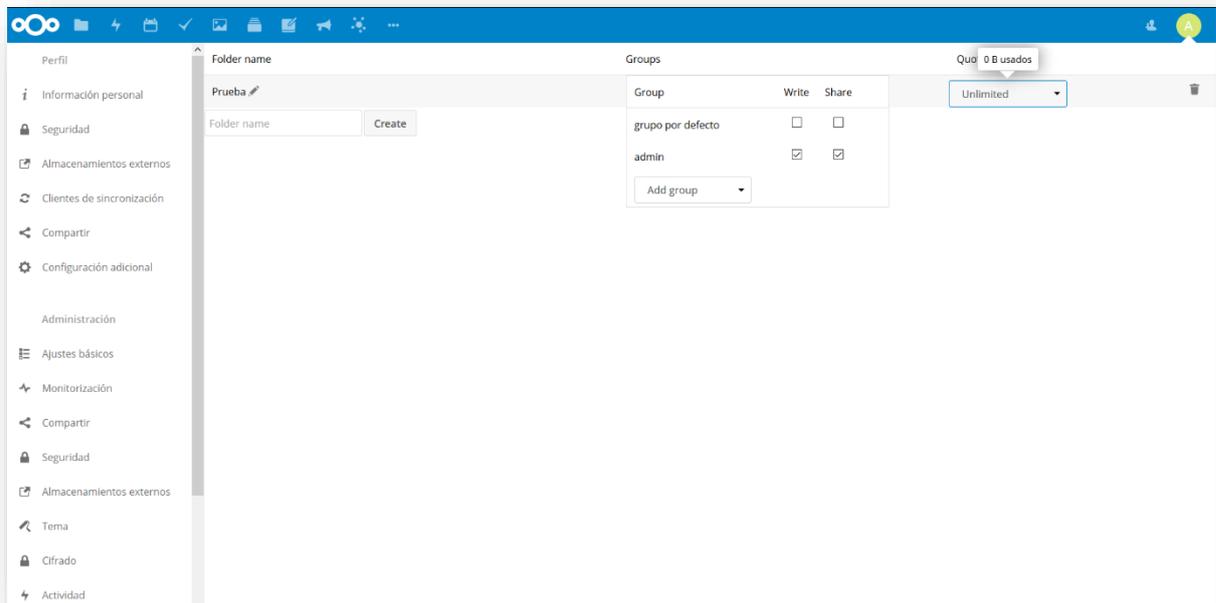
1. Crear y eliminar usuarios.
2. Crear y eliminar grupos. En una instalación básica de Nextcloud existe por defecto el grupo *admin*, cuya pertenencia da derechos de administrador en Nextcloud.
3. Asignar o retirar usuarios de grupos específicos.
4. Cambiar la cuota de espacio asignada a cada usuario.

6.2. Control de acceso a archivos.

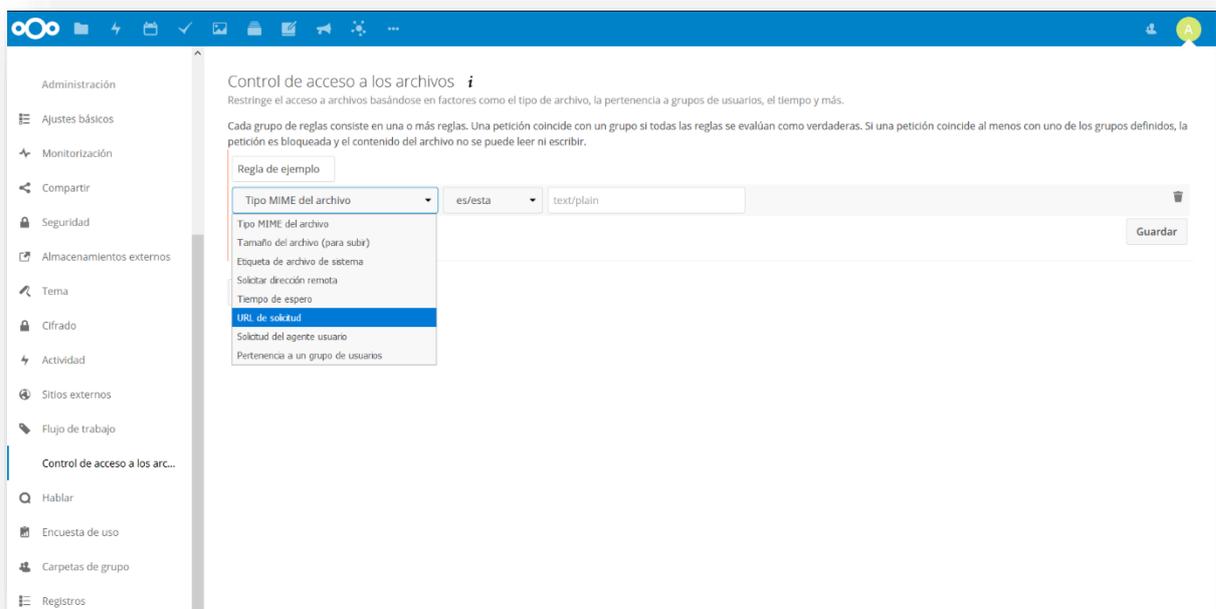
Por defecto, el control de acceso a archivos y carpetas es muy simple: cada usuario puede acceder a sus propios archivos, y al compartir sus archivos con otros usuarios y grupos éste puede definir si lo hace con permiso de escritura o solo lectura.

Con la llegada de las aplicaciones de *Carpetas de grupo* y *Control de acceso a ficheros*, tendremos acceso como administrador a nuevas opciones de control de acceso de archivos y carpetas.

Con *Carpetas de grupo*, podremos crear carpetas que no pertenecen a ningún usuario específico y sobre la que podremos dar permisos de lectura, escritura y compartición granulados a grupos. También podremos asignar una cuota de espacio a cada una de estas carpetas:



En *Control de acceso a archivos* (menú **Configuración**) podremos definir reglas más o menos complejas para restringir aún más el acceso a archivos según se evalúen como verdaderas o falsas las reglas que definamos:



7. Seguridad y cifrado en Nextcloud.

Existen tres tipos de cifrado en Nextcloud, cada uno con un objetivo y una serie de peligros, bien definidos en cada caso, sobre los que protege y los que no. En los siguientes puntos hablamos de cada uno de ellos.

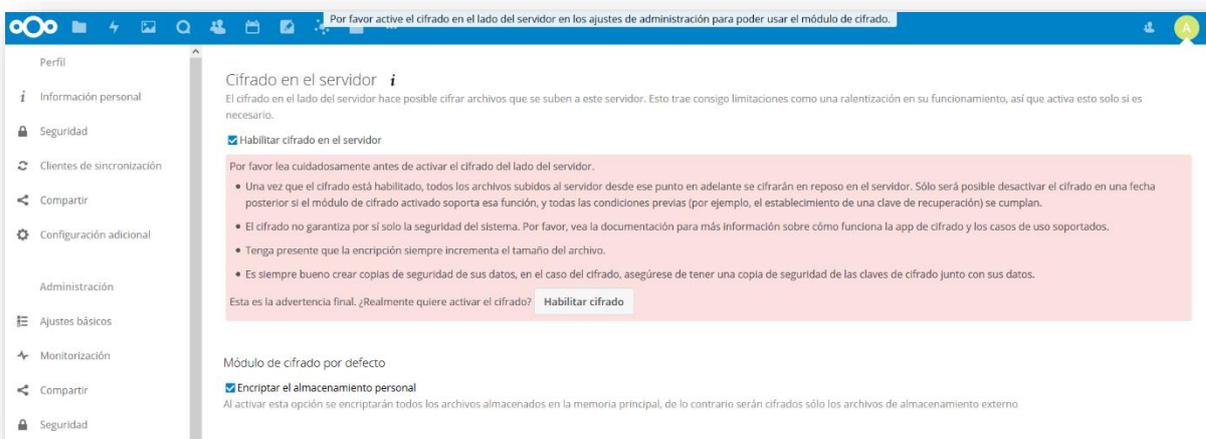
7.1. Cifrado en la comunicación entre cliente y servidor.

El primer tipo de cifrado en *Nextcloud* es la encriptación de las comunicaciones entre cliente y servidor. Ya hemos tratado e implementado este cifrado en la **Sección Configuración de Nextcloud y el servidor web para la conexión cifrada**. La implementación de este cifrado es imprescindible para tener un despliegue funcional y que cumpla las más mínimas normas de seguridad y privacidad en Internet.

- **De qué protege:** de cualquier intento de capturar la información que viaja entre cliente y servidor a través de Internet, redes de área local, wifis públicas y privadas, etc.
- **De qué no protege:** de un cliente o un servidor comprometidos o *hackeados*.

7.2. Cifrado del almacenamiento en el servidor.

Nextcloud permite activar el cifrado en el servidor desde el menú de **Configuración**:



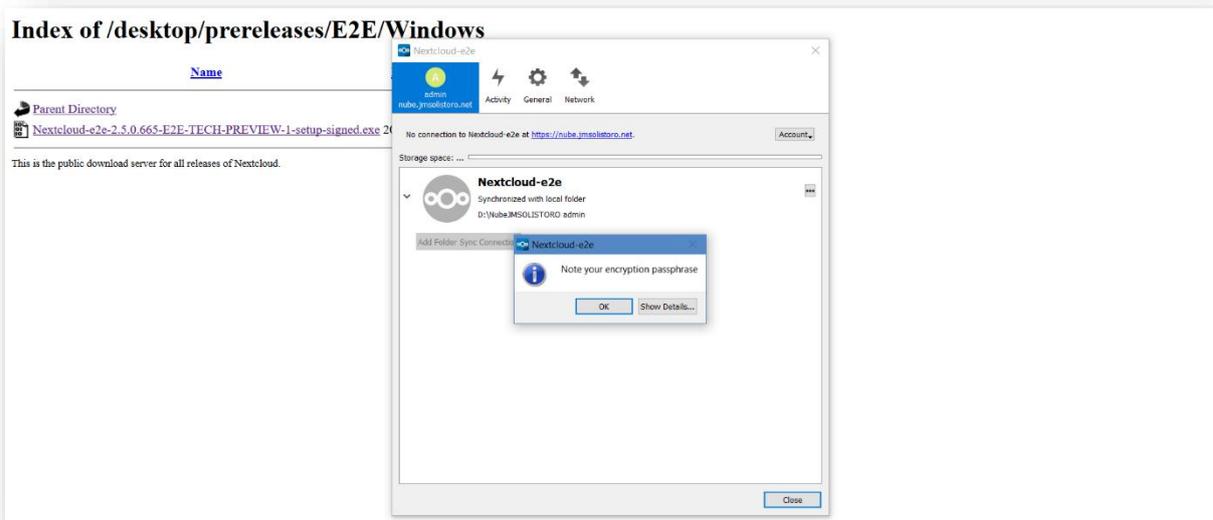
Básicamente lo que hace es cifrar cada archivo que se almacena en el servidor mediante una clave única que depende de una clave global del servidor o de las claves de usuario de las cuentas, según como se desee configurar.

- **De qué protege:** protege los datos de usuarios almacenados siempre y cuando éstos no se encuentren en el mismo servidor que la instalación de *Nextcloud*, sino en un almacenamiento externo.
- **De qué no protege:** de un servidor comprometido o un administrador desleal, aunque se opte por cifrar cada cuenta de usuario con las claves de sus cuentas.

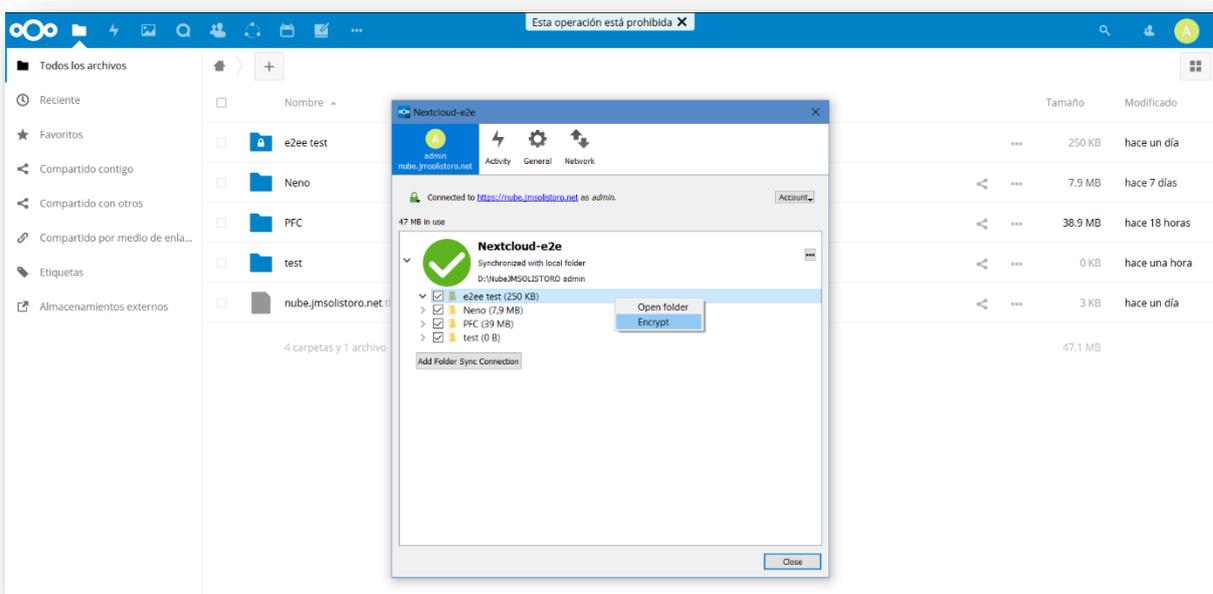
7.3. Cifrado de extremo a extremo.

El cifrado de extremo a extremo es muy nuevo en *Nextcloud* y todavía está en fase alfa al momento de escribir esta memoria, pero se espera que en pocos meses esta característica alcance la madurez suficiente para que pueda ser adoptada en servidores en producción.

Para usar esta característica tenemos que instalar y activar la app **End to End encryption** y, en el momento de escribir esta memoria, descargar un cliente especial en fase beta desde <https://download.nextcloud.com/desktop/prereleases/E2E/>.



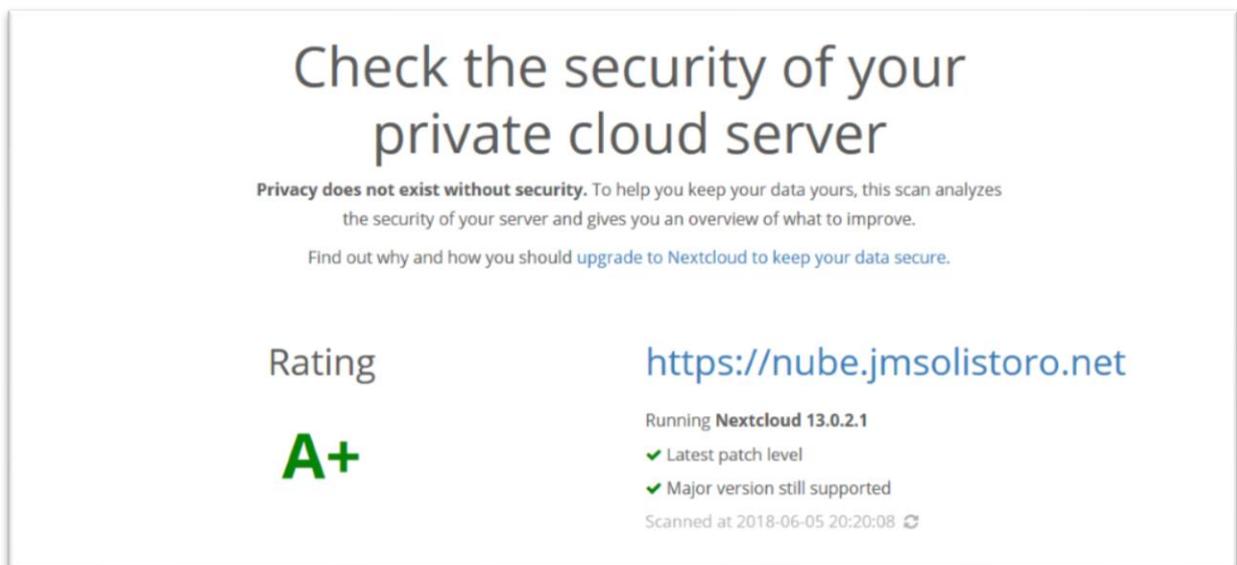
Este nuevo cliente nos permitirá definir carpetas que se almacenarán cifradas en el servidor y sólo aparecerán descifradas en nuestro cliente:



- **De qué protege:** de cualquier ataque que ocurra fuera de los equipos cliente de los usuarios, incluso en el caso de que el servidor haya sido (o esté inadvertidamente) comprometido, y contra administradores de servidores desleales.
- **De qué no protege:** no protege los datos en los equipos clientes si éstos no estuvieran también cifrados ni de ataques para robarles las claves privadas y usarlas para descifrar los datos.

7.4. Auditoría de seguridad externa para servidores *Nextcloud*.

Los desarrolladores de *Nextcloud* ponen a disposición de los administradores de servidores un escáner de seguridad en el que podemos poner la dirección de nuestro *Nextcloud* y nos mostrará un informe sobre la seguridad de nuestro servidor y el *Nextcloud* instalado en él. Con los pasos descritos en esta memoria **obtenemos la más alta puntuación:**



Check the security of your private cloud server

Privacy does not exist without security. To help you keep your data yours, this scan analyzes the security of your server and gives you an overview of what to improve.

Find out why and how you should [upgrade to Nextcloud to keep your data secure](#).

Rating

A+

<https://nube.jmsolistoro.net>

Running **Nextcloud 13.0.2.1**

- ✓ Latest patch level
- ✓ Major version still supported

Scanned at 2018-06-05 20:20:08 ↻

F = This server version is end of life and has no security fixes anymore. It is likely trivial to break in and steal all the data or even take over the entire server.

E = This server is vulnerable to at least one vulnerability rated "high". It is likely quite easy to break in and steal data or even take over the server.

D = This server is vulnerable to at least one vulnerability rated "medium". With bit of effort, like creating a specially crafted URL and luring a user there, an attacker can likely steal data or even take over the server.

C = This server is vulnerable to at least one vulnerability rated "low". This might or might not provide a way in for an attacker and will likely need some additional vulnerabilities to be exploited.

A = This server has no known vulnerabilities but there are additional hardening capabilities available in newer versions making it harder for an attacker to exploit unknown vulnerabilities to break in.

A+ = This server is up to date, well configured and has industry leading hardening features applied, making it harder for an attacker to exploit unknown vulnerabilities to break in. [Learn more about these preventive hardening features](#).

8. Copias de seguridad.

8.1. Sistema de revisiones y papelera de *Nextcloud*.

Aunque no puede considerarse un sistema de copia de seguridad o *backup*, *Nextcloud* tiene dos características que pueden ayudar en caso de borrado de datos por parte de los usuarios:

1. **Papelera de reciclaje de cada usuario:** en caso de borrado de archivos (accidental o no), *Nextcloud* tiene un sistema de papelera de reciclaje desde donde recuperar dichos archivos. *Nextcloud* administra esta papelera de manera que su tamaño no alcance el 50% del espacio libre restante en la cuenta.
2. **Control de versiones:** si la app **Versions** está activada, *Nextcloud* conservará versiones antiguas de los archivos modificados, por lo que podremos recuperar ficheros sobrescritos (modificados) de manera accidental o intencionada.

La app de versiones automáticamente mantiene las versiones más antiguas según este patrón:

- Para el último segundo se mantiene una versión.
- Para los últimos 10 segundos se mantiene una versión por cada 2 segundos.
- Para el último minuto se mantiene una versión por cada 10 segundos.
- Para la última hora se mantiene una versión por cada minuto.
- Para las últimas 24 horas se mantiene una versión por cada hora.
- Para los últimos 30 días se mantiene una versión por cada día.
- Para después de los últimos 30 días se mantiene una versión por cada semana.

Las versiones disponibles se ajustan a este patrón cada vez que se crea una versión nueva. Al igual que con la papelera, si el espacio ocupado por las versiones supera el 50% del espacio libre de la cuenta, las versiones más antiguas son borradas hasta que se baja de nuevo del límite.

8.2. Copia de seguridad completa en un servidor externo.

El objetivo de la copia de seguridad es poder restaurar el servicio de **Nextcloud**, en el menor tiempo posible, en el caso de que ocurriera un fallo catastrófico en el servidor. Como por ejemplo en el caso de la destrucción física de la máquina.

La estrategia de copias seguridad que vamos a seguir será:

1. **Copias de seguridad incrementales una vez al día:** en cada proceso de copia solo se copiarán los archivos nuevos y los que hayan sido modificados desde la última copia.
2. **Política de retención de archivos:** 30 días (se guardarán en el servidor de backup todos los archivos -y sus versiones- con hasta 30 días de antigüedad).
3. **Elementos que se copiarán:**
 - Base de datos de *Nextcloud*, de nombre **nextcloud**.
 - Carpeta de completa de *Nextcloud*, que incluye la carpeta de archivos de usuarios (**data**) y configuraciones de *Nextcloud*: **`/var/www/html/nube/`**

- Archivos de configuración relevantes del servidor, que servirán como guía para replicar el servicio de forma más rápida en otro: configuraciones de **Apache, PHP, Redis, Fail2ban, Crontab** y **Let's Encrypt**.
- Carpeta de *scripts* de **BASH** que hemos creado para automatizar las tareas del servidor: **/root/scripts/**.

La base de datos, la carpeta **data/**, la carpeta **config/** y la carpeta **themes/** de *Nextcloud* es lo mínimo necesario para restaurar (o clonar) una instancia de *Nextcloud* completa. Con los dos primeros puntos de *Elementos que se copiarán* nos aseguramos de que tenemos esas 3 carpetas y la base de datos.

Para el proceso que vamos a describir nos valdría cualquier servidor, probablemente necesitemos mucho espacio, con conexión a Internet y al que tengamos acceso por **SSH** como usuario con o sin privilegios. Debería tener también un buen ancho de banda si queremos transferir grandes cantidades de datos en un tiempo razonable.

Para las copias de seguridad dispongo de un segundo servidor, también en la nube y en una ubicación física diferente a donde se encuentra el servidor de **Nextcloud**, dedicado en exclusiva a albergar copias de seguridad:

- Procesador ARMv7 rev 1 (v7l) de 2 núcleos.
- 2GBytes de RAM.
- 1 HD de 6TBytes brutos, 5,41TBytes netos.
- Sistema operativo Ubuntu 16.04 LTS para procesadores ARM.
- SSH: **backup.jmsolistoro.net:41093**.

En dicho servidor ya tenemos asignado un usuario sin privilegios que usaremos para guardar y restaurar las copias:

1. Usuario: **jmsolistoro**.
2. Clave: **LaDelBackup33**.

Para acceder al servidor **backup.jmsolistoro.net** vamos a preparar el acceso sin clave usando certificado, para que no haga falta que aparezca la clave en los *scripts* de *backup*. El usuario que ejecutará las operaciones de *backup* desde **nube.jmsolistoro.net** será **root**, y desde éste vamos a preparar la conexión sin claves:

1. Desde el usuario **root** en **nube.jmsolistoro.net** (no hace falta cambiar la carpeta por defecto ni añadir una *passphrase*):

```
root@nube:~# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
```

2. Copiamos el certificado en **backup.jmsolistoro.net** desde **nube.jmsolistoro.net** usando la clave de **jmsolistoro** por última vez:

```

root@nube:~# ssh-copy-id jmsolistoro@backup.jmsolistoro.net -p 41093
/usr/bin/ssh-copy-id: INFO: Source of key(s) to be installed:
"/root/.ssh/id_rsa.pub"
The authenticity of host '[backup.jmsolistoro.net]:41093 ([213.32.0.213]:41093)'
can't be established.
ED25519 key fingerprint is SHA256:uWNwvYHamr6CtRswb4B480bLKpGosYK+mQaZZdAzOFk.
Are you sure you want to continue connecting (yes/no)? yes
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out
any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted
now it is to install the new keys
jmsolistoro@backup.jmsolistoro.net's password:

Number of key(s) added: 1

Now try logging into the machine, with:  "ssh -p '41093'
'jmsolistoro@backup.jmsolistoro.net'"
and check to make sure that only the key(s) you wanted were added.

```

Los certificados se han guardado en la carpeta `/root/.ssh/` y es ahí donde vamos a crear un archivo adicional, llamado simplemente **config**, en el que especificaremos la configuración por defecto para conectar con el servidor de *backup*:

```

host backup.jmsolistoro.es
port 41093

```

Esto es necesario porque el puerto **SSH** para conectar al servidor de *backup* no es el estándar. Estableciendo la configuración de esta manera no tendremos que acordarnos de explicitar el puerto al hacer las conexiones a **backup.jmsolistoro.es**, y tendremos menos problemas para configurar las copias remotas.

Ya podemos entrar desde el usuario **root** de *nube.jmsolistoro.net* en *backup.jmsolistoro.net* como el usuario **jmsolistoro**, y vamos a comprobarlo y a crear la carpeta donde se guardarán las copias de seguridad:

```

root@nube:~# ssh jmsolistoro@backup.jmsolistoro.net
jmsolistoro@backup:~$ mkdir nube.jmsolistoro.net
jmsolistoro@backup:~$ ls -l
total 4
drwxrwxr-x 2 jmsolistoro jmsolistoro 4096 may 28 13:42 nube.jmsolistoro.net

```

La carpeta de los *backups* será `/home/jmsolistoro/nube.jmsolistoro.net` en *backup.jmsolistoro.net*.

Para realizar las copias de seguridad incrementales desde *nube.jmsolistoro.net* vamos a usar la herramienta [backupninja](#), la cual vamos a instalar y configurar:

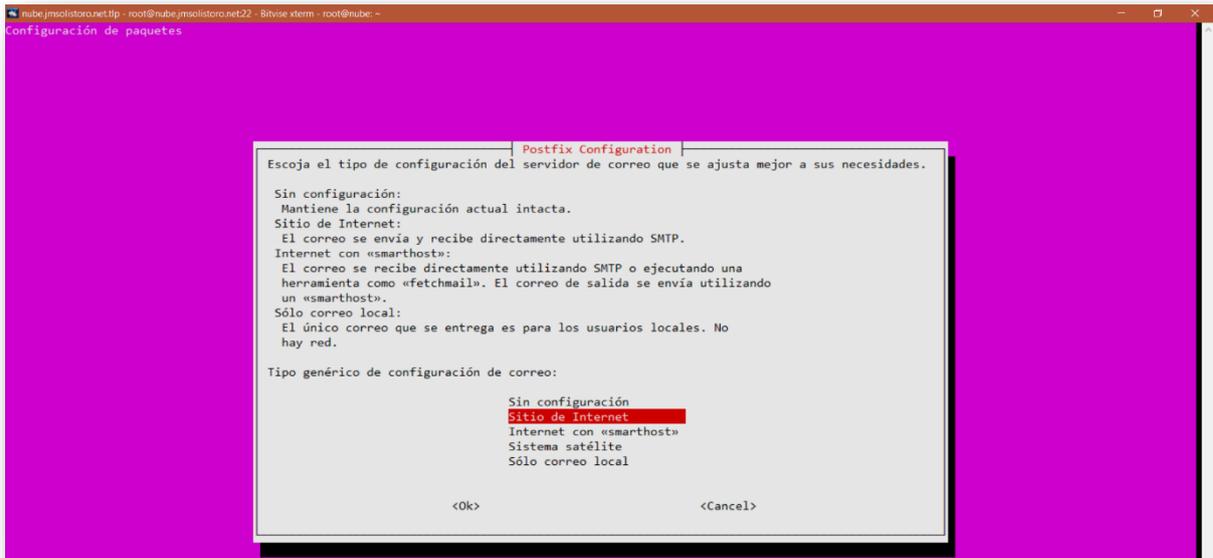
```

root@nube:~# apt install backupninja hwinfn rdiff-backup

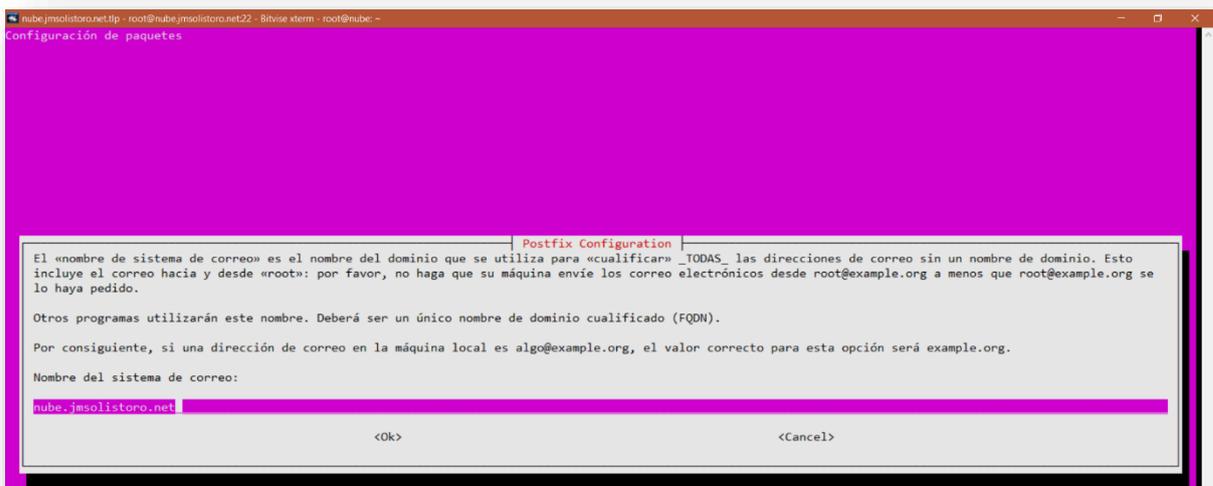
```

Durante la instalación se nos pedirá cierta información para configurar el servidor de correo **postfix**, que **backupninja** usará para notificarnos por correo electrónico del resultado de los *backups*. Debemos configurarlo como sigue:

1. Configuración del servidor de correo: **sitio de internet**.



2. Nombre de dominio cualificado (FQDN): *nube.jmsolistoro.net*.

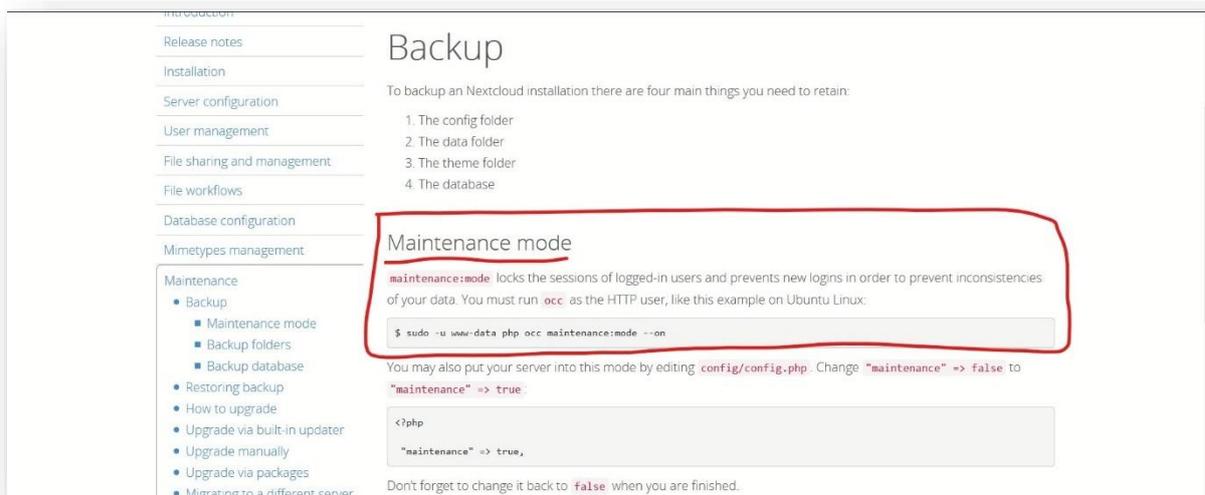


Podemos comprobar que nuestro servidor de correo funciona al enviar correos enviándonos uno, a la dirección nube@jmsolistoro.net por ejemplo, desde la línea de comandos:

```
root@nube:~# echo 'Prueba de correo' | mail -s 'Correo enviado desde
nube.jmsolistoro.net' nube@jmsolistoro.net
```

En `/var/log/mail.log` podemos consultar los mensajes del sistema de correo si no funcionara bien.

Nextcloud debe estar en modo mantenimiento para proceder con la copia de la base de datos y los archivos de usuario, [tal y como se explica en la documentación de Nextcloud](#):



Se hace así para evitar inconsistencias si mientras se realiza el *backup* hay cambios en los archivos y en la base de datos. Tiene como inconveniente que mientras se realiza la copia, el servicio de *Nextcloud* no estará disponible para la sincronización de archivos. Y si la copia es de mucho volumen, como ocurre cuando hacemos la primera copia de un *Nextcloud* que ya esté en producción y con muchos usuarios, el proceso puede conllevar bastantes horas.

Una vez instalado *backupninja* es momento de configurar las copias de seguridad. Usaremos la herramienta *ninjahelper*, que es parte de *backupninja*:

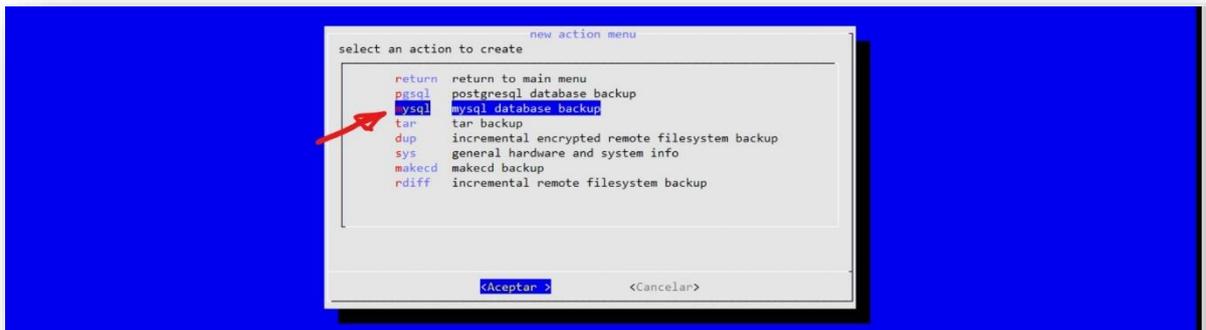
```
root@nube:~# ninjahelper
```

Y empezaremos por hacer la copia de la base de datos de *Nextcloud*:

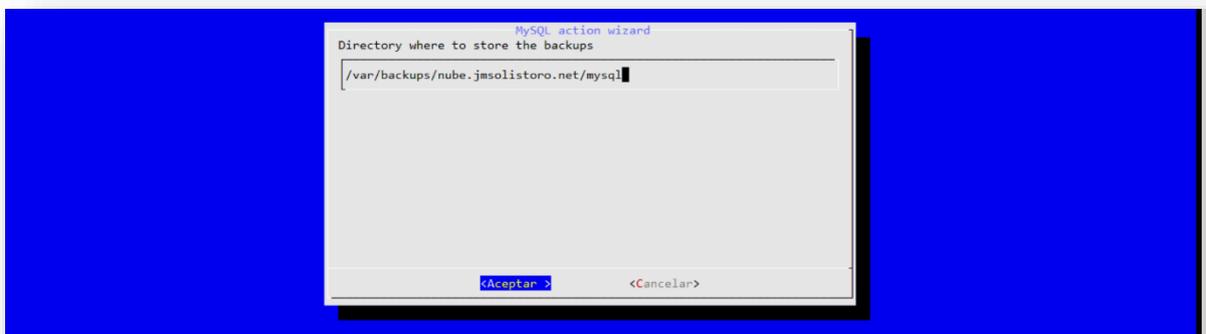
1. La primera vez que ejecutamos *ninjahelper* nos preguntará si queremos crear la primera tarea de *backup* o salir. Elegiremos nueva:



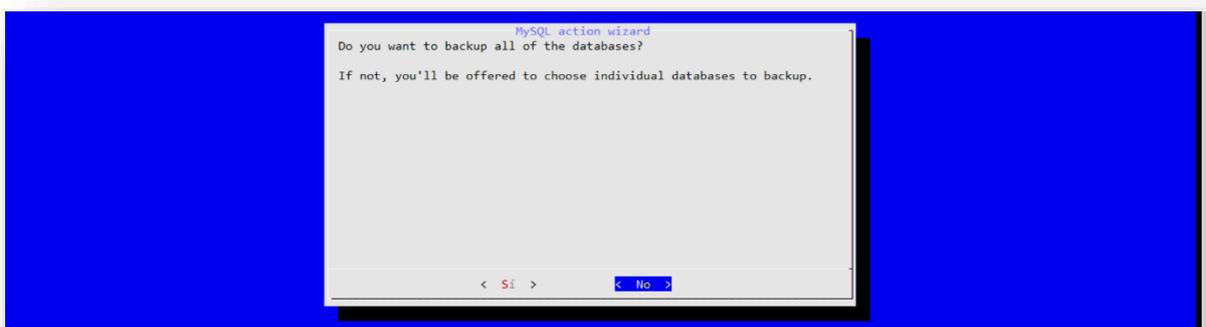
2. Seleccionamos base de datos *MySQL*:



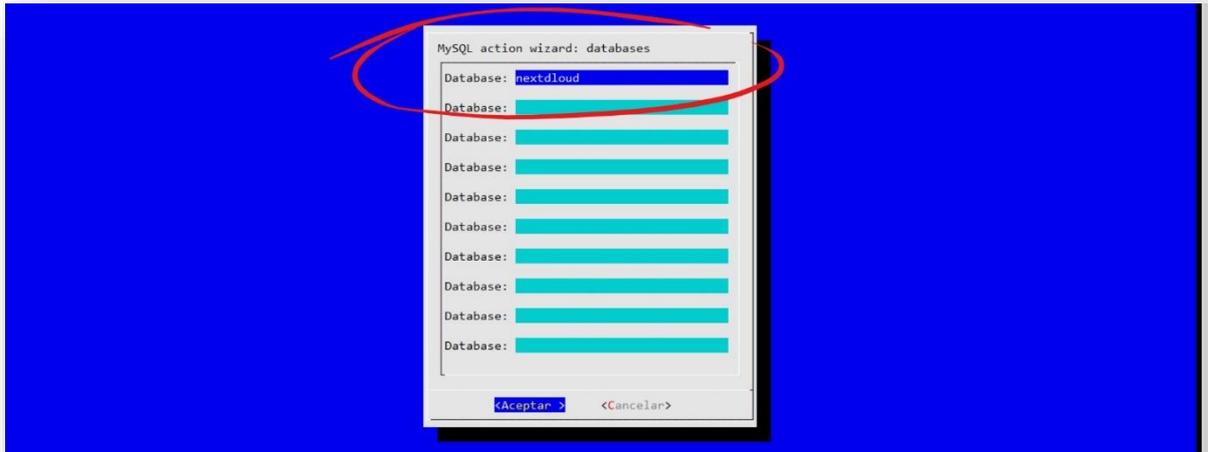
3. Lo primero que tenemos que poner es el lugar donde guardaremos el volcado de la base datos. Esta carpeta será local, `/var/backups/nube.jmsolistoro.net/mysql/`, y en un proceso posterior se copiará al servidor remoto de las copias de seguridad.



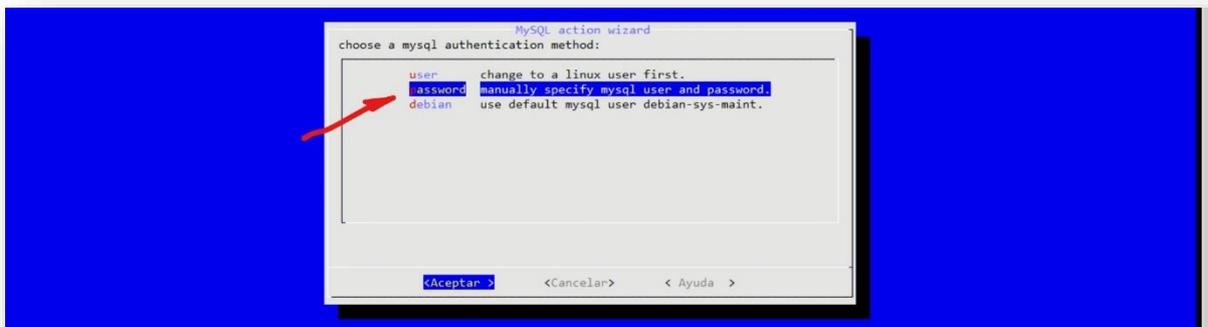
4. No queremos volcar todas las bases de datos, solo la que corresponde a *Nextcloud* seleccionando **No**:



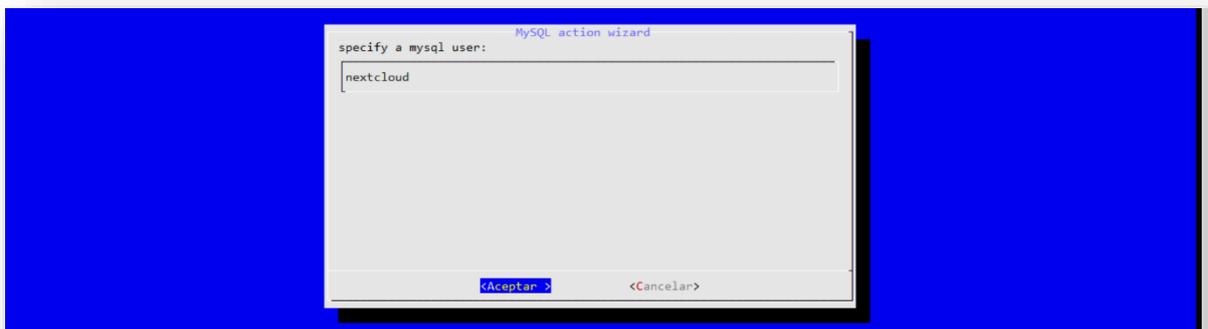
5. Debemos poner el nombre de la base de datos en la siguiente pantalla, *nextcloud*:



6. Elegiremos establecer un usuario y una clave manualmente:



7. El usuario será **nextcloud**:



8. Y la clave del usuario **nextcloud** de la base de datos era **LaClave22**:



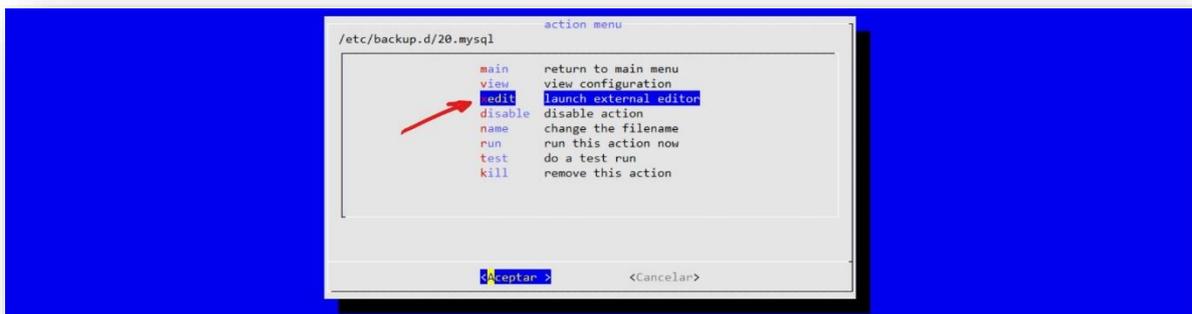
9. Las opciones que debemos marcar son **sqldump** y **compress**:



10. Hemos llegado al final del proceso y nos devuelve al menú principal, donde vemos la tarea recién creada. Debemos seleccionarla para editarla:



11. Y elegimos la opción **Launch external editor** para añadir al final de la configuración la hora de la tarea, así como para comprobar que las otras opciones (bases de datos, usuario, clave, etc.) se introdujeron correctamente. La configuración adicional para la hora de la tarea está resaltada en rojo:



```

### backupninja MySQL config file ###
# hotcopy = < yes | no > (default = no)
# make a backup of the actual database binary files using mysqlhotcopy.
hotcopy = no

# sqldump = < yes | no > (default = no)
# make a backup using mysqldump. this creates text files with sql commands
# sufficient to reconstruct the database.
#
sqldump = yes

# sqldumptions = <options>
# (default = --lock-tables --complete-insert --add-drop-table --quick --quote-names)
# arguments to pass to mysqldump
# sqldumptions = --add-drop-table --quick --quote-names

# compress = < yes | no > (default = yes)
# if yes, compress the sqldump output.
compress = yes

# dbhost      = <host> (default = localhost)

# backupdir = <dir> (default: /var/backups/mysql)
# where to dump the backups. hotcopy backups will be in a subdirectory
# 'hotcopy' and sqldump backups will be in a subdirectory 'sqldump'
backupdir = /var/backups/nube.jmsolistoro.net/mysql

# databases = <all | db1 db2 db3 > (default = all)
# which databases to backup. should either be the word 'all' or a
# space separated list of database names.
databases = nextcloud

dbusername = nextcloud
dbpassword = LaClave22

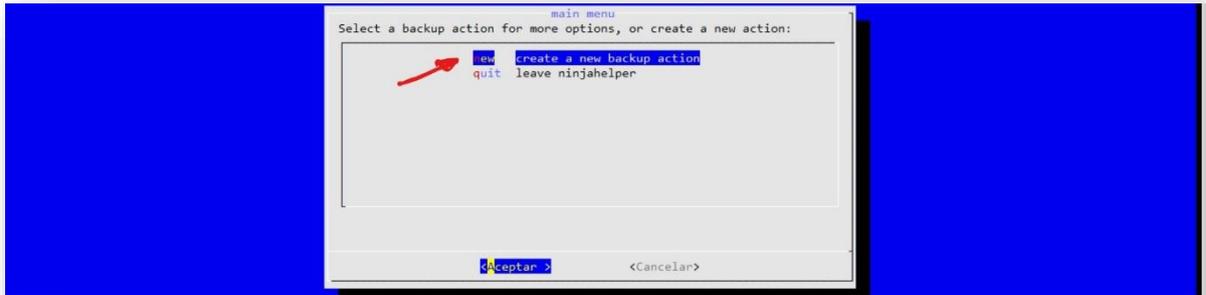
#Hora de la copia
when = everyday at 03:00

```

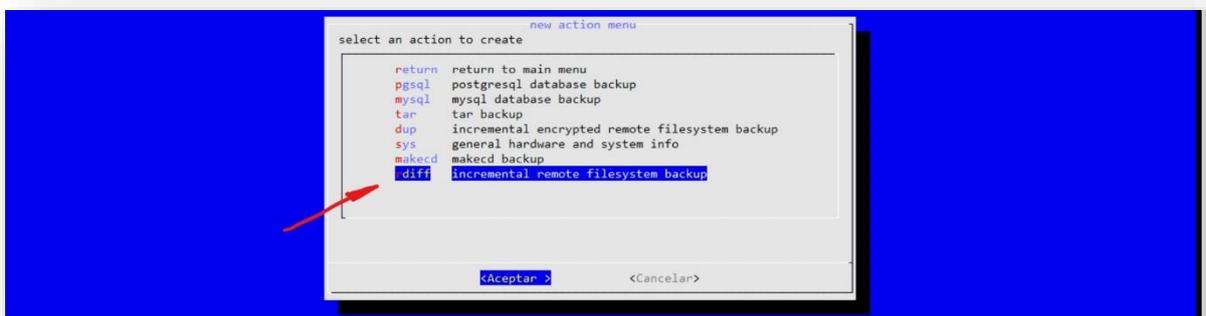
Después de configurar el volcado de la base de datos en local, ya podemos pasar a crear la tarea que copiará todo el contenido de **Nextcloud**, el volcado de su base de datos y los archivos de configuración relevantes del servidor, al servidor remoto de **backup**.

Volviendo a **ninjahelper**:

1. Seleccionaremos **create a new backup action**:



2. Y elegiremos como tarea ***incremental remote filesystem backup***:



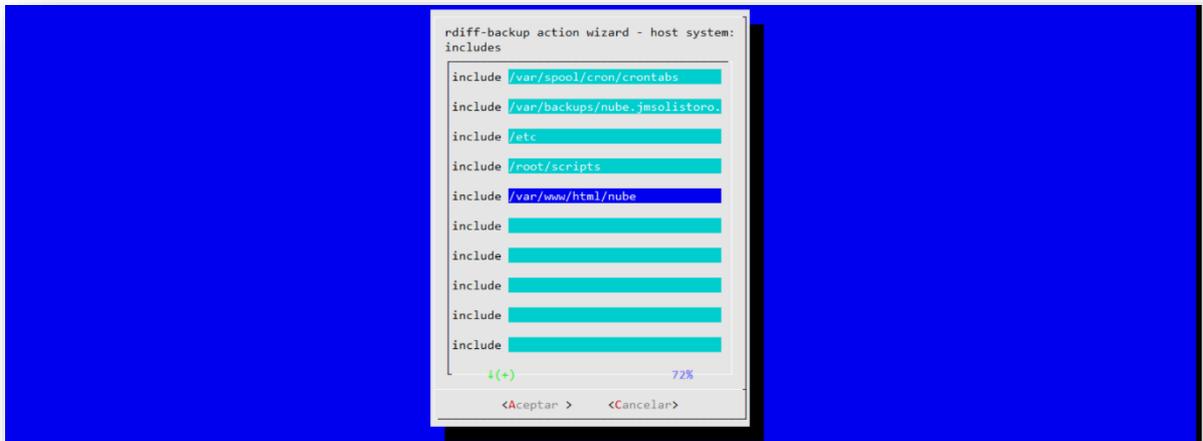
3. Tenemos múltiples apartados de configuración, y empezaremos por el primero: ***choose files to include & exclude***.



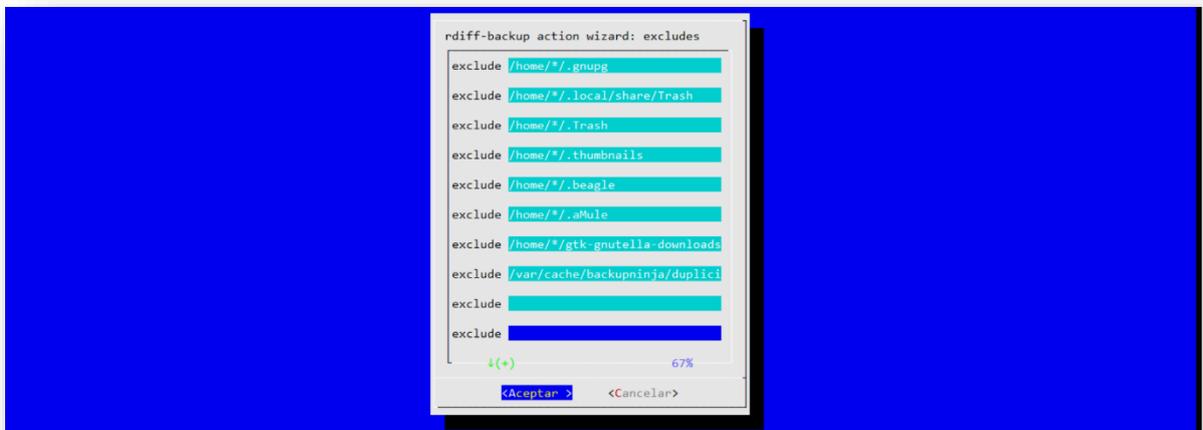
4. En la nueva pantalla pondremos todas las carpetas a grabar que ya hemos mencionado:

- ***/var/spool/cron/crontabs***: aquí se guardan las tareas de mantenimiento del *Cron*.
- ***/var/backups/nube.jmsolistoro.net***: contiene el volcado de la base de datos de *Nextcloud*.
- ***/etc***: esta carpeta contiene todas las configuraciones del servidor, que incluyen todas las que a nosotros nos interesan, como ***Apache, PHP, Redis, Fail2ban, Crontab*** y ***Let's Encrypt***.

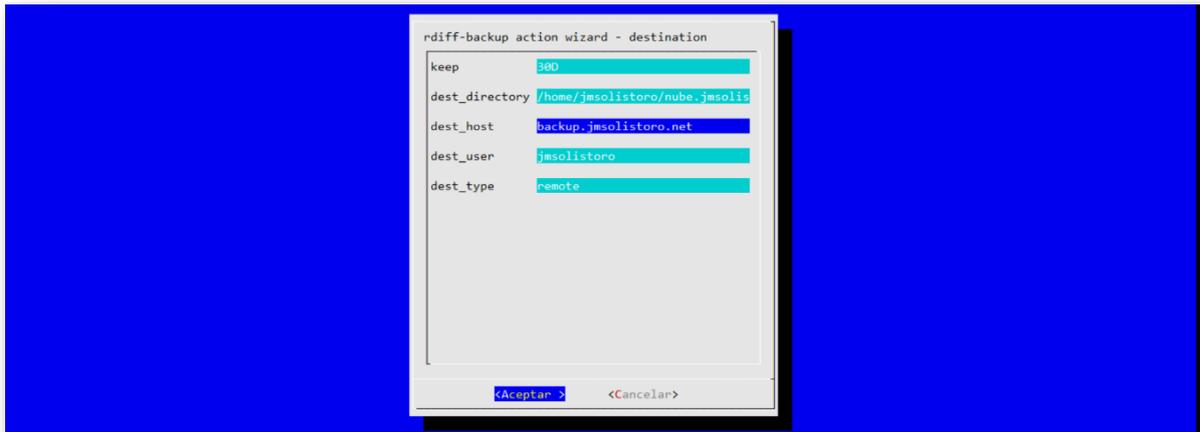
- **/root/scripts**: los *scripts* de *BASH* que ejecuta el *Cron* para tareas de mantenimiento.
- **/var/www/html/nube**: contiene todos los archivos de usuario y de configuración de *Nextcloud*.



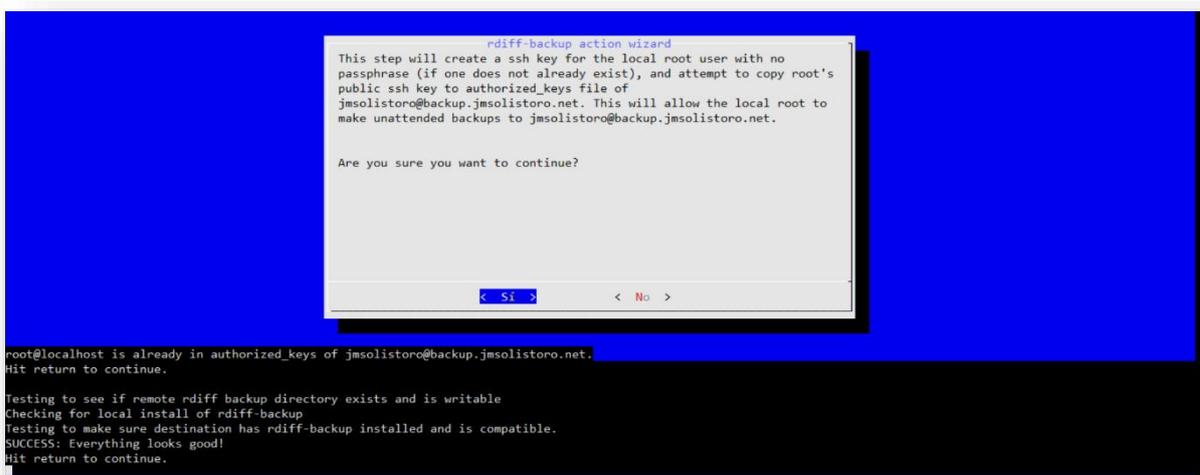
5. Después nos pedirá que especifiquemos las exclusiones. No hace falta que explicitemos ninguna, pudiendo borrar todas las que aparecen por defecto. Pero podemos dejarlo como está y pulsar aceptar, ya que no tendrán influencia en las carpetas que hemos configurado:



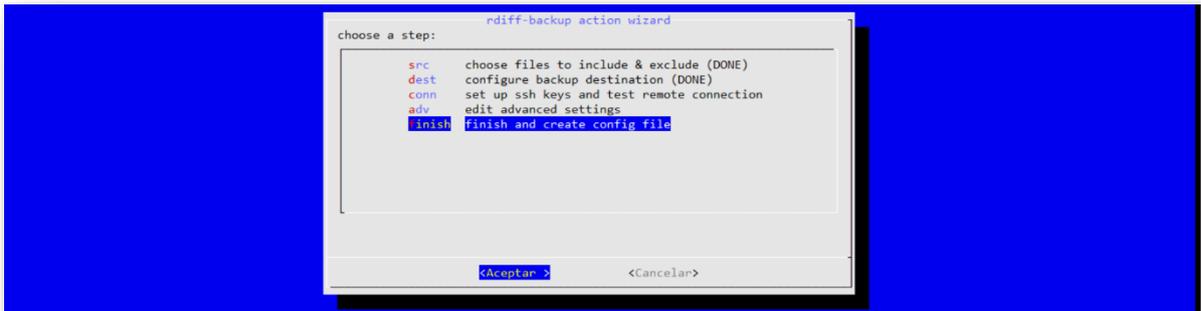
6. El siguiente paso es configurar el destino con las siguientes opciones:
 - **keep**: 30D (los ficheros borrados o cambiados se guardarán durante 30 días).
 - **dest_directory**: /home/jmsolistoro/nube.jmsolistoro.net
 - **dest_host**: backup.jmsolistoro.net
 - **dest_user**: jmsolistoro
 - **dest_type**: remote



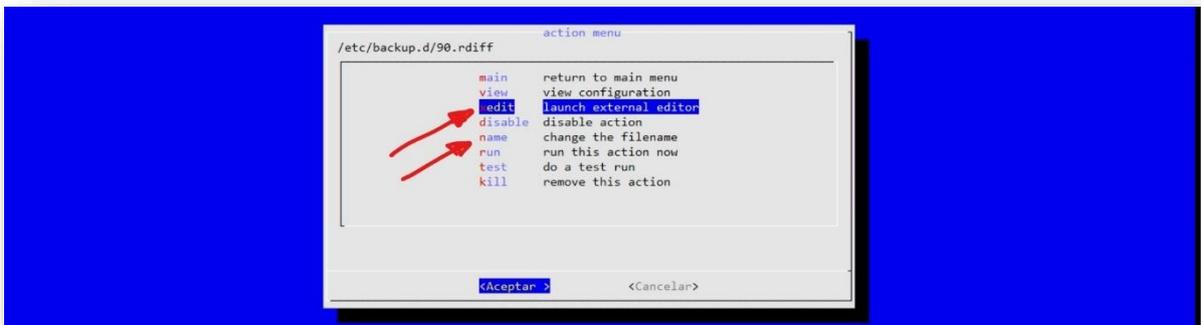
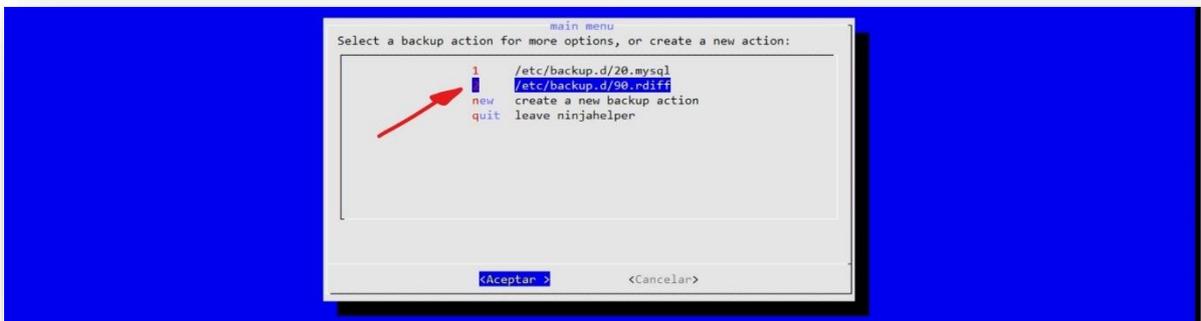
7. **Ninjahelper** nos obliga en el siguiente paso a configurar los certificados para el acceso remoto y a probar la conexión. Comprobará que la conexión remota por certificados ya está configurada y no duplicará nada:



8. Podemos ignorar las opciones avanzadas y grabar ya directamente la tarea:



9. Tenemos la tarea creada, pero hay que editarla para añadirle una hora (03:00, la misma que el volcado de base de datos) y cambiarle el nombre:



10. Para poner la hora añadimos al final un **when**, quedando el código de la tarea como sigue (líneas añadidas en rojo):

```
# options = --force
# when = everyday at 02

## should backupninja write program output as Info messages rather than Debug
## messages? (default: no)
## Usually rdiff-backup output (for increment expiration and backup) is written
## to output as Debug messages; this option causes backupninja to use Info-level
## messages instead. Since backup reports include Info messages, this option is
## useful to receive output like rdiff-backup session statistics in reports. In
## addition, since rdiff-backup has a habit of using a zero exit code when
## non-fatal errors are encountered (causing backupninja to conclude the backup
```

```

## was entirely successful), this option is useful for inspecting non-fatal
## filesystem and permission errors from rdiff-backup.
output_as_info = no

[source]
type = local
keep = 30D

# A few notes about includes and excludes:
# 1. include, exclude and vsinclude statements support globbing with '*'
# 2. Symlinks are not dereferenced. Moreover, an include line whose path
#    contains, at any level, a symlink to a directory, will only have the
#    symlink backed-up, not the target directory's content. Yes, you have to
#    dereference yourself the symlinks, or to use 'mount --bind' instead.
#    Example: let's say /home is a symlink to /mnt/crypt/home ; the following
#    line will only backup a "/home" symlink ; neither /home/user nor
#    /home/user/Mail will be backed-up :
#    include = /home/user/Mail
#    A workaround is to 'mount --bind /mnt/crypt/home /home' ; another one is to
#    write :
#    include = /mnt/crypt/home/user/Mail
# 3. All the excludes come after all the includes. The order is not otherwise
#    taken into account.

# files to include in the backup
include = /var/spool/cron/crontabs
include = /var/backups/nube.jmsolistoro.net
include = /etc
include = /root/scripts
include = /var/www/html/nube
exclude = /home/*/.gnupg
exclude = /home/*/.local/share/Trash
exclude = /home/*/.Trash
exclude = /home/*/.thumbnails
exclude = /home/*/.beagle
exclude = /home/*/.aMule
exclude = /home/*/.gtk-gnutella-downloads
exclude = /var/cache/backupninja/duplicity

#####
## destination section
## (where the files are copied to)

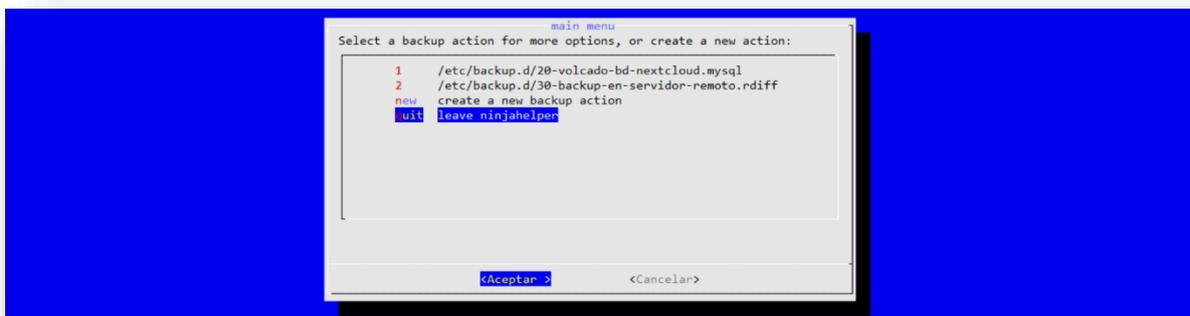
[dest]
type = remote
directory = /home/jmsolistoro/nube.jmsolistoro.net
host = backup.jmsolistoro.net
user = jmsolistoro

#Hora de la copia
when = everyday at 03:00

```

11. Vamos a cambiarles los nombres a las 2 tareas, la de volcado y la copia a servidor remoto. Esto debe hacerse porque las tareas que tienen la misma hora, como en este caso ambas a las 3 de la mañana, se ejecutan por orden alfabético. Queremos que primero haga el volcado de la base de datos y después la copia remota:

- **20-volcado-bd-nextcloud.mysql**
- **30-backup-en-servidor-remoto.rdiff**



Nos queda añadir dos tareas más: pasar a modo mantenimiento antes del volcado de la base de datos de *Nextcloud* y volver del modo mantenimiento tras terminar el backup remoto. No las podemos añadir con *ninjahelper*, así que las crearemos a mano en la carpeta */etc/backup.d/*:

1. *10-modo-mantenimiento-ON.sh*:

```
#Hora
when = everyday at 03:00
sudo -u www-data php /var/www/html/nube/occ maintenance:mode --on
```

2. *40-modo-mantenimiento-OFF.sh*:

```
#Hora
when = everyday at 03:00
sudo -u www-data php /var/www/html/nube/occ maintenance:mode -off
```

3. Por seguridad modificaremos los permisos para que las configuraciones de las tareas de *backup* no puedan ser leídas ni editadas por ningún usuario que no sea */root*:

```
root@nube:~# chmod 600 /etc/backup.d/*
```

Tras esto ya tenemos configuradas las cuatro tareas que componen la copia de seguridad remota, que se ejecutarán secuencialmente (en el orden correcto) a partir de las 3 de la mañana. Se ha elegido una hora suficientemente tarde para que haya el menor número posible de usuarios activos y suficientemente temprano para que si la copia remota tarda mucho no alcance a las horas activas de la mañana.

8.3. Configuración adicional de *backupninja*.

Vamos a hacer unos cambios en la configuración por defecto de *backupninja* para recibir por correo electrónico los informes de las tareas, aunque éstas no hayan dado error ni advertencias. Debemos hacerlo para comprobar que el sistema de avisos por correo electrónico de *backupninja + postfix* funciona correctamente.

Lo haremos editando el fichero de configuración */etc/backupninja.conf* y cambiando algunas opciones (cambios en rojo):

```
#
#
# B A C K U P N I N J A 
#
# main configuration file
#
# how verbose to make the logs
# 5 -- Debugging messages (and below)
# 4 -- Informational messages (and below)
# 3 -- Warnings (and below)
# 2 -- Errors (and below)
# 1 -- Fatal errors (only)
loglevel = 4

# send a summary of the backup status to
# this email address:
reportemail = nube@jmsolistoro.net

# if set to 'yes', a report email will be generated
# even if all modules reported success. (default = yes)
reportsuccess = yes
```

```

# if set to 'yes', info messages from handlers will be
# sent into the email (default = no)
reportinfo = yes

# if set to 'yes', a report email will be generated
# even if there was no error. (default = yes)
reportwarning = yes

# if set to 'yes', disk space usage will be included in
# the backup email report
reportspace = yes

# where to rsync the backupninja.log to be aggregated in
# a ninjareport
reporthost =

# what user to connect to reporthost to sync the
# backupninja.log
reportuser = ninja

# where on the reporthost should the report go
# NOTE: the name of the log will be used in the report,
# use a globally unique name, preferably the hostname
reportdirectory = /var/lib/backupninja/reports

# set to the administration group that is allowed to
# read/write configuration files in /etc/backup.d
admingroup = root

#####
# for most installations, the defaults below are good #
#####

# where to log:
logfile = /var/log/backupninja.log

# directory where all the backup configuration files live
configdirectory = /etc/backup.d

# where backupninja helper scripts are found
scriptdirectory = /usr/share/backupninja

# where backupninja libs are found
libdirectory = /usr/lib/backupninja

# whether to use colors in the log file
usecolors = yes

# default value for 'when'
when = everyday at 01:00

# if running vservers, set to yes
vservers = no

# programs paths
# SLAPCAT=/usr/sbin/slapcat
# LDAPSEARCH=/usr/bin/ldapsearch
# RDIFFBACKUP=/usr/bin/rdiff-backup
# CSTREAM=/usr/bin/cstream
# MYSQL=/usr/bin/mysql
# MYSQLHOTCOPY=/usr/bin/mysqlhotcopy
# MYSQLDUMP=/usr/bin/mysqldump
# PSQL=/usr/bin/psql
# PGSQLDUMP=/usr/bin/pg_dump
# PGSQLDUMPALL=/usr/bin/pg_dumpall
# GZIP=/bin/gzip
# GZIP_OPTS='--rsyncable'
# RSYNC=/usr/bin/rsync
# VSERVERINFO=/usr/sbin/vserver-info
# VSERVER=/usr/sbin/vserver
# VROOTDIR=/var/lib/vservers

```

Así recibiremos correos informativos en nube@jmsolistoro.net de todas las tareas que ejecute **backupninja**, sea cual sea el resultado. Más adelante, cuando comprobemos que todo está funcionando como esperamos, se pueden cambiar las opciones para recibir correos solo cuando haya advertencias y/o errores:

```

# if set to 'yes', a report email will be generated
# even if all modules reported success. (default = yes)
reportsuccess = no

# if set to 'yes', info messages from handlers will be

```

```
# sent into the email (default = no)
reportinfo = no
```

8.4. Protocolo de restauración del *backup* externo.

Si quisiéramos restaurar las copias de seguridad almacenadas en **backup.jmsolistoro.net** en otra máquina distinta a la original, para volver a poner en marcha el servicio de *Nextcloud*, deberíamos seguir este orden:

1. Prerrequisitos:
 - Servidor con Ubuntu 18.04 LTS o similar.
 - **Apache**, **MySQL** y **PHP** instalados en el servidor.
 - Suficiente espacio como para albergar los datos de usuario del *Nextcloud* que estamos restaurando.
2. Debemos configurar el acceso para el usuario **root** del nuevo servidor al servidor de *backup*, de la misma manera como se hizo en la **Sección 8.2**: acceso mediante certificados y configuración del puerto.
3. Crearemos una carpeta temporal en el nuevo servidor, por ejemplo **/root/copia_de_seguridad_restaurada/**, en la que restauraremos todos los archivos que hay en la copia de seguridad en **backup.jmsolistoro.net**:

```
root@nube:~# mkdir copia_de_seguridad_restaurada
```

4. Nos traeremos a esa carpeta todos los archivos en el servidor de *backup* usando el comando **rdiff-backup**:

```
root@nube:~# rdiff-backup -r 0h
jmsolistoro@backup.jmsolistoro.net:~/home/jmsolistoro/nube.jmsolistoro.net/
/root/copia_de_seguridad_restaurada
```

Ese comando restaurará todo el backup más reciente dentro de la carpeta **/root/copia_de_seguridad_restaurada**. Para explicación más exhaustiva de cómo funciona el comando **rdiff-backup** puede consultarse la documentación:

```
root@nube:~# man rdiff-backup
```

5. En la carpeta **/root/copia_de_seguridad_restaurada** se encuentra todo lo necesario para restaurar *Nextcloud*:
 - **/root/copia_de_seguridad_restaurada**
/var/backups/nube.jmsolistoro.net/mysql/sqldump/nextcloud.sql.gz. Base de datos que tendremos que cargar en el servidor (suponemos que ya tenemos configurados la base de datos y el usuario como se explicó en la **Sección 2.2**:

```
root@nube:~# mysql -u nextcloud -p
mysql> CREATE DATABASE database_name;
mysql> exit
root@nube:~# zcat /root/copia_de_seguridad_restaurada
/var/backups/nube.jmsolistoro.net/mysql/sqldump/nextcloud.sql.gz | mysql -u
nextcloud -p nextcloud
root@nube:~#
```

- **/root/copia_de_seguridad_restaurada** **/var/www/html/nube**. Archivos de *Nextcloud* que tendremos que mover a **/var/www/html/nube** y ejecutar:

```
root@nube:~# cd /var/www/html/  
root@nube:/var/www/html# chown -R www-data:www-data nube
```

Además de los archivos y la base de datos de *Nextcloud*, hemos hecho copia (y restaurado) las configuraciones de **Apache, PHP, Redis, backupninja, crontab**, etc. (y todo */etc*) que nos servirán para levantar los servicios con sus configuraciones salvadas, siguiendo las instrucciones de las **Secciones 2 y 3**, más que crear y modificar las configuraciones solo tendremos que restaurarlas a su lugar desde la copia.

Cuando tengamos **Apache, PHP y MySQL** funcionando (con la base de datos de *Nextcloud* cargada también) debemos lanzar una tarea de mantenimiento para que se recupere la sincronización, tal y como se explica en la [documentación oficial](#):

```
root@nube:~# cd /var/www/html/nube  
root@nube:/var/www/html/nube# sudo -H -u www-data php occ maintenance:data-  
fingerprint
```

9. Administración del servidor.

9.1. Renovación automática de certificados.

La versión de **Certbot** que hemos instalado (la última existente a la hora de escribir esta memoria) crea una tarea en el **Cron** del sistema (/etc/cron.d/certbot) que comprueba dos veces al día de si procede la renovación de alguno de los certificados de **Let's Encrypt** instalados. Los certificados caducan a los 90 días, y mediante esta tarea automática se procederá a renovar aquellos certificados que tengan una caducidad de 30 días o menos.

Para ver el código de dicha tarea:

```
root@nube:/etc/cron.d# cat /etc/cron.d/certbot
# /etc/cron.d/certbot: crontab entries for the certbot package
#
# Upstream recommends attempting renewal twice a day
#
# Eventually, this will be an opportunity to validate certificates
# haven't been revoked, etc. Renewal will only occur if expiration
# is within 30 days.
SHELL=/bin/sh
PATH=/usr/local/sbin:/usr/local/bin:/sbin:/bin:/usr/sbin:/usr/bin
0 */12 * * * root test -x /usr/bin/certbot -a \! -d /run/systemd/system && perl -e
'sleep int(rand(43200))' && certbot -q renew
```

Podemos comprobar que la renovación automática funcionará lanzando una prueba con el comando **certbot renew --dry-run**:

```
root@nube:~/scripts# certbot renew --dry-run
Saving debug log to /var/log/letsencrypt/letsencrypt.log

-----
Processing /etc/letsencrypt/renewal/nube.jmsolistoro.net.conf
-----
Cert not due for renewal, but simulating renewal for dry run
Plugins selected: Authenticator apache, Installer apache
Renewing an existing certificate
Performing the following challenges:
http-01 challenge for nube.jmsolistoro.net
Waiting for verification...
Cleaning up challenges

-----
new certificate deployed with reload of apache server; fullchain is
/etc/letsencrypt/live/nube.jmsolistoro.net/fullchain.pem
-----

** DRY RUN: simulating 'certbot renew' close to cert expiry
**          (The test certificates below have not been saved.)

Congratulations, all renewals succeeded. The following certs have been renewed:
/etc/letsencrypt/live/nube.jmsolistoro.net/fullchain.pem (success)
** DRY RUN: simulating 'certbot renew' close to cert expiry
**          (The test certificates above have not been saved.)
-----

IMPORTANT NOTES:
- Your account credentials have been saved in your Certbot
  configuration directory at /etc/letsencrypt. You should make a
  secure backup of this folder now. This configuration directory will
  also contain certificates and private keys obtained by Certbot so
  making regular backups of this folder is ideal.
```

Al final de la comprobación se nos recomienda que hagamos copias de seguridad regularmente de la carpeta **/etc/letsencrypt**, ya que ahí se guardan, entre otras configuraciones de

Let's Encrypt, las credenciales de la cuenta para la renovación de los certificados. Esto ya lo hacemos gracias a las copias de seguridad diarias que hemos configurado en la **Sección 8**.

9.2. Envío automático de información del servidor por correo electrónico.

Puede ser conveniente, para administrar un servidor remoto, que recibamos cada cierto tiempo un correo electrónico con información sobre el estado de la máquina. En un servidor de archivos y trabajo en grupo, una de las cosas importantes a controlar es el espacio libre.

Aunque podamos establecer una cuota por cada usuario, eso no nos asegura que no vayamos a rebasar el espacio total disponible, a no ser que especifiquemos una cuota por usuario tal que el nº de usuarios x cuota < espacio total disponible. Tal escenario es poco recomendable ya que no podríamos añadir más usuarios, y probablemente unos usuarios se quedarían sin espacio mientras otros tendrían espacio de sobra asignando una cuota fija.

En un escenario en el que puede cambiar el número de usuarios, y haya usuarios con cuotas altas o en la práctica ilimitadas, es recomendable estar pendiente del espacio libre en el servidor para tomar acciones cuando disminuya por debajo de un umbral predeterminado.

La información del espacio disponible podría llegarnos diariamente por correo electrónico para tener esa información en nuestro buzón de entrada cada mañana. Se propone enviar por correo electrónico la información del espacio ocupado por *Nextcloud* (carpeta */var/www/html/nube*) y el espacio libre global en el servidor.

El *script* que ejecutaremos diariamente, de nombre */root/scripts/free_space_report.sh*, será:

```
#!/bin/bash
du -hs /var/www/html/nube > /root/space_report.txt
df -h |grep /dev/sda1 >> /root/space_report.txt
cat /root/space_report.txt | mail -s "[nube.jmsolistoro.net] Free space report"
nube@jmsolistoro.net
```

Tenemos que crear ese fichero y darle permisos de ejecución (*chmod +x /root/scripts/free_space_report.sh*). Añadiremos la tarea al **Cron**, usando el comando *crontab -e*, quedando como sigue (el código añadido en este punto está en rojo):

```
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow  command
MAILTO="nube@jmsolistoro.net"
```

```
# NEXTCLOUD cronjob cada 15 minutos
*/15 * * * * /root/scripts/nextcloud-cron.sh
# Free space report
30 8 * * * /root/scripts/free_space_report.sh
```

La tarea que hemos añadido se ejecutará cada día a las 08:30 horas.

10. Conclusiones.

A medida que los usuarios y las empresas han ido creando más y más contenido digital, surgió la necesidad de almacenarlo y compartirlo de manera organizada, rápida y eficiente. Esta tendencia, que comenzó a finales del siglo XX y crece exponencialmente, ha alcanzado dimensiones gigantescas en los últimos diez o quince años.

De las soluciones anticuadas, caras y poco eficientes de almacenamiento y compartición de contenido digital existentes al principio del milenio hemos pasado a sofisticados sistemas de almacenamiento y trabajo en grupo basados en la nube, ofrecidos de manera gratuita (o casi) por grandes empresas y corporaciones norteamericanas como Google, Microsoft, Dropbox, etc.

Por las razones ya expuestas en el Capítulo 1 (Introducción y motivación), la adopción masiva de estos sistemas comporta la pérdida efectiva del control del contenido por parte de los usuarios, empresas y organismos que los adopten. Su contenido.

El valor de dicho contenido, con el que además comercian las empresas que proporcionan el alojamiento, es tan alto, que han convertido a estas empresas en líderes tecnológicos mundiales.

Las consecuencias de la pérdida del control del contenido para los usuarios, como ciudadanos libres, es la pérdida de su privacidad. Y la pérdida de la privacidad es el primer paso necesario, pero no suficiente, para la pérdida de la libertad.

Las empresas y organismos públicos (o privados) con sede en Europa, no pueden usar estos servicios y tener la seguridad de cumplir las nuevas leyes europeas de privacidad y seguridad, la última de ellas la *General Data Protection Regulation*, que las empresas americanas no tienen que cumplir en su país. Aparte de que todas estas empresas estadounidenses son sospechosas de colaborar activamente con su gobierno en el espionaje masivo a ciudadanos del resto del mundo.

Hasta hace relativamente poco tiempo, no había disponible una alternativa barata y confiable a las plataformas privadas norteamericanas, pero como ha podido comprobarse a lo largo de esta memoria, hoy día ya no es así.

Mediante el proceso descrito en esta memoria, es posible desplegar un sistema de almacenamiento, gestión y compartición de contenido digital y trabajo en grupo, basado en *Nextcloud*, autogestionado, y por un coste inicial irrisorio (si sacamos de la ecuación el *know-how* del técnico implicado): poco más de 10€ al mes por el alquiler de un servidor web para alojar *Nextcloud* (localizado en Europa), más 20€ al mes la máquina con el espacio suficiente para guardar las copias del primer servidor. Y el tiempo necesario para hacerlo podría ser cuestión de minutos si automatizamos el proceso de instalación y configuración.

Todo ello permitiría a las empresas, organismos y particulares europeos que usen este sistema, retener el control de su contenido, aumentar la productividad, cumplir las regulaciones europeas en materia de privacidad y satisfacer los más altos estándares de seguridad corporativa.