Fun with migasfree

Alberto Gacías

October 09, 2015

Contents

A Patricia.

El deber de un ciudadano no es creer en ninguna profecía del futuro, sino actuar para realizar el mejor futuro posible.

—Richard Stallman.

Part I

Introducción

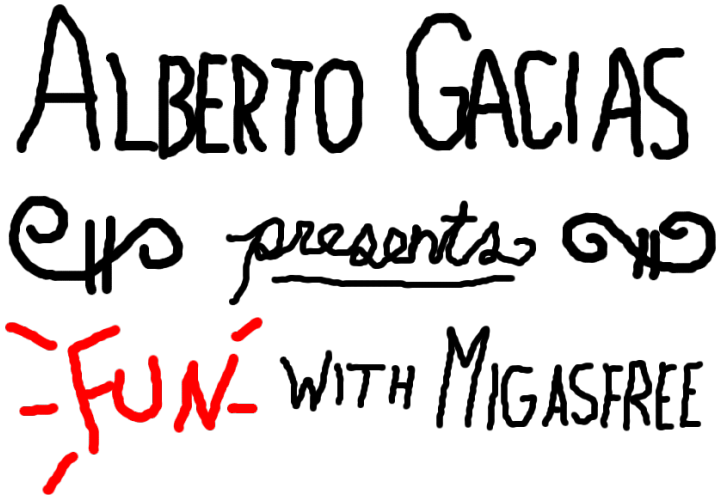
CHAPTER 1

Acerca de este libro

1.1 Licencia y Copyright

Fun with migasfree

1.2 Presentación



Hola. Soy Alberto Gacías. Bienvenidos al primer capítulo de "Alberto Gacías presenta diversión con migasfree". Durante las próximas páginas, usted y yo vamos a explorar el dinámico mundo de la migasfreelogía. [1]

[1] Recordando a Sheldon Cooper en "Fun with Flags" en la serie The Big Bang Theory.

Migasfree es una de las herramientas que estamos utilizando con éxito en AZLinux, el proyecto de migración a escritorio libre del Ayuntamiento de Zaragoza.

Se ocupa principalmente del proceso de la liberación de software y de la posterior auditoría de los cambios producidos en los equipos como consecuencia de esa liberación.

Este software se ha hecho indispensable en nuestro día a día, y creo que es una buena solución para personalizar y administrar escritorios de forma eficaz.

Este libro te introducirá en el uso de migasfree y lo escribo a medida que mejoramos el software, con lo que si te lo descargaste hace tiempo quizás ya esté obsoleto, tenlo en cuenta.

1.3 Estructura

Introducción

En esta primera parte, repasaremos la Gestión de la Configuración del Software. Conocer los aspectos básicos de este proceso de la Ingeniería del Software te dará una visión de conjunto que considero esencial porque es precisamente aquí donde se integra migasfree.

Te explicaré las dificultades a las que un administrador de escritorios va a encontrarse y cómo se pueden sortear de forma sencilla basándome en la experiencia adquirida en AZLinux.

Podrás conocer la historia, características y componentes que utiliza migasfree.

Primeros pasos

Aqui te enseñaré a instalar y probar un servidor y cliente migasfree con la configuración mínima para que puedas verlos en funcionamiento cuanto antes.

Guía de uso

Te permitirá conocer tanto el cliente como el servidor migasfree más en detalle.

Puesta en producción

Se tratarán los aspectos a tener en cuenta si quieres utilizar migasfree en un entorno de producción, asi como las FAQs y la resolución de problemas.

Ajustes

Detalla los ajustes necesarios para configurar correctamente tanto el servidor migasfree como los clientes.

Empaquetado

Contiene intrucciones para empaquetar migasfree en cualquier Distribución.

Anexos

Contiene la API de migasfree, la bibliografía, referencias y licencia de este libro.

1.4 A quién va dirigido

Este libro puede serte útil si eres administrador de escritorios (y/o servidores) y quieres personalizar y administrar de forma eficaz tus equipos manteniendo la integridad de los sistemas.

1.5 Agradecimientos

Detrás de cada proyecto hay personas que lo hacen posible, manteniendo, animando, corrigiendo, colaborando, apoyando...

Deseo expresar en primer lugar mi gratitud a Eduardo Romero. Me dio el estímulo necesario para liberar la primera versión de migasfree, haciendo visible este proyecto en internet. También aportó la primera y única donación que ha recibido migasfree (aunque fuera por una apuesta perdida, no se lo tuve en cuenta y fue muy bien recibida).

A Jose Antonio Chavarría, compañero de fatigas (y alegrías), también me siento agradecido. Ha sido y es piedra angular en migasfree. Ha mejorado sustancialmente el proyecto reescribiendo el código spaguetti a buen código [2], aportando ideas y soluciones. Me tranquiliza cuando quiero correr en exceso, y es el guardián de la simplicidad de migasfree.

A Jesús González por su empeño en crear equipos de trabajo donde las personas nos sentimos a gusto trabajando.

Y a todo el grupo de Asistencia a Usuarios del Ayuntamiento de Zaragoza, especialmente al equipo de Software Libre. Disfruto trabajando con ellos y me hacen reír a diario.

Fig. 1.1: Grupo de Software Libre del Ayuntamiento de Zaragoza.

1.6 Acerca de mí

De joven me atraía la programación. Estudié electrónica, y aquí me enseñaron a programar en código máquina el microcontrolador 8751.

Aún me gusta cacharrear con transistores, condensadores, circuitos integrados, leds... y enseñar lo poco que recuerdo de todo aquello a Jesús. A los dos nos gusta jugar con arduino scratch y s4a.

Trabajé como electrónico mis primeros años laborales y poco después, con el boom de la informática personal, empecé a desarrollar aplicaciones de todo tipo.

Actualmente trabajo como técnico informático en el equipo de Software Libre del Ayuntamiento de Zaragoza desarrollando y manteniendo AZLinux, el escritorio libre que usamos los trabajadores municipales.

Parte de mi tiempo libre lo dedico a desarrollar migasfree.

Me encanta mirar el cielo en las noches de verano de Peñiscola, la cerveza, las migas, y los huevos rotos.

[2] Proceso conocido muy localmente como chavarrización.

También me gusta escuchar música, el olor a tierra mojada y que me hagan reír con cualquier tontada.

Amo a Patricia, y a Jesús nuestro hijo.

1.7 Nota del autor

Algunos han criticado a las personas que entregamos parte de nuestro tiempo en producir software libre. El argumento se basa en que nuestra aportación hace que se eliminen puestos de trabajo o que algunas empresas no pueden hacer negocio por competencia desleal. Dicen estar hartos de gente que "trabajamos gratis" y que les "quitamos" el sustento.

No puedo estar de acuerdo. En primer lugar porque el software libre no es un asunto económico sino que sencillamente lo que plantea es una cuestión de **libertad**.

Pienso que los modelos de negocio obsoletos deben adaptarse y evolucionar hacia nuevas formas de generar riqueza, creando nuevas relaciones entre productor y consumidor. Los modelos de negocios basados en software libre a menudo nos indican el camino a seguir, ya que están estableciendo estas nuevas relaciones y obteniendo la confianza y el reconocimiento del consumidor, y no precisamente por cuestiones económicas. ¿No desean esto las empresas para sí?.

En cuanto a que producir algo que otros obtienen gratis no genera puestos de trabajo, pienso que es falso. Sólo hace falta fijarse en como las tecnologías de interconexión, protocolos y servicios de accesibilidad de la red Internet ha generado, y seguirá generando, infinidad de puestos de trabajo. Estoy convencido que Internet no sería ni siquiera una sombra de lo que es si estas tecnologías se hubieran patentado, cerrado y/o explotado económicamente.

Creo que el movimiento del software libre es, junto con otros, una esperanza para que el Conocimiento vuelva a ser producido por la sociedad y para la sociedad, en contraposición al Conocimiento creado, comercializado y controlado por determinadas organizaciones y que en ocasiones causa un perjuicio a la sociedad.

1.8 Enlaces

Versión html: http://fun-with-migasfree.readthedocs.org/en/master/

Versión pdf: https://media.readthedocs.org/pdf/fun-with-migasfree/master/fun-with-migasfree.pdf

Codigo Fuente: http://github.com/migasfree/fun-with-migasfree

Poyecto migasfree: http://migasfree.org

Twitter: @migasfree @albertogacias

CHAPTER 2

Gestión de la Configuración Software

> Nada es permanente a excepción del cambio
>
> —Heráclito de Éfeso.

Estamos acostumbrados a actualizar periódicamente nuestras aplicaciones: los sistemas se hacen obsoletos rápidamente, aparecen nuevas tecnologías, hay errores que son resueltos, surgen nuevas necesidades. Sin importar en qué momento del ciclo de vida del sistema nos encontremos, el sistema cambiará, y el deseo de cambiarlo persistirá a lo largo de todo el ciclo de vida. [1]

Por tanto, el cambio en el software es **inevitable** y **deseable** que ocurra.

Es inevitable porque los desarrolladores cometemos errores y es mediante una modificación como los corregimos. A éste tipo de cambios los llamamos **correctivos**.

Por otro lado el cambio es deseable ya que a menudo queremos incorporar nuevas funcionalidades al software o mejorar aquellas que ya existían. Mediante los cambios **evolutivos** es como mejoramos el software.

El cambio genera **confusión** e incertidumbre y se produce desde que concebimos, construímos y tambien mientras mantenemos un proyecto software.

El gran reto reside precisamente en gestionar de forma controlada dichos cambios usando alguna estrategia que los favorezca y facilite.

De esto trata precisamente la Gestión de la Configuración Software (GCS), un proceso de la Ingeniería del Software que identifica, hace seguimiento y controla cada uno de los cambios que se producen en los sistemas.

2.1 Objetivo

El objetivo de la GCS es **conservar la integridad** de los sistemas frente a los cambios.

Un sistema será íntegro frente al cambio si:

- Mantiene correctamente las **relaciones** entre los distintos cambios a medida que se van produciendo (el típico problema de dependencias entre elementos).
- Permite la **auditoría** de cambios (conocimiento del estado de un sistema al que se le han ido aplicando cambios sucesivamente).

[1] Primera ley de la Ingeniería de Sistemas, Software Configuration Management, Bersoff, Henderson & Siegel, Prentice-Hall, 1980

2.2 El proceso

El proceso de la GCS es un conjunto de actividades que nos permitirá garantizar dicha integridad, y que podemos resumir en:

- Petición de cambio
- Cambio
- Liberación.

2.2.1 Peticion de cambio

Cuando se nos reporta un error o una petición de mejora, lo primero que hacemos es identificar el elemento de configuración software (ECS) al que se refiere.

Un ECS es cualquier objeto software sometido a la GCS. Puede ser un manual de usuario, una especificación, un conjunto de datos para realizar tests, una aplicación, una librería, incluso las herramientas que se usan para realizar dichos cambios, etc.

Una vez identificado el ECS se registra la petición de cambio.

Las herramientas típicas para registrar y hacer el seguimiento del cambio son los denominados `gestores de proyectos` (Redmine, Bugzilla, Tracker, etc.)

Cada petición de cambio es analizada más tarde pudiendo ser aceptada o rechazada. Si es rechazada se avisa al informador y se cierra la petición; si es aceptada se asigna la petición a alguien para que realice dicho cambio.

2.2.2 Cambio

El cambio es la actividad que modifica el ECS, generando una nueva versión del ECS.

En esta actividad se utilizan un conjunto muy diverso de herramientas, desde procesadores y editores de texto, sistemas de control de versiones, entornos de desarrollo integrados (IDE), depuradores, compiladores...

2.2.3 Liberación

La liberación es la actividad de situar la nueva versión del ECS generada, en un repositorio o almacén para que posteriormente los clientes del ECS puedan acceder a él e instalarlo.

2.3 Elemento de Configuracion Software

Si observamos cómo los diferentes proyectos de Software Libre realizan la GCS, vemos que realizan las actividades mencionadas liberando finalmente el código fuente del proyecto en internet.

Estos proyectos trabajan con distintos tipos de ECS (.png, .txt, .py, .c, .bin, etc.) usando los Sistemas de Control de Versiones junto con las plataformas de desarrollo colaborativo como sourceforge.net, github.com, etc.

Este código fuente será posteriormente compilado por los mantenedores de las Distribuciones GNU/Linux (Fedora, Red Hat, Debian, Ubuntu, etc.) realizando su propia GCS, pero a diferencia de los primeros, las Distribuciones GNU/Linux sólo trabajan sobre un único tipo de ECS: **el paquete**, donde introducirán el programa ya compilado.

Este simple hecho permite garantizar la integridad frente a los cambios de forma eficaz y sencilla como veremos a continuación.

2.3.1 El paquete

Un paquete es un contenedor que encapsula un conjunto de ECS liberados por un determinado proyecto, junto a su metainformación.

Contendrá, por tanto, el programa compilado para una determinada Distribución y arquitectura, más un conjunto amplio de información como puede ser:

- El autor del programa.
- La dirección del repositorio del proyecto.
- La versión del ECS.
- La arquitectura.
- El nombre y dirección e-mail del empaquetador.
- La fecha de empaquetado.
- El Nombre del equipo en que se produjo el empaquetado.
- Una descripción corta del contenido del paquete.
- Una descripción larga.

Pero además suelen incluir:

- Código a ejecutar antes y después de:
  - Instalar.
  - Actualizar.
  - Eliminar el paquete.
- Dependencias con otros paquetes.

Una vez que un mantenedor de una Distribución GNU/Linux ha creado un paquete, lo libera poniéndolo en un repositorio público a disposición de los clientes.

2.3.2 Puesta en producción

Los encargados de aplicar los cambios son los programas denominados **gestores de paquetes** tales como yum, zypper ó apt.

Un gestor de paquetes es un programa que permite poner en producción los cambios que han sido liberados en los repositorios.

La actualización de un equipo se realiza comparando las versiones de los paquetes instalados con los paquetes de los repositorios públicos, detectando los que han aumentado su versión, resolviendo sus dependencias y finalmente, si no hay conflictos, obtienen desde los repositorios los paquetes necesarios.

Una vez han descargado los paquetes, dan órdenes a los **backends** (rpm, dpkg, etc.) para que se produzca la desinstalación de los paquetes antigüos y la instalación de los nuevos.

Los backends abren el paquete, y grosso modo:

1. Extraen los ficheros del programa copiándolos en el sistema, y ejecutando además el código programado para antes y después de la actualización.

2. La metainformación es extraída del paquete y se almacena en la base de datos del backend.

Decía Ian Murdock, fundador de Debian, que el gran aporte del software libre a la industria ha sido precisamente la invención del sistema de paquetería (paquete, repositorio, gestor de paquetes).

Y no es para menos, ya que este sistema nos proporciona los dos requisitos necesarios que garantizan la integridad frente a los cambios:

1. El control de dependencias, mediante el gestor de paquetes.
2. La auditoría, mediante las consultas a la base de datos del backend.

Note: Si estás acostumbrado a instalar programas mediante el típico "./configure, make, install", tienes que ser consciente de que estás rompiendo la integridad frente a los cambios, ya que la base de datos del backend no es actualizada con este procedimiento. Todo lo que no sea instalar programas mediante el gestor de paquetes o el backend rompe la integridad.

A continuación se muestra una tabla con las operaciones más comunes que puedes emplear sobre algunos de los gestores de paquetes más conocidos:

Operaciones Gestores	apt	yum	zypper
Actualizar listado de paquetes disponibles	apt-get update	yum check-update	zypper refresh
Actualizar sistema	apt-get upgrade	yum update	zypper update
Instalar paquete	apt-get install <pkg>	yum install <pkg>	zypper install <pkg>
Borrar paquete	apt-get remove <pkg>	yum erase <pkg>	zypper remove <pkg>
Buscar paquete	apt-cache search <pkg>	yum list <pkg>	zypper search <pkg>
Buscar repositorio de un paquete	apt-cache madison <pkg>	yum whatprovides <pkg>	zypper what-provides <pkg>

Veamos ahora las operaciones más comunes que puedes emplear sobre los backends dpkg y rpm:

Operaciones Backends	dpkg	rpm
Instalar fichero de paquete	dpkg -i <file .deb>	rpm -ivh <file .rpm>
Borrar paquete	dpkg -r <pkg>	rpm -e <pkg>
Paquete que instala el fichero <file>	dpkg -S <file>	rpm -qf <file>
Ficheros del paquete	dpkg -L <pkg>	rpm -ql <pkg>
Descripción del paquete	dpkg -s <pkg>	rpm -qi <pkg>
Paquetes instalados	dpkg -l	rpm -qa

CHAPTER 3

Administrando escritorios

> En todo lo que nos rodea y en todo lo que nos mueve debemos advertir que interviene en algo la casualidad.
>
> —Anatole France.

En el capítulo anterior hemos hablado de la GCS y de cómo las Distribuciones GNU/linux utilizan el sistema de paquetería para garantizar la integridad frente al cambio.

Si tienes un equipo doméstico, todos los cambios producidos y liberados por los distintos proyectos y que hayan sido empaquetados y liberados por tu Distribución GNU/Linux, serán instalados convenientemente con el simple hecho de dar la orden al gestor de paquetes para que actualice tu sistema.

Ahora bien, en una organización donde se requiera administrar los escritorios esto no es suficiente, veamos el porqué.

3.1 La personalización

La primera dificultad importante a la que se va a enfrentar un administrador va a ser el de la personalización.

Imagina que tienes que migrar y administrar 1000 equipos a GNU/Linux y que tienes en tu red un servicio NTP, requiriéndose que todos tus escritorios estén con la hora sincronizada con este servicio.

Vas a tener que personalizar el cliente NTP en todos tus escritorios.

Una manera que se suele utilizar es instalar en un equipo una Distribucion GNU/Linux desde un DVD, editar el fichero de configuración del cliente NTP y configurar la IP del servidor donde se encuentra el servcio NTP. Después, puedes crear una imagen del disco duro con un sistema de clonado como pueda ser Clonezilla y clonar uno a uno los equipos usando dicha imagen.

Con este método la personalización inicial reside en dicha imagen, pero sigamos imaginando...

Un día a mitad de migración recibes un correo y lees:

> Alberto: El servicio NTP dejará de dar servicio a partir del día 10. En su lugar vamos a disponer de un nuevo servicio llamado QueHoraEs que es mucho mejor porque ...

En este momento ya estarás pensando en los 400 equipos que tienes migrados y te echarás las manos a la cabeza porque es evidente que este sistema de personalización no es adecuado.

Note: La personalización inicial es muy sencilla de realizar pero un cambio en la personalización puede darse en cualquier momento, y tienes que estar preparado para poder realizarlo.

3.1.1 Gestores de Sistemas

Afortunadamente, existen unas herramientas denominadas Gestores de Sistemas (Systems Management Systems) que pueden ayudarnos en la administración de los escritorios.

Algunos de estos Gestores de Sistemas se centran en la adquisición del estado de los equipos como Nagios, y otros permiten automatizar tareas mediante la ejecución de código en los equipos de manera centralizada como Zenworks, Landscape, chef, puppet, cfengine, ansible.

Los Gestores de Sistemas están muy influenciados por las iniciativas realizadas en los sistemas de gestión de redes de telecomunicaciones, pudiendo realizar una o un conjunto de las siguientes tareas:

- Inventario Hardware.
- Monitoreado de disponibilidad de Servidores y mediciones.
- Inventario Software e instalación de Software.
- Gestión de Antivirus y anti-malware.
- Monitoreado de las actividades de los usuarios.
- Monitoreado de la capacidad de los sistemas.
- Gestíon de Seguridad.
- Gestión de almacenamiento.
- Monitoreado de la utilización y capacidades de la Red.

Éstas tareas podemos clasificarlas de acuerdo a FCAPS, un modelo y marco de trabajo de red de la gestión de telecomunicaciones de ISO para la gestión de redes. FCAPS es un acrónimo de Fault, Configuration, Accounting, Performance, Security (Falla, Configuración, Contabilidad, Desempeño, Seguridad) que son las categorías en las cuales el modelo ISO define las tareas de gestión de redes.

> **Fault** (Fallas): Es un evento que tiene un significado negativo. Su objetivo es reconocer, aislar , corregir y registrar fallos. Puede utilizar análisis de tendencias para predecir errores. Cuando se detecta un fallo o evento se envía una notificación.
>
> **Configuration** (Configuración): En el proceso de gestión de la configuración, las operaciones diarias son monitoreadas y controladas.
>
> Los objetivos de la gestión de la configuración son:
>
> - Recolectar información.
> - Modificar la configuración.
> - Generación de reportes
> - Gestión de cambios.
>
> Los cambios de Hardware y Software son controlados por este proceso:
>
> - Actualización, Instalación y eliminación de programas.
> - Actualización, Instalación y eliminación de equipamiento (impresoras, scaners,
>
> memoria, etc.)
>
> Este proceso debe tener en cuenta:
>
> - Permitir acceso rápido a la información de la configuración.
> - Facilitar la configuración remota de los dispositivos.
> - Proporcionar un inventario actualizado de Software y Hardware.

- Simplificación de la configuración de dispositivos.
- El seguimiento de cambios a la configuración.

Accounting (Contabilidad): Su objetivo es reunir las estadísticas de los usuarios.

Performance (Desempeño). Recolentando y analizando los datos de rendimiento el estado general de los sistemas pueden ser monitorizado. Las tendencias pueden avisar de fallos de capacidad o de cuestiones relacionadas con la fiabilidad de los sistemas antes de que en estos ocurran. Umbrales de rendimiento pueden ser establecidos para lanzar alarmas que serían controlada por la gestión de fallos habitual. Las alarmas se pueden clasificar atendiendo al grado de severidad.

Security (Seguridad). Se encarga de controlar el acceso a recursos de red. La seguridad de los datos puede ser conseguida con la autenticación, cifrado y permisos principalmente.

Note: Migasfreee atendiendo a FCAPS tiene capacidades de Faults, Configuration y Accounting.

Un ejemplo de funcionamiento típico de un Gestor de Sistemas que incorpore tareas de *Configuration* usaría un lenguaje que especificaría a qué estado se quiere llevar a los equipos, no cómo llegar a ese estado, en nuestro caso sería algo parecido a esto:

- asegúrate de que el paquete ntp-client está desinstalado
- asegúrate de que el paquete quehoraes-client está instalado
- asegúrate de que el fichero de configuración de quehoraes-client es el

mismo que el que está en el servidor.

Periódicamente, los clientes se conectarían al servidor para obtener este código que será ejecutado mediante el intérprete propio del Gestor de Sistemas instalado en el cliente.

Este sistema permite automatizar aquellas tareas que realizan a menudo los administradores de sistemas, y aunque algunos Gestores de Sistemas se las ingenian para llevar un control de versiones, mantienen una base de datos independiente a la de los backends de los gestores de paquetes, dejando en entredicho todo lo relativo a la integridad de los sistemas.

3.1.2 Empaquetando la personalización

En AZLinux usamos otro método: Empaquetamos siempre la personalización.

Para el caso del cliente "QueHoraEs" crearíamos el paquete azl-quehoraes-client [1] con la siguiente información:

- Dependencias: quehoraes-client
- Obsoletos: ntp-client
- En el script de postinstalación escribiríamos el siguiente código:

  En el fichero de configuración del cliente QueHoraes, modificar el valor de la entrada "server=" por la IP del servidor QueHoraEs

¡Listo! Con esto queda garantizada la integridad frente al cambio de la personalización aprovechándonos de la integridad que nos proporciona el sistema de paquetería de nuestra Distribución GNU/Linux.

Una vez empaquetada nuestra personalización se hace relativamente sencillo realizar cualquier cambio posterior en ella. Pero crear un paquete desde cero para personalizar una Distribucion GNU/Linux no es tan fácil, no tanto por la creación del paquete en sí, sino porque la personalización requiere de los conocimientos suficientes sobre el sistema GNU/Linux y sobre la propia aplicación que se personaliza.

[1] En AZLinux empleamos como nombre de paquete el prefijo "azl-" más el nombre del paquete que queremos personalizar.

Note: Empaquetar la personalización nos asegura la integridad de los sistemas frente a sus cambios.

Date cuenta que no es necesario ningún Gestor de Sistemas para instalar dicha personalización. Sólo necesitas el Gestor de Paquetes, y éste siempre lo tienes disponible en cualquier Distribución GNU/linux.

3.1.3 Niveles de personalización

Las aplicaciones suelen incorporar dos niveles de personalización:

- La del usuario
- La del sistema (para todos los usuarios del sistema)

La personalización del usuario es prioritaria a la del sistema siempre y cuando ésta última no sea obligatoria.

Es conveniente conocer si la aplicación que vas a configurar incorpora la personalización a nivel de sistema, ya que ésta es la que se tendrá que configurar.

En los casos en que las aplicaciones sólo tengan la configuración a nivel de usuario, o en los casos en los que se requiera, tendrás que recorrer todos los usuarios para aplicar la personalización a cada uno de ellos.

3.2 La liberación

Es el segundo problema importante con el que vas a tener que lidiar.

Por un lado debes independizarte de los repositorios públicos de tu Distribución GNU/Linux por el simple motivo de que no puedes permitir que el control de los cambios que se instalarán en tus máquinas lo tenga tu Distribución GNU/Linux en vez de tu organización.

¿Imaginas que habría pasado en AZLinux cuando OpenSuSE sustituyó OpenOffice por LibreOffice?. Cuando los usuarios hubieran encendido las máquinas a las 8:00 de la mañana, se iniciaría la actualización a LibreOffice automáticamente pudiéndose producir muchas incidencias, ¿funcionaría todo? ¿No es mejor probar LibreOffice en tu organización antes de que se instale en todos tus equipos?

Tener la posibilidad de deshacer un cambio que se haya determinado como no deseado es importante.

Tienes que decidir por tí mismo el software que deben tener tus usuarios y por tanto debes tener los gestores de paquetes configurados contra tus propios repositorios de paquetes y gestionarlos de alguna manera.

Ademas, es conveniente que puedas planificar a quién y cuándo se deben liberar dichos cambios.

Imagina nuevamente el ejemplo de la sustitución de OpenOffice por LibreOffice, estaríamos hablando de una actualización de cerca de 500 MB por equipo que multiplicado por todos los equipos de una organización podría resultar mucho tráfico de red.

Una ventaja de planificar la liberación es que permite distribuir poco a poco los cambios, de tal manera, que si hay errores afectará inicialmente a muy pocos equipos permitiendo actuar de manera más relajada para corregir cualquier incidencia.

Por todo esto, y como los repositorios estándar de las Distribuciones no tienen ningún mecanismo de planificación de la liberación, es por lo que decidimos desarrollar migasfree, extendiendo el concepto de repositorio de paquetes al concepto de repositorio de paquetes dinámico y planificable.

3.3 Repositorio Migasfree

Un repositorio de migasfree es simplemente un repositorio estándar más la capacidad de poder especificar, de forma centralizada, cuándo y quién accede a ese repositorio.

Veamos como actúa migasfree en lo relativo a los repositorios:

1. Los cambios que se quieren liberar son empaquetados y subidos a un servidor migasfree.

2. Se crea un repositorio lógico con los paquetes subidos y se establece a quién (atributos de usuario + equipo) y en qué momento se deben aplicar dichos cambios. Esto no es más que un registro en la tabla de repositorios de la base de datos de migasfree.

3. El servidor migasfree crea un repositorio físico (idéntico al de cualquier Distribución GNU/Linux) con dichos paquetes, utilizando las herramientas estándar de creación de repositorios (createrepo para paquetería RPM o dpkg-scanpackages para paquetería Debian).

4. Cuando un cliente migasfree se conecta al servidor envía sus atributos al servidor.

5. El servidor consulta los Registros Lógicos para determinar, en función de esos atributos enviados, la lista de los repositorios físicos que tiene el cliente a su disposición y se los envía al cliente.

6. El cliente migasfree configura, la lista de los repositorios físicos recibidos desde el servidor en el Gestor de Paquetes (Por esto decimos que los repositorios migasfree son dinámicos)

7. A continuación el cliente migasfree da instrucciones al Gestor de Paquetes para que se produzca la eliminación, instalación y actualización de los paquetes desde los repositorios físicos.

3.4 La GCS en tu organización

En el capítulo anterior, hemos visto el proceso de la GCS en los distintos proyectos de software libre y también en las Distribuciones GNU/Linux.

Pues bien, en una organización también debe realizarse el proceso de la GCS.

En AZLinux realizamos nuestra propia GCS y vemos cómo de nuevo se repiten las mismas actividades: petición de cambio, cambio y liberación.

Usamos dos tipos de peticiones de cambio:

- **Actualización de aplicaciones**. Si recibimos una petición para actualizar, por ejemplo, Mozilla Firefox, descargamos desde los repositorios de la Distribución la versión deseada, la probamos en laboratorio, registrando cualquier información relevante en la petición de cambio. Finalmente, si todo es correcto, se liberan los paquetes a través de un repositorio migasfree, planificando su distribución (ver A en figura 3.2)
- **Personalización de aplicaciones**. Se produce cuando llega p.e., una petición de cambio para añadir un motor de búsqueda de sinónimos a Mozilla Firefox. Introducimos entonces en un paquete propio de AZLinux (azl-firefox), el código que instala dicho motor de búsqueda y liberamos dicho paquete en un repositorio de migasfree planificando su distribución (ver B en figura 3.2).

Las herramientas que usamos actualmente en cada actividad son:

- En la petición de cambio:
  - Gestor de proyectos: Redmine
- En el cambio:
  - Editor de textos: Geany
  - IDE: Ninja-ide

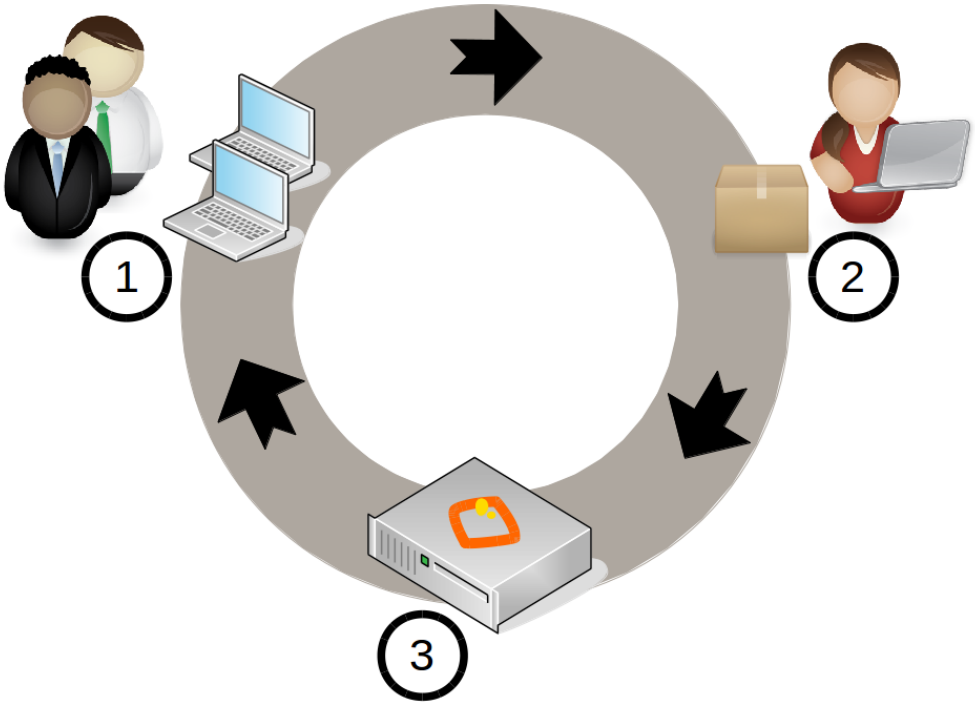


Fig. 3.1: Proceso GCS en tu organización con migasfree.

1. Un usuario hace un **petición** de cambio.
2. Un desarrollador programa el **cambio** de la configuración software dentro de un paquete y lo sube a un servidor migasfree.
3. La **liberación** es realizada por el servidor migasfree a los ordenadores requeridos.

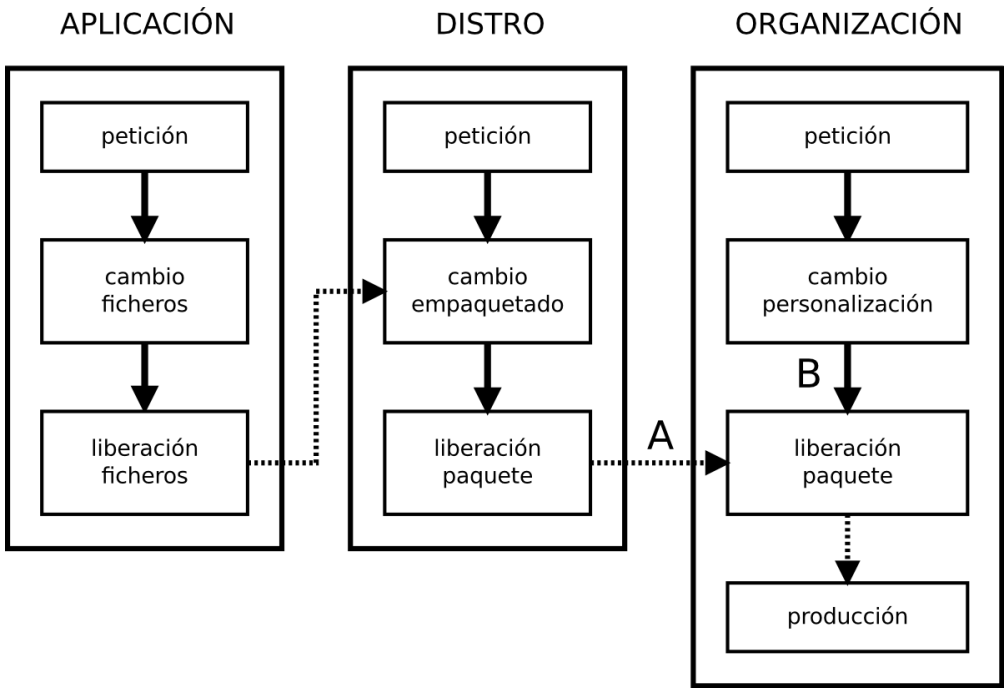


Fig. 3.2: Procesos de la Gestión de la Configuración Software

  - Sistema de control de versiones: Git
  - Gestor de proyectos: Redmine
- En la liberación:
  - Gestor de sistemas: Migasfree
  - Gestor de proyectos: Redmine

Note: Migasfree nos proporciona de manera centralizada conocer el estado, no sólo del servidor migasfree, sino de cada uno de los equipos registrados en el servidor, convertiéndose en una herramienta ideal para hacer una auditoría tanto de software como de hardware.

3.4.1 Beneficios

Los principales beneficios que obtendrá tu empresa, como resultado de aplicar una GCS, serían:

1. Reducción del coste de los servicios de desarrollo y mantenimiento.
2. Optimización del uso de los recursos.

Y para ti, como administrador:

1. Dispondrás de equipos más estables.
2. Vas a pasar de ser un administrador que se echa las manos a la cabeza ante cualquier cambio a ser un administrador favorecedor del cambio, ya que dispones de las herramientas para hacer el seguimento y control de los cambios.
3. Y, en última instancia, vas a mejorar sustancialmente la resolución de incidencias.

CHAPTER 4

Características de migasfree

Las cosas no se dicen, se hacen, porque al hacerlas se dicen solas.

—Woody Allen.

4.1 El nacimiento de migasfree

En el año 2005 todos grupos políticos del Ayuntamiento de Zaragoza manifestaron por unanimidad en pleno de gobierno municipal apoyar las políticas de uso de Software Libre y, en concreto, el fomento de los programas de SL en el entorno de escritorio del funcionario municipal. La Dirección General de Ciencia y Tecnología asume, inicia y potencia este importante reto. [1]

Este proyecto se planificó en tres etapas:

- Primera: Migrar a aplicaciones que presentaban un impacto bajo sobre usuarios y técnicos en el SO actual.
- Segunda: Migrar la plataforma ofimática Microsoft Office 97 por la suite libre OpenOffice.
- Tercera: Sustituir el SO Windows XP por un sistema operativo basado en Linux. Esta etapa se inició en 2008 y todavía sigue abierta.

Para iniciar la tercera etapa se tuvieron que realizar los primeros prototipos de lo que llegaría a ser la primera version de AZLinux. En estos prototipos la personalización se realizaba manualmente en un equipo cuya imagen del disco duro nos servía para clonarla en otros equipos y hacer las pertinentes pruebas.

En aquel tiempo aprendimos a empaquetar y empezamos a introducir nuestra personalización en nuestros propios paquetes. La ventaja frente a la personalización manual era muy significativa.

Con los primeras migraciones reales, nos surgió la necesidad de actualizar nuestros paquetes y despues de probar sin éxito Zenworks for Linux, decidimos crear nuestros propios repositorios de paquetes. Quisimos emular lo que ya estabamos haciendo con los escritorios XP, esto es, distribuir software basándonos en el contexto al que pertenecía un usuario en nuestro LDAP. Con un poco de scripting bash en Mayo de 2009 implementamos lo que serían unos repositorios dinámicos que se configuraban en el cliente en función del contexto.

Esto fue ,sin duda, una gran idea pero la gestión de estos repositorios dinámicos era manual y muy propensa a errores.

La gestión de estos repositorios dinámicos recayó en mí, por lo que decidí simplificarla inmediatamente y crear el primer prototipo de migasfree. Dos semanas de programación, en horas no laborales, fueron suficientes para presentar a mis compañeros de trabajo un prototipo, que fue puesto en producción en Junio de 2009.

Note: Una de las ventajas de trabajar con software libre es la facilidad con la que puedes crear proyectos ya que puedes mezclar, como si de piezas de puzzle fueran, diferentes componentes sin preocuparte

[1] Eduardo Romero Moreno, Migración Escritorio Software Libre, 2011

en exceso del tema de las licencias. Un ejemplo de esto ha sido la incorporación de la funcionalidad de captura del hardware en los equipos. Utilicé el comando lshw y unas pocas líneas de código para adaptarlo a la base de datos de migasfree.

4.2 Versiones

El primer prototipo solo trabajaba con paquetería rpm y gestor de paquetes yum, y el código bash que se ejecutaba en el cliente se generaba en el servidor.

Despues de usar migasfree un tiempo en producción vimos que podría ser un buen sistema para otras organizaciones, y mis compañeros me dieron el impulso necesario para publicar el codígo, y así durante el verano de 2009, reorganice los menús, limpié un poco el código, e hice que migasfree pudiera trabajar con distintas versiones de SO y de sistemas de paquetería. Fue publicado en github en abril de 2010 y bautizado como "migasfree with fried eggs", porque mis compañeros decían que el logotipo se parecía a un huevo frito, ¡Qué sabrán ellos de Arte!.

En Noviembre de 2011, Jose Antonio Chavarría desarrollador de AZLinux reescribe y publica el cliente migasfree. Realizó tambien grandes cambios en la estructura del servidor. Tuvimos que definir la API con la que el cliente y el servidor debían comunicarse. Usamos claves asimétricas para dotar de seguridad al sistema. Esta nueva versión fue denominada "migasfree no trans" supongo que por incorporar un código más "limpio", por decirlo de alguna manera.

Poco a poco fuímos dotando al sistema de nuevas funcionalidades, y para principios de 2013 Jose Antonio Chavarría cambió la navegación y aspecto del servidor. Esta nueva versión fue denominada "migasfree with chocolate".

En febrero de 2014 liberamos la versión 4 del servidor (migasfree grape). Esta versión hace uso de bootstrap con el fin de dotar a la aplicación de un diseño web adaptable a distintos dispositivos. Además incorpora distintas mejoras de todo tipo y utiliza la última versión estable de Django, la 1.6.2. Actualmente esta es la versíón que utilizamos en AZLinux.

4.3 Características

- Migasfree es simple, y hacemos esfuerzos por mantenerlo así. Tendemos a lo que denominamos gestión cero, es decir, procuramos que la gestión de añadir nuevas entradas en migasfree no requiera ninguna tarea administrativa.
- Está basado en la arquitectura cliente / servidor.
- Es Seguro. Las comunicaciones entre cliente y servidor están firmadas con claves asimétricas.
- Es adaptable. Puedes programar las propiedades para adaptarlas a tus necesidades.
- Es Software Libre licenciado bajo la GNU Public License.
- Captura de datos. Almacena tanto el inventario software y hardware de los equipos, permitiendo hacer consultas sobre ellos. Almacena también información de los equipos tales como sus atributos, actualizaciones, migraciones que se han realizado, etc.
- Consultas. Puedes programar consultas contra la base de datos de migasfree.
- Gestión de errores. Los errores que se producen en los equipos son enviados al servidor y almacenados permitiendo hacer su seguimiento.
- Gestión de fallas. Puedes programar código que será ejecutado en los clientes con el fin de obtener información de los equipos.
- Alertas. Permite conocer en tiempo real el estado del sistema facilitando al administrador su trabajo.
- Estadísticas.

4.4 Principales componentes empleados

- Django un framework de desarrollo web.
- Servidor web Apache. Puedes emplear otro si quieres.
- Lenguaje de programación Python.
- Base de datos Posgresql. Puedes usar otras.
- Intérprete de comandos Bash.
- Sistemas de paquetería como APT ó RPM.
- Información Hardware: Lshw.
- Bootstrap un framework para desarrollo web.
- Flot una librería gráfica.

Part II

Primeros pasos

CHAPTER 5

Probando migasfree

La unidad es la variedad, y la variedad en la unidad es la ley suprema del universo.

—Isaac Newton.

Si bien puedes instalar el servidor migasfree en distintas distribuciones, en este capítulo voy a explicarte como instalarlo sobre Debian 7 Wheezy.

El objetivo de este capítulo es que dispongas rápidamente de un servidor y un cliente migasfree totalmente funcional, por eso no me voy a extender en explicaciones.

Si decides usar otra Distribución GNU/linux de la recomendada tendrás que conseguir los paquetes apropiados. Accede a http://migasfree.org/repo/dists para ver si tu Distribución se encuentra aquí. En caso negativo puedes generar los paquetes como se indica en *Empaquetando migasfree*. Ten en cuenta que las instrucciones de éste capitulo pueden variar segun la Distribución que elijas.

Note: Usa una máquina virtual de virtualbox realizando la instalación mínima por red de Debian 7 para ver el funcionamiento de migasfree y familiarizarte con él antes de poner a *Migasfree en producción*

5.1 Instalando el servidor

Para añadir el repositorio que contiene los paquetes necesarios para debian 7, ejecutaremos la siguiente instrucción:

```
# echo "deb http://migasfree.org/repo debian7 PKGS" > /etc/apt/sources.list.d/migasfree.list
```

Actualizamos las listas de paquetes:

```
# apt-get update
```

A continuación instalamos el paquete python-django con la versión 1.6.11-1 que es la última versión soportada por el servidor migasfree y lo retenemos a dicha versión.

```
# apt-get install python-django=1.6.11-1
# apt-mark hold python-django
```

Finalmente, instalaremos el paquete migasfree-server:

```
# apt-get install migasfree-server
```

Note: Al instalar el paquete del servidor migasfree se añade al sistema el fichero `/etc/apache2/conf.d/migasfree.conf`. Este fichero contiene la configuración del servidor web.

Note: Al instalar el paquete del servidor migasfree se crea el usuario `migasfree` en Postgresql con password `migasfree` y se añade al fichero `/etc/postgresql/9.1/main/pg_hba.conf` la línea `'local all migasfree password'` para permitir al usuario migasfree autenticarse mediante password. Recuerda que para poner en producción el servidor deberás cambiar la contraseña de éste usuario tal y como se indica en *Migasfree en producción*.

5.2 Comprobando el servidor

En un navegador web accede a la dirección del servidor. Si todo ha ido bien verás la figura 5.1.

Fig. 5.1: Acceso al servidor migasfree.

Haz login con el usuario "admin" y password "admin" y verás algo parecido a la figura 5.2. Observa como arriba a la derecha pone `alertas 0`. Esto nos indica que todo esta bien.

5.3 Instalando el cliente

5.3.1 Instalando el paquete migasfree-client

Ahora instala el cliente migasfree sobre la misma máquina donde has instalado el servidor. Para ello actualiza la lista de paquetes e instala el paquete migasfree-client:

```
# apt-get update
# apt-get install migasfree-client
```

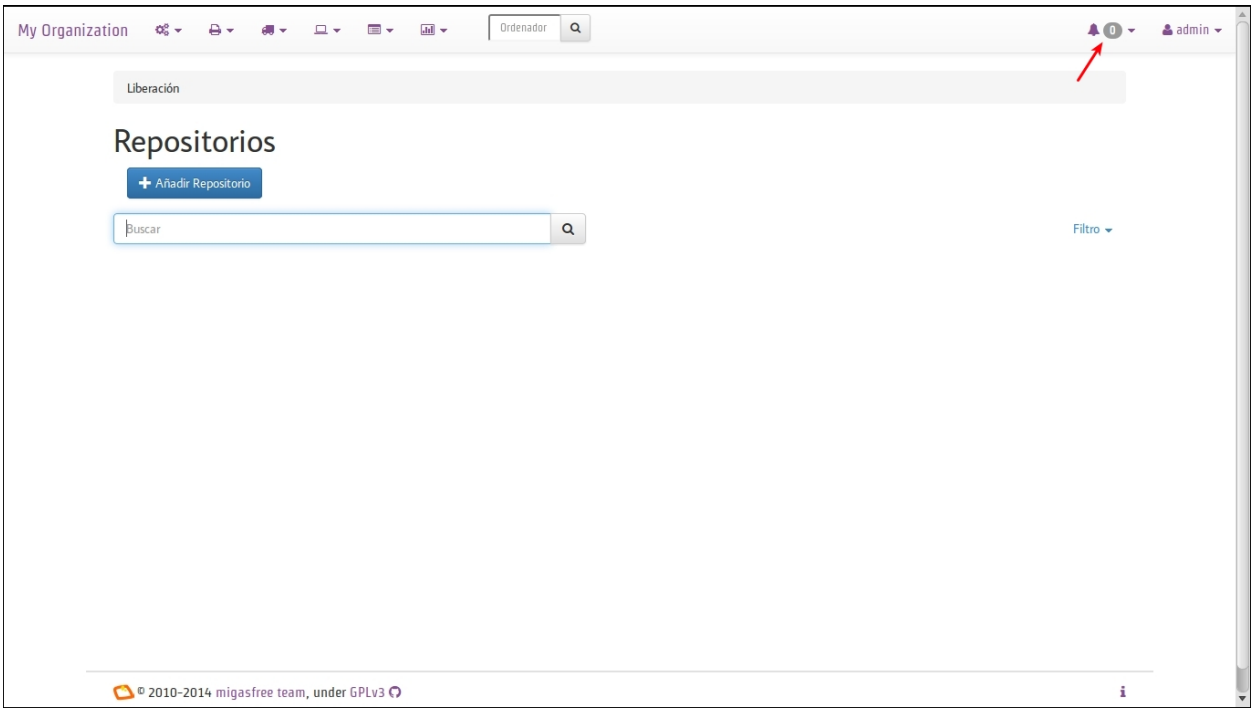


Fig. 5.2: Estado del servidor con 0 alertas.

5.3.2 Registrando el cliente

Ejecuta el comando:

```
# migasfree -u
```

te devolverá una salida parecida a esta:

```
root@debian7:~# migasfree -u
Sesión gráfica no detectada
Versión de migasfree client: 3.1

Opciones de ejecución:
    Versión: debian-7.4
    Servidor: 192.168.92.133
    Proxy: None
    Certificado SSL: None
    Package Proxy Cache: None
    Depuración: False
    Nombre del ordenador: debian7
    GUI detallado: True
    Usuario gráfico: root
    PMS: apt-get

Autoregistrando ordenador...
¡Clave /root/.migasfree-keys/migasfree-client.pri creada!
¡Clave /root/.migasfree-keys/migasfree-server.pub creada!

****************** Conectando al servidor migasfree... ********************
**************************** Correcto

********************** Obteniendo propiedades... ************************
**************************** Correcto
```

```
************************* Evaluando atributos... *************************
VER: debian-7.4

SET: ALL SYSTEMS

IP: 192.168.92.133

NET: 192.168.92.0/24

PCI: 8086:1237~Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] ...

PLT: Linux

HST: debian7

USR: root~root


************************* Subiendo atributos... **************************
***************************** Correcto

************************** Ejecutando fallas... **************************
LOW_HOME_PARTITION_SPACE:
LOW_SYSTEM_PARTITION_SPACE:

************************** Subiendo fallas... ****************************
***************************** Correcto

************************* Creando repositorios... ************************
***************************** Correcto

************* Obteniendo los metadatos de los repositorios... **************
Des:1 http://ftp.es.debian.org wheezy Release.gpg [1.672 B]
Ign http://migasfree.org debian7 Release.gpg
Des:2 http://ftp.es.debian.org wheezy-updates Release.gpg [1.571 B]
Obj http://security.debian.org wheezy/updates Release.gpg
...
Des:11 http://ftp.es.debian.org wheezy-updates/main Translation-en [14 B]
Descargados 16,3 MB en 15seg. (1.025 kB/s)
Leyendo lista de paquetes... Hecho
***************************** Correcto

*********************** Desinstalando paquetes... ************************
***************************** Correcto

****************** Instalando paquetes obligatorios... *******************
***************************** Correcto

************************ Actualizando paquetes... ************************
DEBIAN_FRONTEND=noninteractive /usr/bin/apt-get --assume-yes --force-yes ...
Leyendo lista de paquetes...
Creando árbol de dependencias...
Leyendo la información de estado...
0 actualizados, 0 se instalarán, 0 para eliminar y 0 no actualizados.

***************************** Correcto

***************** Subiendo el inventario del software... *****************
***************************** Correcto

************************* Operaciones completadas ************************
root@debian7:~#
```

5.4 Comprobando el estado del servidor

Comprueba los datos que se han recogido accediendo al servidor con tu navegador web.

- Fíjate ahora que en las `Alertas` tendrás 2 `Notificaciones` (figura 5.3):
  - La primera te notifica que el ordenador `1` ha dado de alta la plataforma `Linux`
  - La segunda notificación te dice que el ordenador `1` ha añadido la version `debian-7.x`

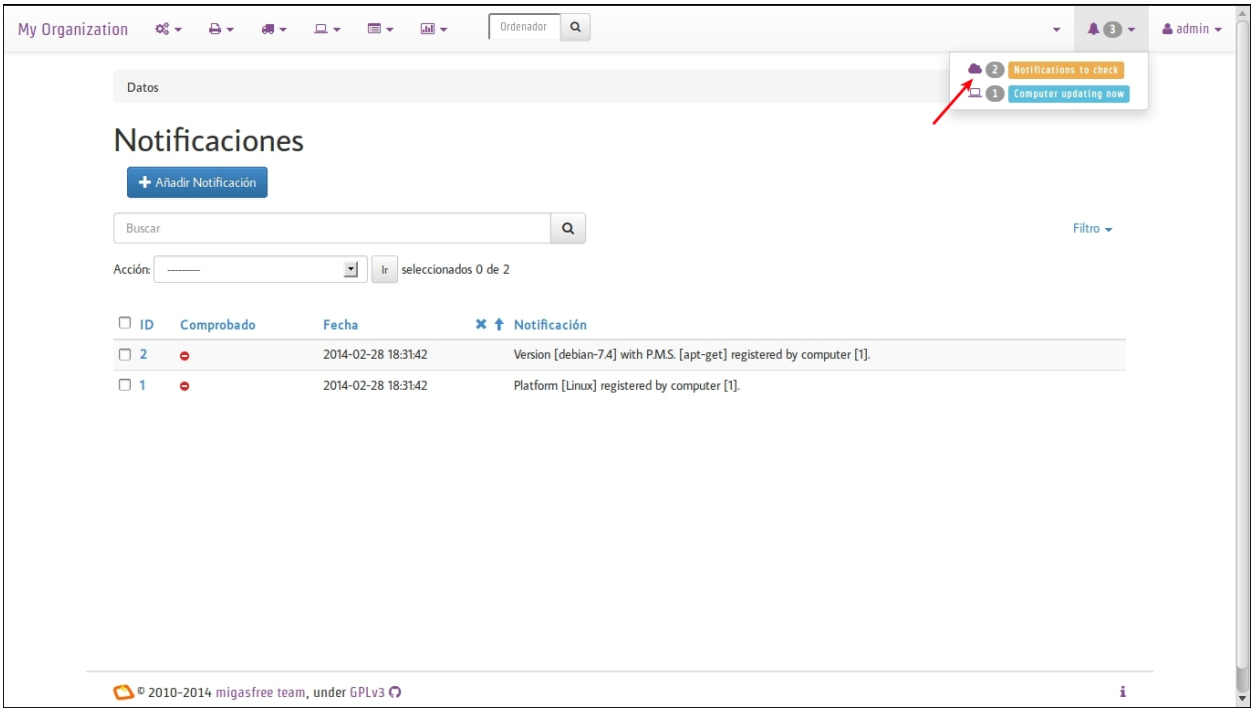


Fig. 5.3: Notificaciones.

- Accede a `Datos - Ordenadores` y observa: (figura 5.4)
  - Los datos del ordenador `1` (pulsando en el número 1)
  - Su `login`, para ver los atributos que ha enviado el cliente.
  - Su `hardware`.

¡Enhorabuena! Has instalado un servidor migasfree y has registrado en él tu primer ordenador.

En el siguiente capítulo vas a aprender a hacer el cambio de configuración software al estilo migasfree.

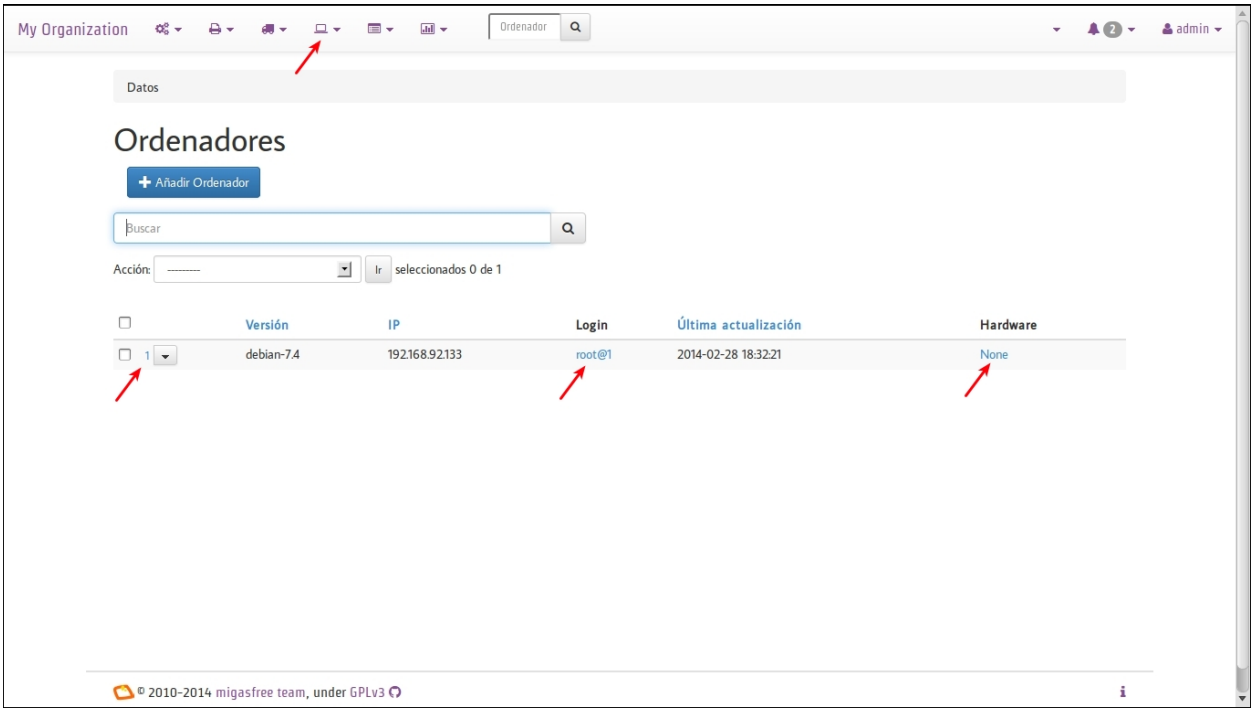


Fig. 5.4: Ordenadores.

CHAPTER 6

Configurando software al estilo migasfree

> No esperes hasta que las condiciones sean perfectas para comenzar, el empezar hace las condiciones perfectas.
>
> —Alan Cohen.

En este capítulo vas a aprender a configurar el servidor migasfree al estilo migasfree.

Quizás no sea el ejemplo más acertado porque vas a configurar sólo un servidor migasfree, pero imagina un escenario donde tienes X centros de trabajo y te interesa tener un servidor migasfree con la misma configuración en cada centro para dar servicio a sus clientes. Uno de estos servidores bien podría ser el que admistrara al resto de servidores.

El objetivo de este capítulo es que veas todo el proceso de la Gestión de la Configuración Software en conjunto.

6.1 Al estilo tradicional

Imagina que te llega una petición de cambio para modificar, en todos los centros, el nombre de la organización de los servidores migasfree con el nombre de tu empresa.

Miras la documentación de migasfree y concluyes que tienes que crear el fichero `/etc/migasfree-server/settings.py` y añadir la siguiente variable:

```
MIGASFREE_ORGANIZATION = "ACME"
```

Decides acceder a cada uno de los equipos por ssh, crear el fichero, reiniciar el servidor apache y olvidarte del tema.

Ahora bien, si estás de vacaciones ¿podría responder fácilmente a las cuestiones siguientes tu compañero de trabajo?

- ¿Qué cambios se han realizado en un determinado equipo desde el 1 de enero? ¿Quién los hizo? ¿Y cúando se realizaron todos esos cambios?
- ¿Qué equipos tienen el cambio propuesto?

Este método es sencillo y rápido, pero difícilmente tu compañero va a poder reponder estas cuestiones de manera eficaz, aunque hayas registrado muy bien tu trabajo. La integridad frente al cambio no está garantizada con éste método.

A continuación te propongo otra forma de realizar los cambios de configuración. Se basa en utilizar el empaquetado para trasladar los cambios a los equipos conservando la integridad del sistema.

Asumo que tienes un gestor de proyectos como Redmine donde vas a registrar las peticiones de cambio (o al menos que hagas como que lo tienes) y que has completado con éxito el capítulo anterior. Todos los comandos de este capítulo los vas a ejecutar como root en el equipo que hayas utilizado en el capítulo anterior.

6.2 Tu primer cambio de configuración

El primer cambio sobre un Elemento de Configuración Software (ECS) es el que te llevará más trabajo porque exige la creación de un paquete.

6.2.1 Petición

Imagina que te llega la siguiente la petición de cambio que registras y aceptas en el gestor de proyectos:

Gestor de proyectos:

Registro: Sustituir el nombre de la organización `My organization` de los servidores migasfree por el de `ACME`

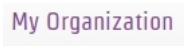


Fig. 6.1: Nombre de la organización.

Lo primero que haces es identificar al ECS que afecta, es decir, cuál es el paquete que debe ser modificado. Como no existe todavía un paquete sobre el que actuar, asigna la petición de cambio a un desarrollador (Qué suerte, siempre te toca a tí) y registra en la petición de cambio:

Gestor de proyectos:

Registro: Crear el paquete **acme-migasfree-server**

Asignado a: *desarrollador*.

6.2.2 Cambio

Empaquetado

Cómo desarrollador tienes que crear el paquete de configuración `acme-migasfree-server`. Si nunca has creado un paquete no te preocupes, para facilitarte las cosas y que puedas avanzar centrándote en el proceso GCS aqui tienes el fuente del paquete.

En la máquina virtual ejecuta:

```
# wget http://www.migasfree.org/repo/book/acme-migasfree-server_1.0-1.tar.gz
# tar -xzvf acme-migasfree-server_1.0-1.tar.gz
```

Observa como modificamos el nombre de la organización

```
# less acme-migasfree-server/etc/migasfree-server/settings.py
```

Note: En los *Ajustes del servidor migasfree* puedes ver el conjunto de ajustes que se pueden emplear para adaptar el servidor a tus necesidades.

Y observa tambien que en la postinstalación del paquete se ejecutará el comando `service apache2 reload` cuando se produzca la configuración del paquete:

```
# less acme-migasfree-server/debian/postinst
```

Ya tienes el fuente del paquete. Ahora genera el paquete, pero para ello antes debes tener instalado el paquete devscripts:

```
# apt-get install devscripts
```

Y ahora sí, genera el paquete:

```
# cd acme-migasfree-server
# /usr/bin/debuild --no-tgz-check -us -uc
# cd ..
```

Felicidades, el cambio está empaquetado en `acme-migasfree-server_1.0-1_all.deb`

Subiendo al servidor el cambio

Usa este comando para subir el paquete generado al servidor.

```
# migasfree-upload -f acme-migasfree-server_1.0-1_all.deb
```

- Introduce usuario: admin
- Contraseña: admin
- Version: debian-7.4
- Ubicacion: acme

La salida que te devolverá el comando `migasfree-upload` será:

```
root@debian7:~# migasfree-upload -f acme-migasfree-server_1.0-1_all.deb
Versión de migasfree upload: 3.1
Usuario para subir ficheros al servidor: admin
Contraseña del usuario:
Versión a la que subir en el servidor: debian-7.4
Ubicación a la que subir en el servidor: acme

Opciones de configuración:
    Servidor: 192.168.92.133
    Proxy: None
    Depuración: False
    Versión: debian-7.4
    Ubicación: acme
    Usuario: admin
    Fichero: acme-migasfree-server_1.0-1_all.deb
    Fichero normal: None
    Crear repositorio: True

Obteniendo las claves de empaquetador...
¡Clave /root/.migasfree-keys/migasfree-server.pub creada!
¡Clave /root/.migasfree-keys/migasfree-packager.pri creada!
```

Finalmente asigna la petición de cambio a un liberador (sí, otra vez vas a ser tú) y registra en la petición:

Gestor de proyectos:

Registro: Creado paquete **acme-migasfree-server_1.0-1_all.deb**

Asignado a: *liberador*

Felicidades, has realizado un cambio de configuración y lo has almacenado en el servidor migasfree.

6.2.3 Liberación

Ahora vas a ver el punto de vista del encargado de liberar los cambios:

Accede mediante navegador web a tu servidor. Observa que en `Alertas` tienes `1 paquete huérfano` (Figura 6.2).

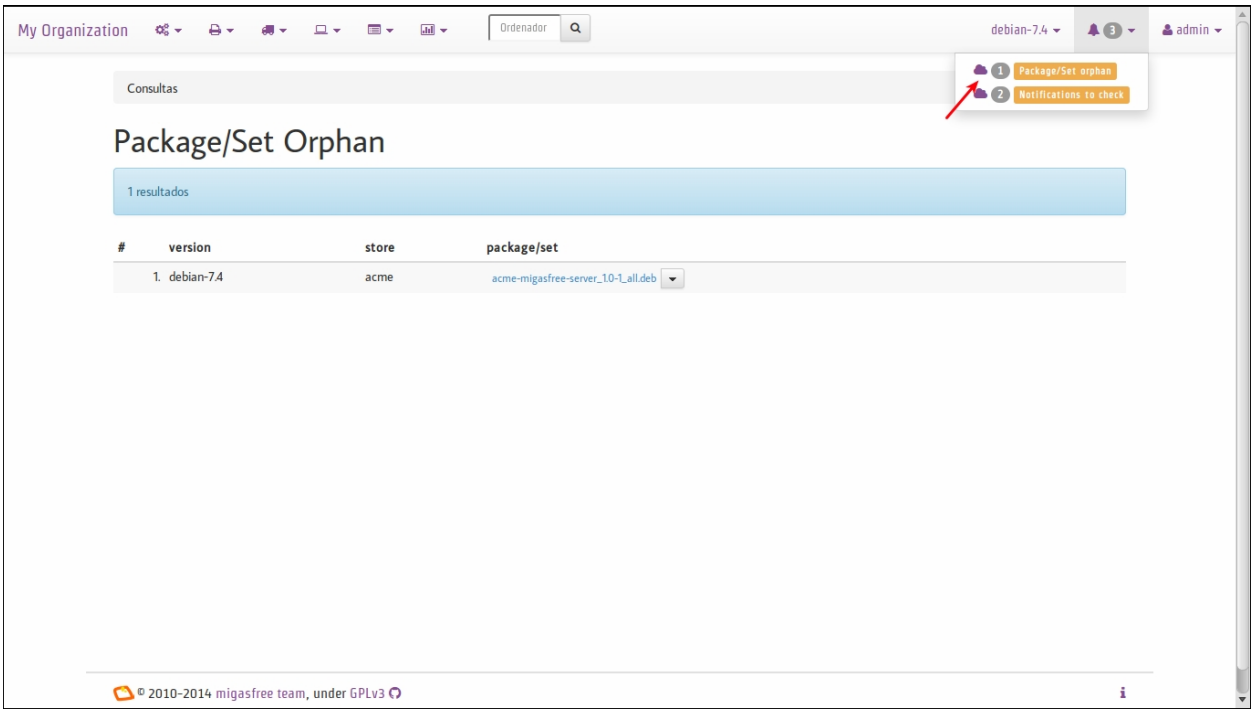


Fig. 6.2: Paquetes huérfanos.

Note: Todos los paquetes que se han subido al servidor y no están asignados en ningún repositorio se denominan `huérfanos`.

Note: Observa tambien que a la izquierda de las `Alertas` aparece un desplegable con las versiones que existen en el servidor. Esto permitirá al usuario que se ha autenticado en el servidor migasfree ver los datos relativos a dicha versión. Selecciona, por tanto, la version `debian-7.x`

Liberando el cambio de configuración

Ahora, vas a liberar el cambio creando un nuevo Repositorio. Para ello, ve a `Liberación` (pulsando sobre el icono del camión) y accede a `Repositorios`. Luego pulsa en `Añadir nuevo repositorio` e introduce estos datos:

- Nombre = `PRINCIPAL`
- Version = `debian-7.4`
- Fecha = `Hoy`
- Ahora despliega la sección `Paquetes`
- Paquetes/Conjuntos = `acme-migasfree-server_1.0-1_all.deb`

En este campo se asignan los paquetes que contendrá este repositorio.

- Paquetes a instalar = `acme-migasfree-server`

  En este campo se escriben los nombres de los paquetes que se instalarán **obligatoriamente** en los clientes.

- Despliega la sección `Atributos`
- Atributos = `SET-ALL SYSTEMS`

  De esta manera indicamos que todos los clientes tendrán acceso a este repositorio.

Guarda el repositorio.

Observa que en `Alertas` ya no tienes ningún paquete huérfano.

Registra y cierra la petición de cambio:

Gestor de proyectos:

Registro: Liberado en Repositorio **PRINCIPAL**.

Petición: *cerrada*.

Aplicando el cambio

Para aplicar el cambio ejecuta el siguiente comando:

```
# migasfree -u
```

Observa en la salida del comando:

```
****************** Subiendo el historial del software... ******************
Diferencia en el software: # 2013-05-19 10:42:33
+acme-migasfree-server-1.0-1
**************************** Correcto
```

Abre el navegador y fíjate que el nombre de la organización ha cambiado (Figura 6.3).

Fig. 6.3: Cambio nombre organización a ACME.

6.3 Tu segundo cambio de configuración

6.3.1 Petición

Te llega la segunda petición de cambio:

Gestor de proyectos:

Registro: Sustituir de nuevo el nombre de la organización en los servidores migasfree ya que el nombre correcto es Acme Corporation.

Como siemrpre, identificas primero el ECS al que afecta el cambio: En este caso es a `acme-migasfree-server`. En la petición de cambio asignas al desarrollador y registras:

Gestor de proyectos:

Registro: Modificar el paquete **acme-migasfree-server-1-0.1**

Asignado a : *desarrollador*.

6.3.2 Cambio

Los cambios que se realizan sobre un paquete ya creado suelen ser más sencillos de realizar porque simplemente se modifica el paquete.

Empaquetado

Edita el fichero del paquete `acme-migasfree-server/etc/migasfree-server/settings.py` y modifica la variable `MIGASFREE_ORGANIZATION`:

```
MIGASFREE_ORGANIZATION = "Acme Corporation"
```

Edita el fichero del paquete‘‘acme-migasfree-server/debian/changelog‘‘ para registrar el cambio realizado. Tendrás que **añadir** estas líneas **al principio del fichero**:

```
acme-migasfree-server (1.0-2) unstable; urgency=low

  * Change organitation to Acme Corporation

 -- Alberto Gacías <alberto@migasfree.org>  Sun, 19 May 2013 13:09:00 +0200
```

Presta atención a:

- La versión del paquete **(1.0-2)**.
- Sustituir **tu nombre y dirección de correo**.
- Modificar la **fecha y hora**.

> **Note:** El formato que se utiliza en el changelog en paquetes debian es muy estricto. Ten cuidado con los espacios, retornos de carro y fechas.

Ahora generamos el paquete:

```
# cd acme-migasfree-server
# /usr/bin/debuild --no-tgz-check -us -uc
# cd ..
```

Observa que se ha generado el mismo paquete pero con la versión `1.0-2`

```
# root@debian7:~# ls -la *.deb
-rw-r--r-- 1 root root 2286 may 19 10:37 acme-migasfree-server_1.0-1_all.deb
-rw-r--r-- 1 root root 2338 may 19 13:25 acme-migasfree-server_1.0-2_all.deb
```

Subiendo al servidor el cambio

```
# migasfree-upload -f acme-migasfree-server_1.0-2_all.deb
```

- Introduce usuario: admin
- Contraseña: admin

- Version: debian-7.4
- Ubicacion: acme

Gestor de proyectos:

Registro: Creado paquete **acme-migasfree-server_1.0-2_all.deb**

Asignado a: *liberador*

6.3.3 Liberación

Liberando el cambio de configuracion

Observa como aparece de nuevo un `paquete huérfano` en `alertas` y que corresponde a `acme-migasfree-server_1.0-2_all.deb`

Accede a `Liberación - Repositorios` y edita el repositorio `PRINCIPAL`. Añade a `Paquetes/Conjuntos` el paquete `acme-migasfree-server_1.0-2_all.deb`

Guarda el repositorio.

Registra y cierra la petición de cambio:

Gestor de proyectos:

Registro: Liberado en Repositorio **PRINCIPAL**.

Petición: *cerrada*.

Aplicando el cambio

Ejecuta de nuevo:

```
# migasfree -u
```

Observa en la salida de este comando el cambio de software:

```
****************** Subiendo el historial del software... ******************
Diferencia en el software: # 2013-05-19 21:51:28
+acme-migasfree-server-1.0-2
-acme-migasfree-server-1.0-1
**************************** Correcto
```

Comprueba si el cambio se ha aplicado.

Fig. 6.4: Cambio nombre organización a Acme Corporation.

6.4 Auditoría

Ahora sí que vas a responder las siguientes cuestiones de manera centralizada desde el servidor migasfree:

6.4.1 ¿Qué cambios se han producido en el ordenador 1 y cuándo?

Accede a Datos - Ordenadores, accede al equipo 1 y mira el final del campo historial de software de la sección Software:

```
# 2013-05-19 21:47:18
+acme-migasfree-server-1.0-1

# 2013-05-19 21:51:28
+acme-migasfree-server-1.0-2
-acme-migasfree-server-1.0-1
```

El signo (-) indica paquete desinstalado y el signo (+) paquete instalado.

6.4.2 ¿Qué se cambió, quién y cuándo lo hizo?

Esta información está en el paquete como metainformación. Para acceder a ella accede a Liberación - Paquetes. Despliega el menú de la derecha del paquete acme-migasfree-server_1.0-2_all.deb y pulsa en Información del paquete.

Aquí podrás ver el registro de los cambios (entre otra información):

```
****CHANGELOG****
acme-migasfree-server (1.0-2) unstable; urgency=low

  * Change organitation to Acme Corporation

 -- Alberto Gacías <alberto@migasfree.org>  Sat, 19 May 2013 08:32:00 +0200

acme-migasfree-server (1.0-1) unstable; urgency=low

  * Change organitation to ACME

 -- Alberto Gacías <alberto@migasfree.org>  Sat, 18 May 2013 08:32:00 +0200
```

6.4.3 ¿Qué equipos tienen el cambio acme-migasfree-server-1.0-2?

Ve a Consultas - Ordenadores con el paquete.... Escribe en el campo Paquete acme-migasfree-server-1.0-2 y obtendrás el resultado.

6.5 Conclusión

Aunque requiera de un esfuerzo inicial *empaquetar la configuración de las aplicaciones*, los beneficios que obtendrás justifican sobradamente el uso de este método, ya que dispondrás de sistemas más estables, te permitirá hacer el seguimento y control de los cambios y mejorarás la resolución de incidencias.

6.5.1 Beneficios de crear paquetes de configuración

- La configuración permacece encapsulada.
- Las configuraciones puede revertirse fácilmente.
- Facilita las pruebas antes del despliegue.
- Facilita la distribución de las configuraciones de forma segura.

- Proporciona integridad frente a los cambios de la configuración.

6.5.2 Desventajas del empaqueteado de la configuración.

- Cuesta más tiempo que otras alternativas ya que hay que crear los paquetes.

6.5.3 Beneficios de usar migasfree

Utilizar migasfree para la realizar la *Liberación* te permitirá:

- Controlar a quién y a partir de qué momento se deben aplicar los cambios
- Tener una auditoría centralizada:
  - Inventario de Ordenadores
    - Hardware
    - Software (actual e histórico)
  - Inventario de los cambios.

  y algunas cosas más que te serán desveladas en los siguientes capítulos.

CHAPTER 7

Configurando migasfree-client

> La libertad no es poder elegir entre unas pocas opciones impuestas, sino tener el control de tu propia vida. La libertad no es elegir quien será tu amo, es no tener amo.
>
> —Richard Stallman.

En el capítulo anterior nos hemos centrado en cómo se realiza el proceso de la GCS.

En este capítulo vas a *configurar el cliente de migasfree* (mediante empaquetado) para que se conecte contra la maquina virtual debian 7 en el que ya tienes un servidor migasfree instalado.

Todos los comandos de este capítulo los vas a ejecutar en otra máquina virtual con Ubuntu instalado y que debes tener en la misma red en la que esté la maquina virtual del servidor.

El objetivo de este capítulo es que conozcas un poco más el empaquetado.

7.1 Instalando migasfree-client en Ubuntu

Para añadir el repositorio que contiene el cliente migasfree para Ubuntu, crea el fichero /etc/apt/sources.list.d/migasfree.list con el siguiente contenido:

```
deb http://migasfree.org/repo ubuntu PKGS
```

Ahora instala el cliente migasfree:

```
# apt-get update
# apt-get install migasfree-client
```

Observa como en el fichero /etc/migasfree.conf que ha instalado el paquete `migasfree-client` no hay, lógicamente, ningún ajuste configurado.

```
less /etc/migasfree.conf
```

A continuación, vamos a configurar este fichero haciendo uso del empaquetado, así que no lo hagas manualmente.

7.2 Obteniendo acme-migasfree-client

Al igual que hiciste con la configuración del servidor puedes bajarte el fuente del paquete que vamos a utilizar de plantilla para configurar el cliente de migasfree.

En la nueva máquina virtual con ubuntu ejecuta:

```
$ wget http://www.migasfree.org/repo/book/acme-migasfree-client_1.0-1.tar.gz
$ tar -xzvf acme-migasfree-client_1.0-1.tar.gz
```

7.3 Adaptando acme-migasfree-client

Accede al directorio acme-migasfree-client y observa su contenido:

```
$ cd  acme-migasfree-client
$ ls -la
total 20
drwxrwxr-x 5 alberto alberto 4096 jun 18 20:54 .
drwxrwxr-x 4 alberto alberto 4096 jun 18 21:04 ..
drwxrwxr-x 3 alberto alberto 4096 jun 18 20:54 debian
drwxrwxr-x 3 alberto alberto 4096 jun 18 20:54 usr
```

7.3.1 Metadatos

Observa el directorio `debian`. Este directorio es el que contiene los metadatos del paquete. Los ficheros más importantes en este directorio son:

- El fichero `control` consiste en un conjunto de campos, representados en un formato común, que permiten al sistema de gestión de paquetes conocer los metadatos del paquete y así poder gestionarlo adecuadamente. Puedes consultar la debian-policy para explorar el conjunto de `datos de control`
- El fichero `changelog` contiene información, en un formato especial, con las modificaciones que se han realizado en cada versión del paquete. Cada vez que se modifica el paquete hay que añadir una entrada en este fichero incrementando la versión y registrando lo que se ha modificado.
- El fichero `copyright` contiene la información sobre los recursos, licencia y derechos de autoría de las fuentes originales del paquete.
- El fichero `rules` contiene las reglas que se utilizan para generar los paquetes a partir de sus fuentes.
- El fichero `install` contiene una lista de ficheros que serán instalados con el paquete

Ahora que conoces el significado de estos ficheros modifícalos cambiando el nombre del paquete `acme-migasfree-client` por `tuempresa-migasfree-client` y pon tu nombre y la fecha actual allí dónde se requiera.

Modifica tambien el nombre del directorio raiz `acme-migasfree-client` por `tuempresa-migasfree-client`

7.3.2 Scripts

Observa ahora los scripts `postinst` y `prerm`. Sus nombres nos indican cuando serán ejecutados por el sistema de gestión de paquetes.

- `postinst` inmediatamente después de que se produzca la instalación del paquete.
- `prerm` justo antes de que se produzca la eliminación del paquete.

Observa ahora el contenido de `postinst` y verás que aquí se hace una llamada al comando `dpkg-divert`. Mediante este comando hacemos lo que se conoce como una desviación de fichero (divert). Mediante la desviación indicamos al sistema de gestión de paquetes que un fichero ya no pertenece a un determinado paquete sino al que nosotros establezcamos.

Así el fichero de configuración `/etc/migasfree.conf`, que pertenece en principio al paquete migasfree-client, hacemos que pertenezca al paquete `tuempresa-migasfree-client` de tal manera que una posible actualización de `migasfree-client` ya no nos afectará. Cada vez que queramos modificar un ajuste del cliente migasfree en `/etc/migasfree.conf` lo haremos a través del fichero `usr/share/divert/etc/migasfree.conf` del paquete `tuempresa-migasfree-client`.

Fíjate tambien que en `prerm` deshacemos esta desviación, para que si desinstalamos el paquete quede todo como estaba.

Modifica ahora el fichero `usr/share/divert/etc/migasfree.conf`. Tendrás que poner el ajuste `Server` con el nombre, o la ip, del servidor migasfree que hemos utilizado anteriormente, y el ajuste `Version` con el nombre de tu distribución, por ejemplo `ACME-1`. El resto de ajustes modifícalos según tus intereses. Una vez hecho esto, y situado en el directorio `tuempresa-migasfree-client`, genera el paquete (debes tener el paquete `devscripts` y `debhelper` previamente instalados).

```
$ /usr/bin/debuild --no-tgz-check -us -uc
```

Con esto tendrás un paquete que configura el cliente migasfree para tu organización. Ahora es momento de instalarlo

```
# dpkg -i tuempresa-migasfree-client_1.0-1_all.deb
```

Observa que al instalar el paquete, dpkg te informa que se añade la desviación de /etc/migasfree.conf. Comprueba ahora que el ajuste `Server` y `Version` son los correctos.

```
# less /etc/migasfree.conf
```

Ahora ya estas preparado para registrar este ordenador en el servidor migasfree.

```
# migasfree -u
```

Comprueba que en el servidor se ha creado la version `ACME-1` y que existe un nuevo ordenador accediendo a la página web del servidor.

Finalmente subimos el paquete a nuestro servidor migasfree con el fin de tenerlo disponible para su liberación a otros escritorios `ACME-1`.

```
# migasfree-upload -f tuempresa-migasfree-client_1.0-1_all.deb
```

- Introduce usuario: admin
- Contraseña: admin
- Version: ACME-1
- Ubicacion: acme

7.4 Ejecución del cliente migasfree

Hasta ahora siempre hemos ejecutado el cliente migasfree desde consola mediante el comando `migasfree -u` como `root`. Ahora vamos a hacer que se ejecute automáticamente cada vez que el usuario abra una sesión gráfica. Para este propósito existe el paquete `migasfree-launcher`.

```
$ wget https://github.com/migasfree/migasfree-launcher/archive/latest.zip
$ unzip latest.zip
$ rm latest.zip
$ cd migasfree-launcher-latest
$ /usr/bin/debuild --no-tgz-check -us -uc
$ cd ..
```

Sube el fichero migasfree-launcher al servidor

```
# migasfree-upload -f migasfree-launcher_1.0-1_all.deb
```

Ahora observa los ficheros que contiene este paquete:

- `etc/sudoers.d/migasfree-launcher` establece los comandos que no requieren password de root para que pueden ser ejecutados desde un usuario cualquiera. Puedes obtener más información sobre la configuración de `sudoers` ejecutando `man sudoers` en un terminal.
- `etc/xdg/autostart/migasfree-launcher.desktop` ejecutará el comando `/usr/bin/migasfree-tray` cuando el usuario inicia sesión gráfica. `migasfree-tray` llamará a `/usr/bin/migasfree-launcher` y éste a su vez a `migasfree --update`

  Puedes aprender más sobre la especificación de los ficheros .desktop en freedesktop.org.

Ahora que ya tienes los paquetes `tuempesa-migasfree-client` y `migasfree-launcher` en el servidor migasfree, crea un repositorio en el servidor y pon estos paquetes en `paquetes a instalar` y asígnale el atributo `SET-ALL SYSTEMS`.

Note: Para aprender mas sobre el empaquetado consulta la Guía del nuevo desarrollador de Debian

---

Note: Para paquetería rpm los metadatos del paquete se especifican en un único fichero llamado `SPEC`. Para aprender más sobre la creación de paquetes rpm puedes consultar rpm.org y la wiki del proyecto fedora.

7.5 Despliegue

A partir de este momento vas a poder administrar fácilmente los escritorios ubuntu de tu organización, de forma generalizada, instalando simplemente estos dos paquetes.

Hay varias formas de realizar esta instalación:

- Bajando los dos paquetes a cada uno de los escritorios e instalándolos mediante el comando `dpkg -i`
- Creando un fichero `/etc/apt/sources.list.d/migasfree.list` con el siguiente contenido:

```
deb http://<myserver>/repo/<version>/REPOSITORIES <store> PKGS
```

  donde sustituirás:

  - `<myserver>` por tu servidor.
  - `<version>` por la versión que pusiste en /etc/migasfree.conf
  - y `<store>` por la ubicación que pusiste al subir el paquete al servidor migasfree con migasfree-upload.

  Una vez creado este fichero ejecuta:

```
# apt-get update
# migasfree -u
```

  y los paquetes se instalarán automáticamente

- Puedes hacer un clon de un equipo donde ya estén instalados estos paquetes utilizando un sistema de clonado como clonezilla. Este es el método que usamos en AZLinux, y nos resulta muy cómodo y rápido ya que en una memoria USB llevamos un clonezilla junto con la imagen clonada de nuestro escritorio consiguiendo instalar un AZLinux en menos de 10 minutos.

- Puedes crear un DVD de tu escritorio tal y como se realiza en el proyecto vitalinux. En concreto tendrías que adaptar el paquete vx-create-iso a tus necesidades. En éste método son los usuarios quienes se bajan la iso del DVD y se instalan ellos mismos el sistema.

Part III

Guía de uso

CHAPTER 8

La configuración del sistema migasfree

> El hombre razonable se adapta al mundo; el irrazonable intenta adaptar el mundo a si mismo.
> Así pues, el progreso depende del irrazonable.
>
> —George Bernard Shaw

En los capítulos anteriores has aprendido a instalar el servidor y el cliente migasfree, así como a crear paquetes. La creación de paquetes no es una tarea trivial, no tanto por su construcción en sí sino por el hecho de que son necesarios amplios conocimientos de los sistemas operativos y de las aplicaciones.

En éste y en los siguientes tres capítulos vas a aprender a adaptar y usar el servidor migasfree.

8.1 Propiedades

En migasfree una `propiedad` es una característica de los equipos o de los usuarios, y que nos servirá para desplegar los paquetes.

Como administrador de migasfree una de las primeras tareas que debes realizar es definir estas propiedades. Debes preguntarte en función de qué características vas a realizar los despliegues. Por ejemplo, ¿te interesa desplegar los paquetes por el HOSTNAME de los equipos? ¿y por subred? ¿Que tal por el grupo al que pertenece el usuario en el LDAP? ¿Y por su contexto LDAP?.

> **Note:** En AZLinux usamos principalmente el contexto LDAP al que pertenece el usuario para desplegar los cambios por los distintos servicios o departamentos de nuestro ayuntamiento, y en menor medida usamos también el HOSTNAME de los equipos.

Una `propiedad` es un código que se programa en un registro de la base de datos de migasfree. Estas propiedades serán ejecutadas en cada uno de los clientes migasfree y su valores de retorno serán devueltos al servidor como `atributos`.

> **Note:** El `atributo` es el valor concreto que toma una `propiedad` al ser ejecutada en un equipo.

Veamos un ejemplo sencillo de todo esto con la propiedad HOSTNAME. Accede a la web de tu servidor migasfree y ve a `Configuración-Propiedades-HST`. Verás en este registro el siguiente código escrito en python.

```
import platform
print platform.node()
```

Si ejectutas `python` en una consola y escribes estas dos líneas verás que python muestra por la salida estandard el nombre de tu equipo.

```
$ python
Python 2.7.3 (default, Apr 10 2013, 05:46:21)
[GCC 4.6.3] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>> import platform
>>> print platform.node()
white
```

En mi caso me ha devuelto `white`, que es el nombre de mi portátil. ¿A que no aciertas de qué color es?

Esto es en definitiva lo que hace el cliente migasfree: obtiene del servidor la propiedad `HST` (su código), la ejecuta y devuelve al servidor el resultado como atributo `HST-white`.

Podrías haber escrito la propiedad tambien en código `bash` simplemente llamando al comando de linux `hostname` o escribiendo `echo $HOSTNAME` (el resultado sería el mismo), pero utilizar código python nos permite, en este caso, usar la misma `propiedad` también para plataformas Windows ó MAC.

8.1.1 Campos de la Propiedad.

Observa cada uno de los campos de la Propiedad:

- **Prefijo**: Es una combinación de tres números o letras. Este prefijo se utiliza para agrupar e identificar los atributos.
- **Nombre**: Denomina a la propiedad.
- **lenguaje de programación**: En el que está escrito el código de la propiedad.
- **Código**: Instrucciones a ejecutar en los clientes para obtener `atributos`.
- **Habilitado**: Indica si está activa la propiedad. Si no está marcado la propiedad no será ejecutada en los clientes.
- **Clase**: Hay cuatro tipos de clases y que nos permiten tratar el valor devuelto por la propiedad de diferentes maneras:
  - **Normal**. El valor devuelto por la propiedad viene con el siguiente formato:

```
<valor>~<Descripción>
```

    ó simplemente como:

```
<valor>
```

  - **Lista**: El valor al ejecutar la propiedad en el cliente es una `lista de atributos` separados por una coma. Puedes ver un ejemplo en la propiedad `PCI`. Su formato es:

```
<valor>~<Descripción>, ...
```

  - **Agrega por la Derecha**: Permite añadir atributos de la siguiente manera: Si el valor devuelto por la propiedad es "CONTEXTO1.CONTEXTO2.MIEMPRESA" el servidor interpreta que el equipo tiene estos tres atributos:
    * MIEMPRESA
    * CONTEXTO2.MIEMPRESA
    * CONTEXTO1.CONTEXTO2.MIEMPRESA

    Se utiliza esta clase para crear atributos relacionados con LDAP.
  - **Agrega por la Izquierda**. Lo mismo que el anterior pero agregando por la izquierda.
    * CONTEXTO1

    * CONTEXTO1.CONTEXTO2
    * CONTEXTO1.CONTEXTO2.MIEMPRESA

- **Automático**: Si este campo está marcado los nuevos atributos será añadidos automáticamente a la base de datos de migasfree. En caso contrario es el administrador de migasfree el responsable de añadir manualmente los atributos para esta propiedad. Se pueden añadir atributos manualmente accediendo a `Datos-Atributos`.

8.1.2 Propiedades específicas

Existen unas propiedades predefinidas que tienen unos objetivos muy concretos y que no pueden ser eliminadas del sistema. Lo más característico de ellas es que **no son ejecutadas en el cliente** sino en el servidor.

- **SET**: Esta propiedad tiene un atributo llamado `SET-ALL SYSTEMS`. Todos los ordenadores tendrán éste atributo sin excepción. Sirve para cuando quieras referirte a **todos** los ordenadores. Por ejemplo, si en un repositorio asignas este atributo, todos los ordenadores tendrán acceso a él. Es habitual usarlo también en la última demora de un calendario. Esta propiedad además se usa internamente para definir *Conjuntos de Atributos*.
- **CID**: Computer Identificator. Esta propiedad generará un atributo que es igual al campo `id` de la tabla `computer` de la Base de datos de migasfree. Es único por cada ordenador. Se utiliza en lugar de referirse al `UUID` de la placa base de un ordenador ya que el `CID` es mucho más corto y por tanto más utíl a la hora de referirte a un equipo concreto. El `CID` aparece por defecto en la etiqueta del ordenador que muestra el comando `migasfree-label`. Por ejemplo, un atributo `CID-572` se correspondería con el `UUID` 5FD85780-9BC9-11E3-91B8-F0921CF3678D.

8.2 Tipos de Etiquetas

Hasta ahora has visto que una propiedad es un código que se ejecuta en el cliente para obtener un atributo automáticamente. Ahora bien, pueden existir casos en que no se puede obtener automáticamente estos atributos. Imagina que quieres "etiquetar" ciertos equipos según la funcionalidad que van a realizar (Tratamiento gráfico, administración, aula, etc). Esto no es algo que a priori se pueda programar.

En migasfree existe la posibilidad de crear estas etiquetas y asignarlas manualmente a los equipos tal y como harías con una etiqueta física que pegas a un ordenador.

Una etiqueta no ejecutará ningún código en el cliente. Es el propio registro del ordenador en el servidor de migasfree quien lleva asignada manualmente estas etiquetas. A todos los efectos una etiqueta es un atributo más del sistema y por tanto te permitirá hacer el despliegue tambien en función de ellas.

Por cada `Configuración-Tipos de etiqueta` existirá un conjunto de etiquetas que manualmente debes añadir en `Datos-Etiquetas`. Una vez añadidas, puedes asignarlas a `Datos-Ordenadores`. También puedes editar `Datos-Etiquetas` y asignarle un conjunto de ordenadores.

Existe en el cliente el comando `migasfree-tags` que permite consultar y asignar etiquetas desde el propio cliente.

Para consultar las etiquetas de un equipo ejecuta:

```
migasfree-tags --get
```

Para asignar etiquetas al equipo, seleccionando manualmente las etiquetas entre las disponibles en el sistema, ejecuta:

```
migasfree-tags --set
```

Para asignar determinadas etiquetas a un equipo escribe las etiquetas separadas por espacios:

```
migasfree-tags --set <ETIQUETA1> <ETIQUETA2> ...
```

Para asignar etiquetas en el servidor migasfree pero que no se produzca **ningún cambio de paquetes** utiliza:

```
migasfree-tags --communicate <ETIQUETA1> <ETIQUETA2> ...
```

Para quitar todas las etiquetas de un equipo ejecuta:

```
migasfree-tags --set ""
```

Las etiquetas están relacionadas con los campos de los repositorios:

- default preinclude packages
- default include packages
- default exclude packages

ya que al ejecutar el comando `migasfre-tags --set` se instalarán los paquetes definidos en `preinclude` e `include` y se desinstalarán los paquetes definidos en el campo `exclude`, siempre y cuando los atributos asignados al repositorio coincidan con los del equipo. Esto se utiliza para crear la imagen iso de los escritorios.

Note: En AZlinux usamos `migasfree-tags` básicamente para, partiendo de una imagen iso de Ubuntu 12.04, desintalar e instalar los paquetes que componen nuestro escritorio y crear una imagen del disco para clonar.

Note: En Vitalinux se emplean las etiquetas para cambiar fácilmente de "sabor". Cuando se quiere cambiar de sabor Vitalinux (Infantil, Primaria, Profes, ...), simplemente se eligen las etiquetas mediante el comando `migasfree-tag --set`, produciéndose automáticamente la instalación y desinstalación de los paquetes correspondientes. Tambien se utiliza en la creación del DVDs, permitiendo hacer una iso para cada sabor o conjunto de sabores.

8.2.1 Campos de Tipos de Etiqueta.

- **Prefijo**: Es una combinación de tres números o letras. Este prefijo se utiliza para agrupar e identificar las etiquetas.
- **Nombre**: Denomina el tipo de etiqueta.
- **Habilitado**: Si no está marcado, las etiquetas de este tipo no serán funcionales.
- **Clase**: El funcionamiento es exactamente igual al campo de mismo nombre que tienen las Propiedades.

  Un valor muy útil que puede tomar este campo es el de `agrega por la derecha`. Imagina que quieres agrupar los ordenadores por ubicación para liberar software por distintas zonas. Una forma de hacerlo es crear un `Tipo de Etiqueta` llamada p.e. `UBICACIÓN` definada de clase `agrega por la derecha`. Después puedes crear las `Etiquetas` de tipo `UBICACION` p.e.:

```
UBI-PLANTA-1.SEDE_CENTRAL.MADRID
```

Cuando un equipo con esta etiqueta asignada se conecta al servidor, automáticamente el servidor interpretará que tiene no una, sino tres etiquetas:

```
UBI-MADRID
UBI-SEDE_CENTRAL.MADRID
UBI-PLANTA-1.SEDE_CENTRAL.MADRID
```

Con lo que finalmente podemos liberar software a todo `MADRID`, a toda la sede central de Madrid, o sólamente a la planta 1ª.

Note: Observa que el caracter de delimitación es el punto: `.`

8.3 Conjuntos de Atributos

En ocasiones puedes necesitar agrupar `Atributos`.

Imagina que tienes muchos equipos a los que asignar una cierta `Etiqueta` y que te resulta pesado tener que hacerlo uno a uno. Puedes entonces crear un `Conjunto de Atributos`.

Supón que tienes subredes con un buen ancho de banda y otras subredes que no, y que necesitas liberar software en función de esto. Podríamos crear dos `Conjuntos de Atributos`:

```
Conjunto 1:
      Nombre:                 RED LENTA
      Atributos asignados:    NET-192.168.1.0/24
                              NET-192.168.8.0/24

Conjunto 2:
      Nombre:                 RED RAPIDA
      Atributos asignados:    SET-ALL SYSTEMS
      Atributos excluidos:    SET-RED LENTA
```

De esta manera, cualquier equipo de las subredes 192.168.1.0/24 ó 192.168.8.0/24 al ejecutar `migasfree -u` se le asignará automáticamente un `Atributo: SET-RED LENTA`. Al resto de equipos se le asignará el `Atributo: SET-RED RAPIDA`.

Ahora ya podríamos crear `Repositorios` y asignarles dichos `Atributos`.

Los `Conjuntos de Atributos` no ejecutan ningún código en el cliente, sino que son evaluados en el servidor. Si un ordenador pertenece a un conjunto se le asigna un `Atributo` con el mismo nombre que el `Conjunto de Atributos`.

8.3.1 Campos de Conjuntos de Atributos

- **Nombre**: Denomina al conjunto.
- **Activo**: Indica si el conjunto será evaluado.
- **Atributos**: Lista de `Atributos` que formarán parte el conjunto.
- **Excluidos**: Lista de `Atributos` a excluir de conjunto.

8.4 Versiones

Migasfree puede trabajar con distintos Sistemas Operativos. Una `version` en migasfree representa a un conjunto de ordenadores que comparten un mismo S.O.

Por ejemplo, en AZLinux tenemos actualmente 5 versiones establecidas:

- AZLinux-1 (SLED 10.2)
- AZLinux-2 (OpenSUSE 11.2)
- AZLinux-12 (Ubuntu 12.04)

- WIN-XP (Windows XP)
- ZA (Ubuntu 10.04 para escritorios tipo kioskos)

Cada ordenador estará configurado en una única versión en un momento dado. Cambios de versión en un ordenador crean en el sistema un registro de `migración` automáticamente. De esta manera es posible conocer las diferentes migraciones de S.O. que se han ido produciendo en los equipos y en qué momento se han hecho efectivas. Puedes consultar las migraciones accediendo a `Datos-Migraciones`.

Mediante el ajuste `MIGASFREE_AUTOREGISTER` se permite, o no, a los equipos registrar automáticamente las versiones. Puedes consultarlo en *Ajustes del servidor migasfree*.

8.4.1 Campos de la Versión.

- **Nombre**: Denomina a la versión.
- **Sistema de gestión de paquetes**: El P.M.S. que se utiliza en el S.O. de esta versión.
- **Actual line computer**: Es un equipo que sirve como referencia para comparar con el resto de equipos. Se debe elegir un equipo que represente la línea actual de la versión y que sea lo más "estandar" posible.
- **Actual line packages**: Lista ordenada de paquetes que componen la actual línea de la versión. Cuando se conecta al servidor el equipo asignado en el campo `Actual line computer` se actualiza automáticamente este campo.

  Este campo tiene relación con el campo `Inventario de software` de los ordenadores, ya que en este último sólo se mostrará la diferencia de paquetes repecto al `Actual line computer`. De esta manera se puede ver fácilmente que cambios se han producido respecto al ordenador asignado como de referencia.
- **Autoregistrado**: Si está marcado se permiten registrar ordenadores desde un cliente automáticamente. En este caso, sólo con que un equipo esté configurado con la versión será añadido automáticamente a la base datos.

  En caso contrario sólo se podrán registrar ordenadores mediante el uso de un usuario que cuente con los permisos adecuados para añadir ordenadores al sistema.
- **Plataforma**: a la que pertenece la versión.

8.5 Plataformas

Las versiones se clasifican por plataformas. Las plataformas vienen establecidas por la función python `platform.system()` y por tanto sus valores pueden ser:

- Linux
- Windows
- (Otras)

Esta clasificación de las versiones te permite realizar consultas y estadísticas en función de la plataforma.

Mediante el ajuste `MIGASFREE_AUTOREGISTER` se permite, o no, a los equipos registrar automáticamente las plataformas. Puedes consultarlo en *Ajustes del servidor migasfree*.

8.6 Usuarios Migasfree

En migasfree existen dos tipos de usuarios, los usuarios que administran migasfree y los usuarios que utilizan los ordenadores. Este apartado se refiere a los primeros.

Cuando se genera la base de datos de migasfree se crean 7 grupos de usuarios y 8 usuarios predeterminados:

8.6.1 Grupos de Usuarios

En función de las tareas que los usuarios de administración de migasfree pueden realizar, se establecen los siguientes grupos de usuarios.

- `Configurator` con permisos de lectura/escritura a:
  - Propiedades
  - Versiones
  - P.M.S.
  - Plataformas
  - Comprobaciones
  - Definicion de fallas
  - Mensajes
  - Actualizaciones
  - Mensajes del servidor
  - Migraciones
  - Notificaciones
- `Computer Checker` tiene permisos de lectura/escritura a:
  - Errores
  - Fallas
  - Mensajes
  - Actualizaciones
- `Liberator`. Permisos de lectura/escritura a:
  - Repositorios
  - Calendarios
- `Packager` cuenta con permisos de lectura/escritura a:
  - Paquetes
  - Almacenes
- `Query`. Permisos de lectura/escritura a:
  - Consultas
- `Device installer` cuenta con permisos de lectura/escritura a:
  - Dispositivos
- `Reader`. Permisos de sólo lectura a todas las tablas.

8.6.2 Usuarios

- admin. Tiene permisos de lectura/escritura a todas las tablas.
- packager. Pertenece a los grupos Reader y Packager.
- configurator. Pertenece a los grupos Reader y Configurator.
- installer. Pertenece a los grupos Reader y Device installer.
- query. Pertenece a los grupos Reader y Query.
- liberator. Pertenece a los grupos Reader y Liberator.
- checker. Pertenece a los grupos Reader y Computer Checker.
- reader. Pertenece al grupo Reader.

Estos usuarios tienen por defecto como contaseña su nombre, es decir, la contraseña de admin es admin, y lo mismo es aplicable al resto de usuarios.

Estos usuarios son ficticios para realizar pruebas y conviene que sean eliminados. Se recomienda crear los usuarios reales que usarán la web del servidor migasfree asignándoles los grupos de usuarios correspondientes.

Note: Es importante que en un entorno de producción se deshabiliten los usuarios que no se vayan a utilizar o que al menos se les cambie la contraseña por motivos de seguridad.

8.6.3 Cambio de contraseña

La contraseña puede ser cambiada por los usuarios pulsando en su nombre de usuario y que aparece arriba a la derecha en todas las páginas web del servidor.

También puede ser modificada por otro usuario que tenga marcado el campo Es superusuario, accediendo al registro del usuario en cuestión y modificando directamente su campo Contaseña.

8.6.4 Version por defecto de un Usuario

Los usuarios tienen un campo version que sirve para filtrar registros. De esta manera cuando un usuario consulta los Repositorios p.e., solo se muestran los repositorios de la versión que tiene asignada.

Un usuario puede seleccionar su versíon mediante el desplegable que aparece a la izquierda de *Alertas*'.

8.7 Comprobaciones

Son un conjunto de comprobaciones que se realizan para alertar al usuario. Pulsando en cada una de las Alertas puedes obtener más información. ver figura 8.1.

Cada Alerta viene programada en un registro de Comprobación. Hay 8 comprobaciones predeterminadas:

- Errors to check. Cuando en un cliente migasfree se produce algún error, éste es enviado al servidor. Esta comprobación hace que se muestren estos errores. Una vez revisado o solucionado un error en el cliente debes editar el error en el servidor y marcar el campo comprobado. Esto hará que ya no aparezca en la lista de errores a comprobar. Puedes también seleccionar un conjunto de errores en la lista de errores y en el desplegable de acción seleccionar La comprobación es correcta.

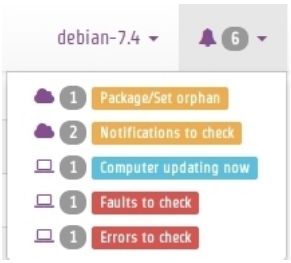


Fig. 8.1: Alertas del sistema.

- `Faults to check`. Cuando en un cliente migasfree se produce una falla ésta es enviada al servidor. Esta comprobación hace que se muestren las fallas pendientes. La manera de proceder con las fallas es similar a la de los `Errors to check`
- `Notifications to check`. Son hechos que se han producido en el sistema y que son informados mediante esta comprobación. Un ejemplo de notificación es cuando un equipo da de alta una plataforma o una versión nueva en el sistema.
- `Package/Set orphan`. Comprueba si hay paquetes que no están asignados a ningún repositorio.
- `Computer updating now`. Cuando un equipo está ejecuando el cliente migasfree, éste va informando al servidor de lo que está haciendo mediante un texto que indica el proceso que está realizando. Cuando el cliente migasfree finaliza, envía al servidor un mensaje de texto vacío. Esta comprobación comprueba cuantos de estos mensajes se han recibido.
- `Computer delayed`. Si pasa un determinado tiempo desde que se recibió el último mensaje del cliente, es muy posible que algo ha ido mal en el cliente. Quizás perdió la conexión, o el usuario apagó el equipo en medio de la ejecución del cliente migasfree, o quizás ha habido algún error. Esta comprobación permite detectar estos casos. La cantidad de tiempo viene establecida por defecto en 30 minutos y puede ser modificado mediante el ajuste `MIGASFREE_SECONDS_MESSAGE_ALERT` de los *Ajustes del servidor migasfree*.
- `Server Messages`. Es similar a `Computer updating now` pero para los mensajes que se producen en el servidor.
- `Server Messages Delayed`. Similar a `Computer delayed` pero para los mensajes que se producen en el servidor.
- **Nombre**: Denomina la comprobación
- **Descripción**: Sirve para describir en detalle la comprobación.
- **Código**: Instrucciones escritas en `Django` para realizar la comprobación. El servidor interpretará las siguientes variables que deben ser asignadas en este campo.

  `result`. Debe ser un numero. Un valor de 0 indica que no hay nada que mostrar en la alerta.

  `alert`. Es el tipo de alerta. Puede ser uno de estos tres valores: 'info', 'warning' ó 'danger'. Se representan con los colores azul, naranja o rojo. El valor por defecto es 'info'.

  `url`. Es el link al que accederá el usuario cuando pulse en la alerta.

  `msg`. Es el texto a mostrar en la alerta.

  `target`. Puede ser "computer" o "server" para indicar que la comprobación está relacionada con el equipo cliente o con el servidor. Se representa con el icono de un ordenador o con el de una nube.

  Mira éste codigo de ejemplo, el de `Errors to check`:

```
from migasfree.server.models import Error
result = Error.objects.filter(checked__exact=0).count()
```

```
        url = '/admin/server/error/?checked__exact=0'
        icon = 'error.png'
        msg = 'Errors to check'
        target = 'computer'
```

Lo primero que hacemos en importar el modelo Error. Depués obtenemos el número de registros de errores que que no se han comprobado y lo asignamos a la variable `result`. A continuación vamos asignando los valores a cada una de las variables.

- **Habilitado**. Activa o desactiva la comprobación.
- **Alerta**. Permite especificar si la comprobación es algo a lo que hay que prestar especial atención o no. Te pongo como ejemplo‘‘Computer updating now‘‘: Que un equipo esté ejecutando el cliente migasfree no es en realidad algo por lo que alarmarse, es más bien una comprobación de tipo informativo. En este caso no marcarás el campo `Alerta`. En cambio que no se reciban más mensajes pasados 30 minutos desde el último mensaje enviado por un cliente sí debes marcarlo como `Alerta` (`Computer delayed`)

Las `alertas` proporcionan al usuario una vista general de la situación actual del sistema, dirigiendo su actuación a lo relevante.

El objetivo en todo momento debería ser mantener el sistema con 0 alertas. Esto indicaría que se han revisado los errores, se han comprobado las fallas, no hay paquetes huérfanos, etc.

8.8 Fallas

Una falla es un hecho negativo que se produce en un equipo cliente. Por ejemplo que un equipo se quede con poco espacio en la partición de sistema, es algo a lo que se debe prestar atención y ser solucionado antes de que sea tarde.

Migasfree mediante las fallas permite lanzar código en el cliente con este objetivo. Fíjate que las posibilidades son inmensas y que te permite ser muy proactivo.

En definitiva, una falla es un código que se ejecuta en el cliente. Si el código escribe algo por la salida estandar ésta será enviada al servidor como `Falla`. El servidor entonces añadirá un registro de `Falla` apareciendo en las `Alertas` de los usuarios de migasfree.

- **Nombre**: Denomina a la falla.
- **Descripción**: Para detallar lo que hace la falla.
- **Habilitado**: Activa o desactiva la falla.
- **Lenguaje de programación**: Especifica en que lenguaje está escrito el `código`. Mi recomendación es que programes en la medida de lo posible en python.
- **Código**: Instrucciones que detectan alguna falla en los equipos y que debe poner en la salida estandar un texto que indique la falla producida. Puede serte útil en algunos casos poner también el procedimiento a seguir.
- **Atributtes**: Permite asignar a que equipos cliente será efectiva la falla. Por ejemplo si escribes el código en bash deberías asignar la falla sólo a los equipos con plataforma Linux `PLT-Linux`, ya que plataformas Windows no serán capaces de ejecutar bash. Tambien te puede interesar programar una falla sólo para obtener información de un equipo o de un grupo de equipos.
- **Users**: Sirve para asignar usuarios de migasfree a los que les aparecerán las fallas de este tipo cuando se accede desde las `Alertas`

  (Sólo se muestran las que están pendientes de comprobar por el usuario autenticado).

  Si una definición de falla no tiene asignado ningún usuario, las fallas que se produzcan aparecerán a cualquier usuario autenticado.

Note: Poder ejecutar código en los clientes proporciona una gran potencia para realizar cualquier cosa. Usa esta capacidad con responsabilidad y sé meticuloso en las comprobaciones antes de activar cualquier falla.

8.9 Consultas

Migasfree incorpora un sistema para crear consultas parametrizables sencillas.

Cada consulta se programa en un registro y podrá ser ejecutada accediendo a `Consultas`

Hay una pocas consultas ya predefinidas, pero puedes programar nuevas o adaptar las que ya existen.

8.9.1 Campos de consulta

- **Nombre**: Denomina la consulta.
- **Descripción**: Describe la consulta.
- **Código**: Instrucción en Django de la consulta. Mediante la asignación de una variables predeterminadas el servidor podrá crear la consulta.

  Las variables en concreto son:

  - **QuerySet**: Conjunto de registros de la consulta.
  - **fields**: Lista de los campos del QuerySet que se quieren mostrar.
  - **titles**: Lista de los titulos de los campos que se quieren mostrar.
  - **version**: Sirve para obtener la version del usuario y poder hacer filtros cuando se requiera.
- **Parámetros**: Permite la petición de parámetros de consulta. Se debe crear una función que se llame `form_params` y que devuelva una clase que herede de `ParametersForm`

En fin, creo que lo mejor es que veas un ejemplo para comprender la programación de consultas: hay una que muestra todas las consultas, se llama `QUERIES`:

Parametros: Aquí se programa un formulario de parametros que pedirá el paŕametro `id`.

```
def form_params():
    from migasfree.server.forms import ParametersForm
    class myForm(ParametersForm):
        id = forms.CharField()
    return myForm
```

Código: Programamos que si el parámetro `id` que ha introducido el usuario es una cadena vacía, la variable query sea igual a todos los regitros de la tabla `Consulta`. En caso de que el usuario introduzca un valor filtramos las `Consultas` por `parameters['id']`.

```
if parameters['id'] == '':
    query = Query.objects.all()
else:
    query = Query.objects.filter(id=parameters['id'])
fields = ('id', 'name', 'description', 'code', 'parameters')
```

Note: Para realizar consultas necesitarás conocer un poco los QuerySet de Django y la `Documentación del modelo de datos`. Está última la tienes disponible al final de todas las páginas del servidor pulsando sobre el icono de información .

8.10 Errores autocomprobables

Por defecto los errores producidos por el P.M.S. se añaden al sistema como no comprobados. Ahora bien, en ocasiones puede resultar tedioso tener que marcar como comprobados uno a uno ciertos errores que más que errores son "alertas".

Para automatizar esta tarea puedes crear un `error autocomprobable`. Simplemente añade un registro con el patrón de búsqueda deseado y los errores que coincidan son ese patrón se marcarán automáticamente como comprobados.

Por ejemplo si quisieras que todos los errores que llegan del tipo:

```
2014-10-03 10:44:47
Error: Generic error
Info: Curl error: Couldn't resolve host 'myserver'
```

se autocomprobaran, podrías emplear el siguiente patrón:

```
.*\sError: Generic error\sInfo: Curl error: Couldn't resolve host 'myserver'
```

CHAPTER 9

La Liberación

El conocimiento nos hace responsables.

—Ernesto Guevara.

Este es el capítulo que mejor define a migasfree, ya que su principal funcionalidad es la de ofrecer unos determinados repositorios de paquetes que estarán disponibles para los cliente en función de sus atributos.

En los proyectos de software libre la liberación tiene que ver con poner a disposición de la comunidad un determinado software. Aspectos como la autoría o la licencia son esenciales, tanto o más como el propio software que se libera.

Liberar software en migasfree implica además decidir a quién y a partir de en qué momento un cliente tendrá acceso a dicho software.

9.1 Subiendo Paquetes al servidor

Antes de poder liberar el software obviamente tienes que subirlo al servidor.

Como viste en los primeros capítulos la manera de hacerlo es utilizando el comando de cliente:

```
migasfree-upload -f <mipaquete>
```

o si quieres subir un conjunto de paquetes (Set) ponlos todos juntos en un directorio y ejecuta:

```
migasfree-upload -d <midirectorio>
```

Para subir paquetes al servidor es necesario utilizar un usuario que tenga permisos de lectura/escritura en la tabla de almacenes y paquetes. Por defecto el usuario `packager` y el usuario `admin` los tienen.

Para no tener que introducir cada vez que subas un paquete al servidor el usuario, su contaseña y/o la versión con la que trabajas, puedes asignar los ajustes indicados en la sección [Packager] de *Ajustes del cliente migasfree*.

9.2 Almacenes

Un almacén es un ubicación o ruta del servidor donde se colocan los paquetes y/o conjuntos de paquetes subidos al servidor. No es más que un directorio colgando de la ruta /var/migasfree/repo/<VERSION>/STORES, y que se utiliza para tener organizados los paquetes. Tambien es accesible desde un explorador web accediendo a la ruta:

```
http://tuservidor/repo/<VERSION>/STORES.
```

Lo anterioriormente expuesto corresponde al lugar donde se almacenan los archivos del paquete, pero además hay una parte lógica que es necesaria llevar en la base de datos de migasfree. Es lo que denominamos registros de "Almacén".

Cuando se utiliza el comando `migasfree-upload` y se indica una ubicación inexistente, el servidor automáticamente creará el registro lógico en la base de datos y creará la carpeta en el sistema de archivos.

- `Nombre`: Denomina al almacén. Corresponde al nombre de la carpeta en el sistema de archivos.
- `Versión`. Indica la versión migasfree a la que pertenece el almacén.

9.3 Paquetes

Cuando subes un paquete o un conjunto de paquetes al servidor además de copiarse en el almacén o ubicación indicada, se crea un registro lógico en la base de datos. Estos registros nos servirán para asignarlos posteriormente en los `Repositorios` que vayamos creando.

- `Nombre`: Es el nombre del fichero del paquete.
- `Versión`: Indica la versión migasfree a la que pertenece el paquete.
- `Almacén`: Especifica la ubicación donde está situado el paquete.

A la derecha del nombre del paquete, en la lista de paquetes, hay un desplegable con las siguientes acciones:

- `Información del paquete`. Permite ver los metadatos del paquete.
- `Descargar`. Permite almacenar el paquete seleccionado en tu equipo.

Si necesitas borrar uno o varios paquetes, selecciónalos y en el desplegable `Acción` elige `Eliminar Paquetes/conjuntos seleccionados` y después pulsa en el botón `ir`.

- `Eliminar Paquetes/conjuntos seleccionados`. Permite borrar el registro del Paquete. A medida que vayas haciendo cambios en el software irás teniendo distintas versiones del mismo paquete. Generalmente te interesará trabajar sólo con la última versión. Si quieres que sólo te aparezca ésta a la hora de asignarlo a los `Repositorios` puedes borrar los registros de `Paquetes` antiguos. Borrar el registro no borrará el archivo del paquete en ningún caso y simplificarás la selección de paquetes.

Un paquete huérfano es un paquete que no está asignado a ningún Repositorio. Cuando un paquete es subido al servidor, o cuando lo quitas de un repositorio y no está en ningún otro repositorio se convierte en un paquete huérfano. Existe una comprobación de `Alerta` que te avisará de cuales son estos paquetes.

9.4 Información de los paquetes

Si accedes a `Liberación-Información de paquetes` verás que te aparecen dos carpetas:

- `STORES`. Muestra ésta carpeta, en donde podrás navegar hasta un determinado paquete que hayas subido previamente.
- `REPOSITORIES` Muestra los Repositorios físicos (en el sistema de archivos) que se hayan creado, y que son los que en última instancia verán los clientes. En realidad los paquetes que veas en `REPOSITORIES` no son más que enlaces simbólicos a los paquetes ubicados en `STORES`.

Si quieres ver los metadatos de un determinado paquete simplemente haz click en él.

9.5 Repositorios

Me gusta la definición: **migasfree es simplemente un gestor de repositorios de paquetes**. En realidad es básicamente esto. De hecho así es como empezó este proyecto, y a partir de aquí ha ido creciendo hasta convertirse en lo que es hoy en día, un gestor de sistemas.

A todos los efectos, y desde el punto de vista del cliente, un repositorio en migasfree es un repositorio de paquetes estándar como los que puedas encontrar en cualquier Distribución. Migasfree permite crear muy fácilmente estos repositorios y asignarlos a los equipos en función de sus atributos a partir de una fecha determinada.

- **Nombre**: Denomina al repositorio.

  Note: En AZLinux solemos incorporar en el nombre del repositorio el número de tarea de redmine al hace referencia el cambio de software que queremos liberar.

- **Versión**: Especifica la versión en la que estará disponible el repositorio.
- **Habilitado**: Activa o desactiva el repositorio.
- **Comentario**: Campo de texto que sirve para registrar aclaraciones sobre el repositorio. Es muy conveniente que registres las modificaciones que vayas haciendo a los repositorios en este campo, indicando quién, cuándo y qué se ha modificado.

  Un ejemplo de como lo hacemos en AZLinux sería:

```
[alberto@2013-03-09] Añadido paquete azl-firefox-12.0-3_all.deb

[alberto@2013-04-10] Añadido paquete azl-firefox-12.0-4_all.deb

[eduardo@2013-05-10] Detectado problemas en algunos clientes. Desactivo
    el repositorio hasta diagnosticar y encontrar solución.
```

- **Fecha**: A partir de la cual estará disponible el repositorio en los clientes.
- **Calendario**: Especifica una programación del repositorio basada en calendario. En el siguiente apartado tienes más información.
- Packages
  - **Paquetes**: En este campo se seleccionan los `paquetes` y/o `conjuntos de paquetes` que se incluirán en el repositorio.

    Que un paquete esté incluido en un repositorio y el repositorio accesible desde el cliente, no implica que se instale el paquete. Los sistemas de paquetería sólo actualizan aquellos paquetes que ya estuvieran instalados en el sistema.

    Cada vez que hay una modificación de este campo y se pulsa el botón `Guardar` se generarán los metadatos del repositorio. Dependiendo de la cantidad de paquetes que se tengan que procesar, el tiempo para realizar este proceso puede ser largo. En los casos en los que se asigne un `conjunto de paquetes` donde se incluyan todos los paquetes de un DVD p.e. puede llegar a ser del orden de decenas de minutos.

    Note: Fíjate que aparecen sólo los `paquetes` (los subidos individualmente) más los `conjuntos de paquetes` a la hora de seleccionarlos en los repositorios. Los paquetes incluidos dentro de los `conjuntos de paquetes` no pueden asignarse individualmente. Esto es así para simplificar y hacer más sencilla la asignación de `paquetes` y no perdernos entre los miles que componen una Distribución.

  - **Paquetes a instalar**: Campo de texto que especifica una lista de paquetes separados por espacios o por retornos de carro. Estos paquetes serán instalados **obligatoriamente** a los clientes que tengan acceso al repositorio.

Se puede especificar sólo el nombre del paquete, o el nombre de paquete mas una versión.

Este campo se tiene en cuenta al ejecutar los comandos de cliente `migasfree --update` y `migasfree-tags --set`

- **Paquetes a desinstalar**: Campo de texto que especifica una lista de paquetes separados por espacios o por retornos de carro que serán desinstalados **obligatorimente** en los clientes.

  Este campo se tiene en cuenta al ejecutar los comandos de cliente `migasfree --update` y `migasfree-tags --set`

- Default.
  - **Default preinclude packages**: Campo de texto que especifica una lista de paquetes separados por espacios o por retornos de carro. Este campo sirve para instalar paquetes que configuran repositorios externos a migasfree. Un ejemplo de este tipo de paquetes lo tienes en el paquete vx-repo-openshot.

    La razón de la existencia de este campo es que después de instalar el repositorio externo es necesario obtener de nuevo los metadatos de los repositorios (apt-get update) a fin de que el cliente tenga acceso inmediatamente a los paquetes contenidos en el repositorio externo.

    Estos paquetes serán instalados a los clientes que tengan acceso al repositorio al ejecutar el comando `migasfree-tags --set`.
  - **Default include packages**: Campo de texto que especifica una lista de paquetes separados por espacios o por retornos de carro. Estos paquetes serán instalados a los clientes que tengan acceso al repositorio al ejecutar el comando `migasfree-tags --set`.
  - **Default exclude packages**: Campo de texto que especifica una lista de paquetes separados por espacios o por retornos de carro que serán desinstalados en los clientes que tengan acceso al repositorio al ejecutar el comando `migasfree-tags --set`.
- Atributtes.
  - **Atributos**: Aquellos clientes que tengan un atributo que coincida con los asignados en este campo tendrán accesible el repositorio (a menos que otro atributo lo excluya).
  - **Excludes**: Sirve para excluir Atributos de la lista de Atributos anterior.

    Por ejemplo si quieres liberar un paquete a toda la subred `192.168.92.0` menos al equipo `PC13098` puedes hacerlo asignando:

    * Atributos: `NET-192.168.92.0/24`
    * Excludes:`HST-PC13098`

9.6 Calendarios

Los calendarios te permiten programar sistemáticamente liberaciones en el tiempo para unos determinados atributos previamente establecidos partiendo de la fecha del Repositorio.

Por ejemplo, en AZLinux usamos distintos calendarios (LENTO, NORMAL, RAPIDO, MUY RAPIDO) según la críticidad del cambio de software que se va a liberar o de su urgencia. En estos calendarios asignamos días de demora para los distintos servicios de nuestro ayuntamiento.

```
    CALENDARIO LENTO
        a los 0 días:  GRP-EQUIPOS DE TEST.
        a los 5 días:  CTX-SERVICIO DE PERSONAL
        a los 10 días: CTX-GESTION TRIBUTARIA
        a los 15 días: SET-ALL SYSTEMS

    CALENDARIO MUY RAPIDO
```

```
        a los 0 días: CTX-SERVICIO DE PERSONAL, CTX-GESTION TRIBUTARIA
        a los 2 dias: SET-ALL SYSTEMS
```

Es conveniente que en la última demora asignes, si procede, el atributo `SET-ALL SYSTEMS`.

Cuando asignas un calendario a un repositorio podrás ver la temporalización resultante en la columna `línea temporal` de `Liberación-Repositorios` (Pulsa en el desplegable que contiene el nombre del calendario).

Asignar un calendario a un repositorio no es obligatorio.

Esta programación de la liberación se utiliza fundamentalmente para conseguir:

- No aplicar una liberación de golpe a muchos equipos, lo que puede provocar un consumo de tráfico de red intenso (imagina 1000 equipos actualizando libreoffice a la vez)
- Liberar poco a poco los paquetes y así poder hacer comprobaciones más tranquilamente. Cualquier error en el empaquetado o bug en los fuentes del paquete puede ser mas manejable si ha afectado a pocos equipos y no a la totalidad.

Un determinado cliente tendrá acceso al repositorio si:

- Tiene un atributo que coincide con alguno de los asignados en el repositorio y ya se ha cumplido la fecha del repositorio.
- O existe un atributo coincidente con el calendario cuya fecha de repositorio mas demora se ha cumplido.
- Siempre y cuando un atributo del cliente no coincida con el campo `Excludes` del repositorio.

Una manera en que puedes ver una estimación de la cantidad de equipos que un calendario va haciendo efectivos los repositorios a lo largo de los días es acceder a `Estadísticas-Ordenadores previstos/demora`.

- **Nombre**: Denomina al calendario.
- **Descripcion**: Describe el calendario.
- Demoras: Es un conjunto de días (demoras) a los que se asignan atributos.
  - **Demora**: Número de días desde la fecha del repositorio a los que los atributos asignados serán efectivos en el repositorio. No se tienen en cuenta ni sábados ni domingos.
  - **Atributos**: Lista de atributos para una demora.
  - **Duración**: Número de días en que se completará el despliegue a los equipos asignados a la demora. O dicho de otra forma, si asignamos el atributo `SET-ALL SYSTEMS` y una duración de 20 días, obtendríamos un incremento diario aproximado del 5% del total de equipos.

9.7 Repositorios internos vs externos

LLamamos repositorio interno al repositorio que controla el servidor migasfree.

Un repositorio externo es un repositorio configurado en los clientes y que no apunta al servidor migasfree, Los repositorios que vienen por defecto configurados en las Distribuiciones son un ejemplo. Otro serían los repositorios tipo `ppa`.

Si quieres tener un mayor control de tus sistemas, mi recomendación es que te bajes todos los paquetes de los repositorios de tu distribución a una fecha y luego los subas como `conjunto de paquetes` al servidor y crees un repositorio al efecto. A esto le denominamos `congelar un repositorio`.

De esta manera tendrás congelados a una fecha los repositorios de tu Distribución, y podrás actualizar sólo el software que te interese. Si te decides por este método obviamente tendrás que empaquetar un código que deshabilite los repositorios externos en los clientes.

Repositorios Internos	Repositorios Externos
Requieren mantenimiento ante las actualizaciones de los paquetes	No requieren mantenimiento ya que es mantenido por el dueño del repositorio
Mayor control de los sistemas frente a los cambios, siendo tu quién decide qué actualizaciones deben producirse	Menor control frente a los cambios
Si el servidor migasfree está en la red local, no produce tráfico internet	Genera tráfico internet

Un pequeño script para obtener los paquetes de los repositorios externos (en este caso para ubuntu-12.04) podría ser:

```
#!/bin/bash

function download(){
  _SERIE_POCKET=$1
  download_repo "$_SERIE_POCKET" "main"
  download_repo "$_SERIE_POCKET" "multiverse"
  download_repo "$_SERIE_POCKET" "restricted"
  download_repo "$_SERIE_POCKET" "universe"
}

function download_repo(){
  _SERVER=http://en.archive.ubuntu.com/ubuntu
  _PKGS=Packages
  _SERIES=$1
  _REPO=$2
  _PATH=`pwd`
  echo "PATH= $_PATH"
  wget $_SERVER/dists/$_SERIES/$_REPO/binary-i386/$_PKGS.bz2
  bzip2 -d $_PKGS.bz2
  _FILES=`grep "^Filename:" $_PKGS| awk '{print $2}'|sort`
  _TARGET=$_SERIES-$_REPO
  echo "$_FILES" > Packages-$_TARGET
  mkdir -p $_TARGET
  cd "$_TARGET"
  for _f in $_FILES
  do
    _file=${_f:6+${#_REPO}}
    _BASE=`basename $_file`
    mkdir -p `dirname $_file`
    echo "Downloading $_SERIES $_f"
    wget -c -t1  $_SERVER/$_f -O $_file
  done
  cd "$_PATH"
  rm $_PKGS
}

download "precise-security"
download "precise-updates"
download "precise-backports"
download "precise"
```

9.8 El proceso de la liberación

Las tareas que debe realizar un liberador son:

- Controlar que no haya paquetes huérfanos, borrando los paquetes antiguos y creando los repositorios adecuados para los nuevos paquetes.
- Decidir que calendario es conveniente aplicar a cada repositorio.
- Decidir cuando un repositorio ha terminado de liberarse (se ha cumplido toda la línea temporal) que debe hacerse con sus paquetes.

En AZLinux mayoritariamente, y para no tener muchos repositorios activos, estos paquetes los asignamos a otro repositorio (ya existente para éste fin) que tiene asignado sólo el atributo `SET-ALL SYSTEMS`. Los repositorios que nos han servido para liberar poco a poco los paquetes son desactivados (no los borramos) para mantener así la historia de lo que se ha ido haciendo.

CHAPTER 10

La Actualización de los sistemas

> Al fin y al cabo, somos lo que hacemos para cambiar lo que somos.
>
> —Eduardo Galeano

En el capitulo anterior has aprendido a liberar paquetes desde un servidor migasfre. Pero para que se produzca el cambio de software no basta sólo con liberarlo. Los clientes deben poder acceder a los repositorios, bajarse los paquetes e instalarlos.

En este capítulo vas a centrarte en el cliente migasfree para ir conociendo los comandos que tienes a tu disposición.

10.1 El proceso de actualización

Ahora creo que es buen momento de aprender que hace `migasfree --update` :

- Envía mensaje de inicio del proceso de actualización al servidor.
- Envía errores de anteriores ejecuciones. Si los hay, el servidor creará un registro de `Error`.
- Recibe las Propiedades definidas en el servidor.
- Ejecuta dichas Propiedades y los resultados son enviados como `Atributos`. El servidor crea entonces un registro de `Login` donde se almacenarán estos `Atributos`.
- Recibe el código de las `Fallas` y los `Repositorios` a configurar basándose en los `Atributos` y fecha actual. Además, la lista de paquetes a desinstalar e instalar obligatoriamente tambien se reciben en este momento del proceso.
- Ejecuta y envía el resultado de las `Fallas`. Si estas se producen el servidor creará un registro de `Falla` por cada una de ellas.
- Configura los `Repositorios` que el servidor ha dispuesto en función de los `Atributos` del cliente y de la fecha actual.
- Actualiza los metadatos de los repositorios configurados en el sistema. Consiste simplemente en obtener el índice de paquetes actualizado de cada repositorio.
- Desinstala los paquetes obligatorios. Conjunto de paquetes definidos en el campo `Paquetes a desinstalar` de los `Repositorios` efectivos.
- Instala los paquetes obligatorios. Conjunto de paquetes definidos en el campo `Paquetes a instalar` de los `Repositorios` efectivos.
- Actualiza paquetes disponibles. En caso de que en los *Ajustes del cliente migasfree* `Auto_Update_Packages` sea `False` no se producirá esta actualización.

- Envía al servidor el historial de cambios en el software. Es la diferencia de paquetes instalados en el sistema antes y despues de desisntalar, instalar y actualizar los paquetes.
- Envía el inventario de software si es el equipo de referencia. Ver en *La configuración del sistema migasfree* los campos de las *Versiones*: `Actual line computer` y `Actual line packages`
- Envía el inventario de hardware periodicamente según `MIGASFREE_HW_PERIOD` de los *Ajustes del servidor migasfree*
- Envía los errores de ejecución. Si los hay el servidor creará un registro de `Error`.
- Por último envía un mensaje de proceso finalizado. Cuando el servidor recibe este mensaje añade un registro de `Actualización` en la base de datos que se emplean para hacer diferentes estadísticas.

10.2 El comando `migasfree`

La opción del comando migasfree `--update` es sin lugar a dudas la más importante. Su sintaxis es:

```
migasfree -u
migasfree --update
```

migasfree -u puede usarse conjuntamente con opción `--force-upgrade` para forzar la actualización de paquetes a pesar que en el ajuste `Auto_Update_Packages` esté asignado a `False`. Consulta el ajuste `Auto_Update_Packages` de los *Ajustes del cliente migasfree*

```
migasfree -u -a
migasfree --update --force-upgrade
```

Existen otras opciones que pueden hacer más fácil el mantenimiento a los administradores.

En las organizaciones que usan distintos S.O. con sistemas de paquetería diferentes, tanto para buscar, instalar ó desinstalar paquetes los administradores tienen que utilizar los comandos propios del sistema de paquetería. Utilizar las opciones del comando migasfree para realizar estas tareas te permite abstraerte del P.M.S. (No tendrás que estar pensando si estás en un sistema basado en Debian o en un Redhat p.e.):

- Para buscar un determinado paquete en los repositorios utiliza:

```
migasfree -s <texto>
migasfree --search <texto>
```

- Para instalar un determinado paquete usa:

```
migasfree -ip <paquete>
migasfree --install --package=<paquete>
```

- Para desinstalar un determinado paquete usa:

```
migasfree -rp <paquete>
migasfree --remove --package=<paquete>
```

Por último tienes la opción que permite registrar el equipo cliente en el servidor migasfree en caso de que en el registro `Version` del servidor el campo `Autoregistrado` esté desmarcado.

```
migasfree -g
migasfree --register
```

10.3 El comando `migasfree-tags`

Puedes ver una explicación de este comando y de su sintaxis en el campo `Etiqueta` de las *Propiedades* en el capítulo *La configuración del sistema migasfree*.

10.4 El comando `migasfree-label`

Consulta el ajuste `MIGASFREE_HELP_DESK` de los *Ajustes del servidor migasfree* donde se describe este comando.

CHAPTER 11

La Auditoría

> Nuestro conocimiento es necesariamente finito, mientras que nuestra ignorancia es necesariamente infinita.
>
> —Karl Raimund Popper

Una vez que liberas los paquetes y que los equipos se van actualizando llega el momento de que veas, de manera centralizada, toda la información que se ha ido generando a consecuencia del proceso de actualización.

Ésto es lo que te vas a encontrar en este capítulo.

11.1 Alertas

Muestra en lo que debe actuar el administrador para tener un sistema lo más íntegro posible. Ya lo viste en apartado *Comprobaciones* del capítulo *La configuración del sistema migasfree*.

11.2 Ordenadores

Accediendo a `Datos-Ordenadores` verás la lista de ordenadores que se han registrado en el servidor.

Puedes acceder al `Hardware` de un equipo desde la última columna de la lista de ordenadores.

Tambien puedes acceder a los datos que están relacionados con un equipo determinado midiante el desplegable que hay a la derecha del identificador del equipo. Así, fácilmente podrías ver los errores que ha habido en un equipo, su fallas, migraciones, actualizaciones, etc.

11.2.1 Campos de Ordenador

- **Nombre**: Es el nombre del equipo o el especificado en el ajuste `Computer_Name` de los *Ajustes del cliente migasfree*
- **Uuid**: Es el identificador único universal de la placa base del equipo.
- **Versión**: La versíon migasfree del ordenador.
- **Fecha de alta**: Fecha de alta del ordenador en migasfree.
- **IP**: La dirección ip del equipo en el momento de la actualización.

- **Inventario de software**: Diferencia actual entre el conjunto de paquetes del ordenador de referencia y el ordenador en cuestión. Ver en *Versiones* los campos de version: `Actual line computer` y `Actual line packages`
- **Historial de software**: Registro de los paquetes instalados y desinstalados según se van produciendo en el tiempo.
- **Última actualización**: Fecha en la que se finalizó por última vez la actualización del cliente migasfree.
- **Actualización hardware**: Fecha de la última actualización hardware.
- **Etiquetas**: Lista de Etiquetas asignadas actualmente al ordenador. Para una explicación del funcionamiento de las etiquetas mira los *Tipos de Etiquetas*.

11.3 Usuarios

A medida que el cliente de migasfree va ejecutándose en los equipos el servidor va añadiendo los usuarios que se han autenticado en el entorno gráfico.

Puedes ver la lista de usuarios en `Datos-Usuarios`

11.3.1 Campos de Usuario

- **Nombre**: Nombre de la cuenta de usuario para acceder al equipo.
- **Nombre Completo**: Nombre y apellidos del usuario.

11.4 Logins

Cuando se ejecuta `migasfree --update` se crea un registro de `Login` en el servidor.

Note: Migasfree sólo lleva por cada equipo el **último login**.

11.4.1 Campos de Login

- **Fecha**: Fecha y hora de la ejecución de `migasfree --update` en el equipo
- **Usuario**: Usuario en el entorno gráfico cuando se ejecutó el cliente migasfree.
- **Ordenador**: El equipo al que hace referencia el login.
- **Atributos** :Lista de `Atributos` que se han obtenido como resultado de ejecutar la `Propiedades` en el ordenador cliente en el proceso de actualización.

11.5 Errores

Conforme se vayan produciendo errores en los clientes irán llegando al servidor y serán mostrados en `Alertas`.

11.5.1 Campos de error

- **Ordenador**: Equipo en el que se ha producido el error.
- **Fecha**: Fecha y hora en que se produjo el error.
- **Error**: Mensaje que describe el error. Generalmente corresponde a la salida de error del front-end del P.M.S.
- **Comprobado**: Campo que se marcará manualmente cuando se ha comprobado y solucionado el error.
- **Versión**: Es la versión que tenía el equipo cuando se produjo el error.

11.6 Fallas

Ya viste el concepto de *Fallas* y como se pueden programar en el capítulo *La configuración del sistema migasfree*, así que no me repitiré.

Lo mismo que ocurre con los errores, conforme las fallas se vayan detectando en los clientes irán apareciendo en el `Alertas`.

11.6.1 Campos de falla

- **Ordenador**: Equipo en el que se ha producido.
- **Definición de falla:**: Tipo de Falla. Hace referencia al código que ha generado la falla.
- **Fecha**: Fecha y hora en que se produjo la falla.
- **Texto**: Mensaje que describe la falla. Corresponde a la salida standard del codigo de la `Definición de la falla`.
- **Comprobado**: Campo que se marcará manualmente cuando se ha comprobado y solucionado la falla.
- **Versión**: Es la versión que tenía el equipo cuando se produjo la falla.

11.7 Atributos

A medida que se vayan actualizando los equipos, el servidor migasfree irá añadiendo los atributos enviados por los clientes con objeto de que puedas liberar paquetes en función de estos atributos.

11.7.1 Campos de Atributo

- **Propiedad de atributo**: Propiedad a la que hace referencia el atributo.
- **Valor**: Identifica el atributo.
- **Descripción**: Describe el atributo.

Una explicación del funcionamiento de los atributos la puedes obtener en el apartado *Propiedades* de *La configuración del sistema migasfree*.

11.8 Etiquetas

Manualmente podrás añadir etiquetas y asignarlas a ordenadores para liberar software en función de éstas.

El funcionamiento de las etiquetas ya lo hemos visto en los *Tipos de Etiquetas*.

11.8.1 Campos de Etiqueta

- **Propiedad**: Hace referencia al tipo de etiqueta.
- **Valor**: Identifica a la etiqueta.
- **Descripción**: Describe la etiqueta.
- **Ordenadores**: Permite asignar ordenadores a la etiqueta.

11.9 Migraciones

Como hemos visto al principio de este capítulo los `Ordenadores` se identifican inequívocamente por el UUID de la placa base y además mantienen un campo `Versión` que se corresponde con el ajuste del mismo nombre de los *Ajustes del cliente migasfree*. Ahora bien, en el momento en que el servidor detecta que no corresponde la versión que tiene el ordenador en la base de datos del servidor con la que recibe del equipo, el servidor actualiza el registro `Ordenador` y además añade un registro de `Migración`. De esta manera se consigue llevar un histórico de migraciones.

11.9.1 Campos de Migración

- **Ordenador**: Equipo que se ha migrado de versión migasfree.
- **Versión**: Version migasfree.
- **Fecha**: Fecha y hora en que se ha detectado el cambio de versión

11.10 Notificaciones

Ante hechos relevantes en el sistema, el servidor genera notificaciones para alertar a los administradores.

11.10.1 Campos de Notificación

- **Fecha**: Fecha y hora en que se ha generado la notificación.
- **Notificación**: Describe el hecho.
- **Comprobado**: Campo que se marcará manualmente cuando se ha recibido la notificación.

11.11 Consultas

Aquí podrás ejecutar las `Consultas` disponibles.

Puedes añadir nuevas consultas o modificar las predeterminadas accediendo a `Configuración-Consultas`. Una pequeña explicación de cómo se programan la puedes encontrar en el apartado *Consultas* de *La configuración del sistema migasfree*.

11.12 Estadísticas

Es una lista con estadísticas predefinidas.

- **ordenadores actualizados/hora**: Gráfica de barras que indica la cantidad (única) de equipos que han completado la actualización de migasfree por hora.
- **ordenadores actualizados/día**: Gráfica de barras que indica la cantidad (única) de equipos que han completado la actualización de migasfree por día.
- **ordenadores actualizados/mes**: Gráfica de barras que indica la cantidad (única) de equipos que han completado la actualización de migasfree por mes.
- **ordenadores previstos/demora**: Gráfica de líneas que representa una previsión, basada en los `Atributos` del último `Login` de cada ordenador, de los equipos que accederían a un repositorio hipotético según `Calendarios`.
- **ordenadores/versión**: Gráfica de tarta donde se aprecia la cantidad de ordenadores por version.

11.13 El proceso de las comprobaciones

Al igual que como liberador debes realizar un conjunto de tareas para mantener el sistema en codiciones, continuamente te llegarán errores, fallas, etc. que debes comprobar y atender. Esta es la misión para un usuario `checker`.

¿Qué tareas tienes que hacer como comprobador del sistema?. Sencillo. Mantén las `Alertas` a 0. Él sistema te irá avisando que debes atender.

- Comprueba periódicamente la existencia de `Errores`. Soluciónalos y márcalos como comprobados.
- Comprueba periódicamente la existencia de `Fallas`. Soluciónalas y márcalas como comprobadas.
- Comprueba periódicamente la existencia de `Notificaciones`. Una vez leídas márcalas como comprobadas.

Part IV

Puesta en producción

CHAPTER 12

Migasfree en producción

El valor del producto se halla en la producción.

—Albert Einstein

Si te has decidido a instalar en producción el servidor migasfree debes cambiar las contraseñas a los usuarios que vienen por defecto, y preparar un backup de la base de datos y de la carpeta /var/migasfree.

12.1 Obtención de los paquetes de migasfree

Si en http://migasfree.org/repo/dists no están los paquetes de migasfree para la Distribución que vas a emplear puedes generarlos tú mismo.

En *Empaquetando migasfree* tienes instrucciones de cómo obtenerlos.

12.2 Configuración del servidor

Crea el fichero `/etc/migasfree-server/settings.py` con el siguiente contenido (no te olvides de sustituir la password por la del usuario migasfree de Postgresql):

```
MIGASFREE_ORGANIZATION="My Organization"
DATABASES = {
        "default": {
            "ENGINE": "django.db.backends.postgresql_psycopg2",
            "NAME": "migasfree",
            "USER": "migasfree",
            "PASSWORD": "mipassword",
            "HOST": "",
            "PORT": "",
        }
    }
```

Este es el fichero de configuración del servidor migasfree. Hay diversas variables que se pueden configurar aquí para modificar el comportamiento de migasfree.

Si necesitas cambiar la password del usuario migasfree en postgresql haz esto:

```
# su postgres
$ psql -c "ALTER USER migasfree WITH PASSWORD 'mipassword';"
$ exit
```

Note: Para una personalización más avanzada mira los *Ajustes del servidor migasfree*.

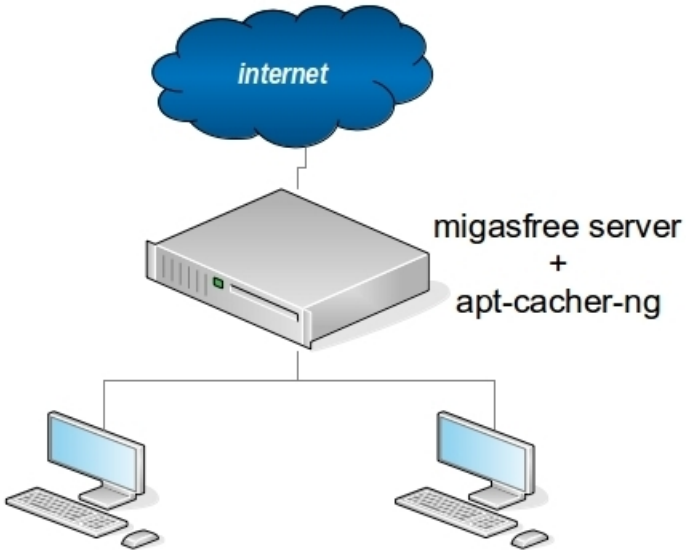
12.2.1 Cambiando las contraseñas

- Accede a `Configuracion-Usuarios` y verás los perfiles de usuarios.
- Edita el usuario `admin`. Cámbiale la contraseña y guárdalo.
- Edíta el resto de perfiles de usuario y deshabilítalos o cámbiales la contraseña.

La explicación de los usuarios y sus grupos lo has visto ya en *La configuración del sistema migasfree*

12.3 Servicio de caché de paquetes

Montar un caché de paquetes para disminuir el tráfico de internet es habitual. Su funcionamiento es muy sencillo. Cuando un equipo necesita descargar un paquete de internet lo solicita al caché. Si el servicio de caché no lo tiene ya almacenado lo descargará de internet, lo almacenará y se lo ofrecerá al equipo. Si otro equipo necesita ese mismo paquete, como ya está en el caché ya no se producirá tráfico internet sino que el servicio de caché lo ofrecerá directamente al equipo.



Puedes instalar el servicio de caché de paquetes en el equipo donde has instalado el servidor migasfree, o en otro servidor.

Por ejemplo puedes instalar `apt-cacher-ng`.

```
# apt-get install apt-cacher-ng
```

Configura el usuario para la administración del servicio.

```
# nano /etc/apt-cacher-ng/security.conf
```

Descomenta la línea que empieza por AdminAuth y modifica el usuario y la contraseña:

```
AdminAuth: <usuario>:<contraseña>
```

Reinicia el servicio.

```
#service apt-cacher-ng restart
```

Por defecto el puerto del servicio apt-cacher-ng es el 3142. Accede a la página http:<miservidor>:3142 para la administración del servicio de caché.

Hasta aquí hemos instalado y configurado el caché en el servidor.

Para la configuración de los clientes, debes crear el fichero `/etc/apt/apt.conf.d/02proxy` con el siguiente contenido:

```
Acquire::http { Proxy "http://<miservidor>:3142"; };
```

Para hacerlo correctamente modifica el paquete acme-migasfree-client añadiéndo este fichero al paquete.

Otra manera de configurar los clientes es haciendo uso del ajuste `Package_Proxy_Cache` de los *Ajustes del cliente migasfree*. La diferencia entre éste método y el anterior es que el primero hará uso del servicio del caché de paquetes tanto cuando ejecutes el comando migasfree en los clientes, como cuando ejecutes el gestor de paquetes (apt-get). En cambio en el segundo método sólo usará el servicio de caché al ejecutar el comando migasfree.

Puede consultar el manual de apt-cacher-ng para una configuración más avanzada del servicio de caché.

12.4 Backups

A continuación te sugiero un manera de hacer los backups.

12.4.1 Dump de la base de datos

Para hacer el dump de la base de datos, crea el fichero `/var/migasfree/dump/migasfree-dump.sh` (deberás modificar "mipassword" por la del usuario migasfree en posgresql):

```
#!/bin/bash
export PGPASSWORD=mipassword
pg_dump migasfree -U migasfree > /var/migasfree/dump/migasfree.sql
```

Crea tambien el fichero `/var/migasfree/dump/migasfree-restore.sh` para el caso que tengas que restaurar un dump de la Base:

```
#!/bin/bash

if [ ! "$UID" = "0" ] ; then
  echo "debes ejecutar como root"
fi

/etc/init.d/apache2 stop

echo "borrando BD..."
echo "DROP DATABASE migasfree;" | su postgres -c psql -

echo "creando BD migasfree..."
su postgres -c "createdb -W -E utf8 -O migasfree migasfree" -

echo "restore dump..."
su postgres -c "psql -U migasfree -f /var/migasfree/dump/migasfree.sql" -

/etc/init.d/apache2 start
```

Finalmente ponemos permisos de ejecución a los scripts:

```
chmod 700 /var/migasfree/dump/migasfree-dump.sh
chmod 700 /var/migasfree/dump/migasfree-restore.sh
```

12.4.2 Tarea periódica

Para programar una tarea que se ejecute periódicamente realizando el dump de la base de datos y la copia de los ficheros de los repositorios, crea el fichero `/var/migasfree/dump/migasfree-backup.sh` con el siguiente contenido:

```
# DUMP de la BD postgresql de migasfree
/var/migasfree/dump/migasfree-dump.sh

# BACKUP FICHEROS
# (aqui se debe programar el backup de /var/migasfree con rsync p.e.)
```

Cámbiale los permisos:

```
chmod 700 /var/migasfree/dump/migasfree-backup.sh
```

Edita como root crontab:

```
crontab -e
```

y programa la tarea para que se ejecute diariamente a las 23:30 p.e. añadiendo la siguiente línea a crontab:

```
30 23 * * * /var/migasfree/dump/migasfree-backup.sh
```

12.5 Etiquetando los clientes

Para facilitar la atención a los usuarios cuando tengan un problema, es conveniente imprimir y pegar físicamente la etiqueta que identifica inequívocamente a cada equipo ejecutando desde el cliente el comando:

```
migasfree-label
```

Consulta el ajuste `MIGASFREE_HELP_DESK` de los *Ajustes del servidor migasfree*

Note: Tambien puedes imprimir la etiqueta desde otro equipo si conoces su UUID accediendo desde un explorador web a la siguiente dirección:

http://<miservidormigasfree>/computer_label/?uuid=<UUID_DEL_ORDENADOR>

CHAPTER 13

Creando tu propia Distro

> No tratéis de guiar al que pretende elegir por sí su propio camino.
>
> —William Shakespeare

Una Distro no es más que un conjunto de software seleccionado y preparado para instalarse fácilmente.

Existen herramientas que te permiten personalizar una Distribución Linux fácilmente sin grandes complicaciones, y también puedes crear tu Linux desde cero, eso sí armándote de paciencia.

Pero como te decía, para crear tu Distribución personalizada debes:

- Seleccionar el software que incluirás en ella.
- Preparar un sistema sencillo para instalar todo ese software.

En este capítulo te describo diferentes opciones que he utilizado para realizar estas dos tareas.

La idea principal al trabajar con migasfree es que todo debe ser empaquetado, incluida la personalización del software. Así que debes elegir que software incluirá tu Distribución y crear los paquetes que la personalicen.

El método que te expongo a continuación es muy versatil, y es el de utilizar migasfree para especificar esta selección de paquetes.

Se trata de usar los campos del `Repositorio`:

- `default preinclude packages`. Lista de paquetes que configuran repositorios externos.
- `default include packages`: Lista de paquetes a instalar.
- `default exclude packages`: Lista de paquetes a desinstalar.

y asignar al `Repositorio` el `Atributo` `SET-ALL SYSTEMS`

No tienes porque indicar todos los paquetes. Como vamos a partir de una Distribución generalista como Debian, Ubuntu, RedHat, etc. indicamos sólo los paquetes que queremos añadir o eliminar a la Distro.

También puedes hacer una selección de los paquetes que compondrían unos "sabores", y en vez de usar el `Atributo` `SET-ALL SYSTEMS`, crear una `Etiqueta` por sabor y asignarla en diferentes `Repositorios`

La creación de etiquetas la viste en *La configuración del sistema migasfree*

Ahora que has elegido y creado uno o varios `Repositorios` en migasfree con los paquetes que debe llevar tu Distribución, es el momento de ver varios métodos para instalar tu Distribución personalizada y controlada desde un servidor migasfree.

13.1 El método de “andar por casa”

Recomendado si no quieres complicarte la vida y tienes pocos equipos en los que instalar tu Distro:

1. Instala la Distribución generalista en el equipo.
2. Instala y configura el cliente migasfree. Si es preciso registra el ordenador mediante `migasfree --register`.
3. Ejecuta `migasfree-tags --set`.

13.2 Generando un Live/CD

Es el método recomendado si la instalación la puede realizar cualquier persona. Se trata de hacer básicamente lo mismo que en el método anterior pero sustituyendo el primer paso por el empleo de un Live/CD en un entorno “chroot”.

1. Prepara un entorno chroot con el Live/CD de partida.
2. Instala y configura el cliente migasfree en el entorno chroot. Si es preciso registra el ordenador mediante `migasfree --register`.
3. Ejecuta `migasfree-tags --set` dentro del entorno.
4. Finalmente genera una imagen iso del entorno

Puedes ver un ejemplo de cómo se hace en vitalinux con vx-create-iso.

13.3 Clonación de imagen

Es el método que usamos en AZLinux y está recomendado si tienes muchos equipos y eres tú quien hace las instalaciones.

1. Instala la Distribución generalista en un equipo que hará de “master”.
2. Instala y configura el cliente migasfree. Si es preciso registra el ordenador mediante `migasfree --register`.
3. Ejecuta `migasfree-tags --set`.
4. Prepara el equipo para clonar y crea una imagen para su clonación.
5. Clona la imagen en los equipos.

13.3.1 Reinstalando la selección de paquetes

Una vez instalada tu Distro, en cualquier momento podrás cambiar de sabor ejecutando otra vez `migasfree-tags --set`. Observa que has podido decidir cambiar la selección de paquetes en migasfree entre tanto, esto te permite ir probando tu Distro fácilmente mientras aún la estás definiendo.

CHAPTER 14

FAQ

> Uno reconoce a las personas inteligentes por sus respuestas. A los sabios se los reconoce por sus preguntas.
>
> —Naguib Mahfuz

14.1 Cuando accedo al servidor web me aparece: `Server error (500)`

Este error puede estar motivado por múltiples causas. La más probable es que la contraseña del usuario `migasfree` en `Posgresql` no sea la misma que la que está configurada en el servidor.

Comprueba la contraseña que tienes en `/etc/migasfree-server/settings.py` es la misma que la del usuario `migasfree` en Postgresql. Si no existe este fichero la contraseña por defecto es `migasfree`.

Si necesitas cambiarla haz esto:

```
# su postgres
# psql
# ALTER USER migasfree WITH PASSWORD 'mipassword';
```

14.2 ¿Cómo hago una propiedad para obtener el contexto LDAP de un usuario?

Necesitas que los clientes tengan instalado el paquete python-ldap. En el servidor tendrás que crear una nueva propiedad:

Prefijo: `CTX`

Nombre: `CONTEXTO LDAP`

Lenguaje: `python`

Clase: `Agrega por la derecha`

Código:

```
import sys
import ldap
import migasfree_client.utils

LDAP_SERVER = 'ldap.miservidor.es'
```

```
    LDAP_BASE = ''
    LDAP_SCOPE = ldap.SCOPE_SUBTREE

    def get_ldap_property(filter_str, property_str, base = LDAP_BASE, scope = LDAP_SCOPE):
        global global_ldap_object

        try:
            #print filter_str # DEBUG
            #print base # DEBUG
            _result = global_ldap_object.search_s(base, scope, filter_str, [property_str])
        except ldap.LDAPError, e:
            print e
            sys.exit(errno.ENOMSG) # no result

        #print 'DEBUG:', _result # DEBUG
        if _result == None or not _result:
            print 'No result in LDAP search'
            sys.exit(errno.ENOMSG) # no result

        if property_str == 'dn': # special case: dn is getted in other field
            return _result[0][0]

        try:
            _ret = _result[0][1][property_str]
            if len(_ret) == 1: # only one result?
                return _ret[0]
        except KeyError:
            return '' # empty value

        return _ret

    def get_dn(user):
        # cn=oXXXXx,ou=XXXX,o=XXXXXX
        return get_ldap_property('(cn=%s)' % user, 'dn')

    def get_context(user):
        result = get_dn(user).split(',')

        ret = ''
        for item in result[:]:
            tmp = item.split('=')
            if tmp[0] == 'ou' or tmp[0] == 'o':
                ret = '%s%s.' % (ret, tmp[1])

        return ret[:-1] # remove trailing '.'

    def run():
        global global_ldap_object
        global_ldap_object = ldap.initialize('ldap://%s:389' % LDAP_SERVER)

        user=migasfree_client.utils.get_current_user().split("~")[0]
        print get_context(user)

    if __name__ == '__main__':
        run()
```

14.3 ¿Cómo hago una propiedad para obtener los grupos LDAP de un usuario?

Necesitas que los clientes tengan instalado el paquete python-ldap. En el servidor tendrás que crear una nueva propiedad:

Prefijo: `GRP`

Nombre: `GRUPOS LDAP`

Lenguaje: `python`

Clase: `Lista`

Código:

```
import sys
import ldap
import migasfree_client.utils
LDAP_SERVER = 'ldap.miservidor.es'
LDAP_BASE = ''
LDAP_SCOPE = ldap.SCOPE_SUBTREE

def get_ldap_property(filter_str, property_str, base = LDAP_BASE, scope = LDAP_SCOPE):
    global global_ldap_object

    try:
        #print filter_str # DEBUG
        #print base # DEBUG
        _result = global_ldap_object.search_s(base, scope, filter_str, [property_str])
    except ldap.LDAPError, e:
        print e
        sys.exit(errno.ENOMSG) # no result

    #print 'DEBUG:', _result # DEBUG
    if _result == None or not _result:
        print 'No result in LDAP search'
        sys.exit(errno.ENOMSG) # no result

    if property_str == 'dn': # special case: dn is getted in other field
        return _result[0][0]

    try:
        _ret = _result[0][1][property_str]
        if len(_ret) == 1: # only one result?
            return _ret[0]
    except KeyError:
        return '' # empty value

    return _ret

def get_groups(user):
    # TODO only groups of organization or all of them?
    _result = get_ldap_property('(cn=%s)' % user, 'groupMembership')
    if not _result:
        return '' # no groups found

    # only one result?
    if type(_result) is str:
        _result = [_result]

    _ret = ''
    for _item in _result:
        _t = _item.split(',')
        if '=' in _t[0]:
            _ret = '%s%s, ' % (_ret, _t[0].split('=')[1])

    return _ret[:-2] # remove trailing ',

def run():
    global global_ldap_object
    global_ldap_object = ldap.initialize('ldap://%s:389' % LDAP_SERVER)

    user=migasfree_client.utils.get_current_user().split("~")[0]
    print get_groups(user),
```

```
if __name__ == '__main__':
    run()
```

14.3.1 Sobre el cliente migasfree

14.4 El cliente migasfree devuelve el mensaje: “firma no válida”

Las claves almacenadas en el cliente no coinciden con la versión indicada en `/etc/migasfree.conf`

Borra las claves del equipo cliente.

Para la versión de migasfree-client 4.6 ó inferior usa:

```
# rm /root/.migasfree-keys/*
```

Para la versión de migasfree-client 4.7 ó superior usa:

```
# rm -rf /var/migasfree-client/keys/[server]/*
```

Note: Si es necesario vuelve a registrar el cliente ejecutando: migasfree –register.

14.5 Imposible obtener /PKGS/binary-amd64/Packages 404 Not Found

Por defecto los repositorios en el servidor se generan para la arquitectura i386.

Accede a `Configuracion - p.m.s. - apt-get` y modifica el campo `crear repositorio` de esta manera:

```
cd %PATH%
mkdir -p %REPONAME%/PKGS/binary-i386/
mkdir -p %REPONAME%/PKGS/binary-amd64/
mkdir -p %REPONAME%/PKGS/sources/
cd ..
dpkg-scanpackages -m dists/%REPONAME%/PKGS /dev/null | gzip -9c > dists/%REPONAME%/PKGS/binary-i386/Packages
dpkg-scanpackages -m dists/%REPONAME%/PKGS /dev/null | gzip -9c > dists/%REPONAME%/PKGS/binary-amd64/Package
dpkg-scansources dists/%REPONAME%/PKGS /dev/null | gzip -9c > dists/%REPONAME%/PKGS/sources/Sources.gz
```

CHAPTER 15

Resolución de problemas

> Un problema deja de serlo si no tiene solución.
>
> —Eduardo Mendoza

A menudo puede ocurrir que migasfree no esté funcionando como se espera. Para obtener mas información y averiguar que te puede estar ocurriendo puedes poner tanto al cliente como al servidor en modo `DEBUG`.

Simplemente debes poner el ajuste `Debug` a True en los *Ajustes del cliente migasfree*

Cuando ejecutes `migasfree --update` en este modo desde una consola, verás en la salida estándar más información de la habitual.

Tambien puede serte útil consultar la información que se va generando en /var/tmp/migasfree.log

Al estar el servidor realizado con Django, puedes usar el ajuste DEBUG a `True`. Este ajuste del servidor debes ponerlo en el fichero /etc/migasfree-server/settings.py y despues reiniciar el servidor web.

Al hacer esto, la página del servidor migasfree que te está fallando te mostrará, en vez de una página de error escueta, otra página de error con información muy extensa y que te permitirá ver que está ocurriendo.

Note: Nunca dejes el modo DEBUG en un entorno de producción por medida de seguridad.

El servidor utiliza el módulo de logging de Python.

Para activar el sistema de logging en el servidor de migasfree añade el ajuste LOGGING en /etc/migasfree-server/settings.py:

```
LOGGING = {
    'version': 1,
    'formatters': {
        'verbose': {
            'format': '%(levelname)s %(asctime)s %(module)s %(process)d
%(thread)d %(message)s'
        },
        'simple': {
            'format': '%(levelname)s %(message)s'
        },
    },
    'handlers': {
        'file': {
            'level': 'DEBUG',
            'class': 'logging.FileHandler',
            'filename': '/tmp/migasfree.log',
            'formatter': 'verbose',
        },
        'console': {
            'level': 'DEBUG',
            'class': 'logging.StreamHandler',
```

```
                'formatter': 'verbose',
            },
        },
        'loggers': {
            'migasfree': {
                'handlers': ['console', 'file'],
                'level': 'DEBUG',
            }
        }
    }
```

Esto hará que en el fichero indicado (/tmp/migasfree.log) se almacenen los logs.

Part V

Ajustes

CHAPTER 16

Ajustes del servidor migasfree

No hay inteligencia allí donde no hay cambio ni necesidad de cambio.

—Herbert George Wells

Los ajustes de configuración del servidor migasfree se asignan en el fichero `/etc/migasfree-server/settings.py`.

Note: Este es un fichero `python`, por lo que hay que llevar cuidado con la sintaxis y la indentación.

16.1 Ajustes propios de migasfree

16.1.1 MIGASFREE_ORGANIZATION

Valor por defecto: 'My Organization'

Establece el nombre de tu organización.

Ejemplo:

```
MIGASFREE_ORGANIZATION = "ACME Corporation"
```

16.1.2 MIGASFREE_AUTOREGISTER

Valor por defecto: True

Especifica si los ordenadores pueden autoregistrar la plataforma y la version migasfree.

Si no quieres que ningún ordenador registre versiones y/o plataformas automáticamente, tienes que darlas de alta manualmente y asignar este ajuste a False.

Ejemplo:

```
MIGASFREE_AUTOREGISTER = False
```

16.1.3 MIGASFREE_COMPUTER_SEARCH_FIELDS

Valor por defecto: ('id', 'name',)

Establece los campos del modelo `Computer` por los que se podrá buscar un ordenador. El primer campo es importante ya que será el que aparezca en la primera columna de la lista de ordenadores. Si quieres ver el nombre del ordenador en vez del id en la lista de ordenadores asigna el campo `name` el primero de la lista.

Ejemplo:

```
MIGASFREE_COMPUTER_SEARCH_FIELDS = ("name", "ip") # Búsquedas por nombre e ip
```

16.1.4 MIGASFREE_TMP_DIR

Valor por defecto: '/tmp'

Asigna la ruta donde se alamacenarán los ficheros temporales generados por el servidor.

Ejemplo:

```
MIGASFREE_TMP_DIR = "/tmp/server"
```

16.1.5 MIGASFREE_REPO_DIR

Valor por defecto: '/var/migasfree/repo'

Directorio dónde se guardarán los paquetes y repositorios de cada una de las versiones

Ejemplo:

```
MIGASFREE_REPO_DIR = "/var/repositories"
```

16.1.6 MIGASFREE_SECONDS_MESSAGE_ALERT

Valor por defecto: 1800

Si un ordenador tarda mas de los segundos especificados en este ajuste en enviar un mensaje mientras se está actualizando, se considera que el ordenador va retrasado (Delayed). Normalmente esto ocurre cuando se ha perdido la conexión con el servidor por cualquier circunstancia, por ejemplo cuando el usuario ha apagado el equipo antes de que el cliente termine el proceso de actualización. De esta forma se queda registrado en el servidor como `Delayed`.

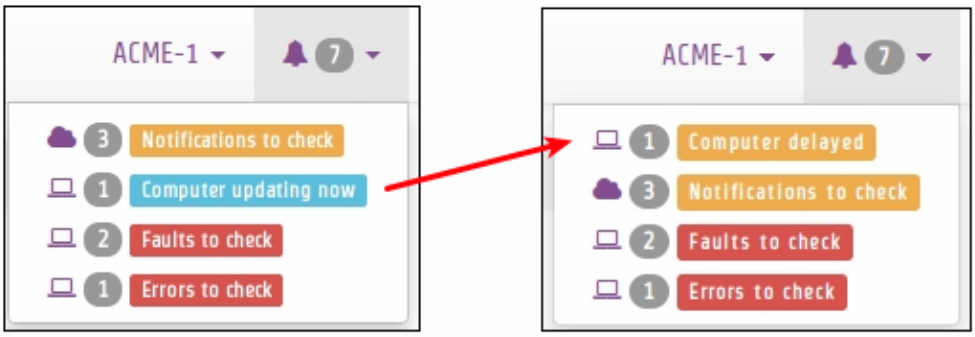


Fig. 16.1: Un equipo pasando a retrasado.

Ejemplo:

```
MIGASFREE_SECONDS_MESSAGE_ALERT = 3600 # Una hora
```

16.1.7 MIGASFREE_HELP_DESK

Valor por defecto: 'Put here how you want to be found'

Texto que apacere al ejecutar el comando del cliente `migasfree-label` para indicar al usuario como ponerse en contacto con Asistencia Técnica.

El comando `migasfree-label` tiene la finalidad de identificar inequívocamente al cliente. Este comando ejecutado en un cliente con entorno gráfico abrirá el navegador web mostrando una pequeña etiqueta que debe ser impresa y pegada en el ordenador con objeto de facilitar la asistencia técnica aún estando el ordenador apagado.

Fig. 16.2: Comando migasfree-label.

Ejemplo:

```
MIGASFREE_HELP_DESK = "Teléfono Asistencia Técnica: 555.12.34.56"
```

16.1.8 MIGASFREE_REMOTE_ADMIN_LINK

Valor por defecto: ''

Cuando se asigna un valor a este ajuste, apaceren nuevas acciones por cada ordenador. El objetivo es poder ejecutar algún código desde nuestro equipo hacia el equipo que se quiere administrar. Generalmente se usa para acceder por vnc ó ssh a los ordenadores, pero puede ser utilizado con cualquier otro fin.

Las variables que se pueden usar dentro de este ajuste son:

`{{computer.<FIELD>}}` para cualquier campo del modelo `Computer`

`{{<<PROPERTYPREFIX>>}}` cualquier propiedad del equipo cliente

Ejemplo vía ssh:

```
MIGASFREE_REMOTE_ADMIN_LINK = "ssh://root@{{computer.ip}}"
```

Ejemplo vía https y puerto (este último definido como propiedad `PRT`):

```
MIGASFREE_REMOTE_ADMIN_LINK = "https://myserver/?computer={{computer.name}}&port={{PRT}}"
```

Pueden usarse varios protocolos separados por un espacio en blanco:

```
MIGASFREE_REMOTE_ADMIN_LINK = "vnc://{{computer.ip}} checkping://{{computer.ip}} ssh://root@{{computer.ip}}"
```

Evidentemente el navegador con el que se accede a la web del servidor debe saber como interpretar dichos protocolos. Por ejemplo, si usas Firefox y quieres permitir el protocolo vnc debes acceder a la dirección `about:config` y añadir:

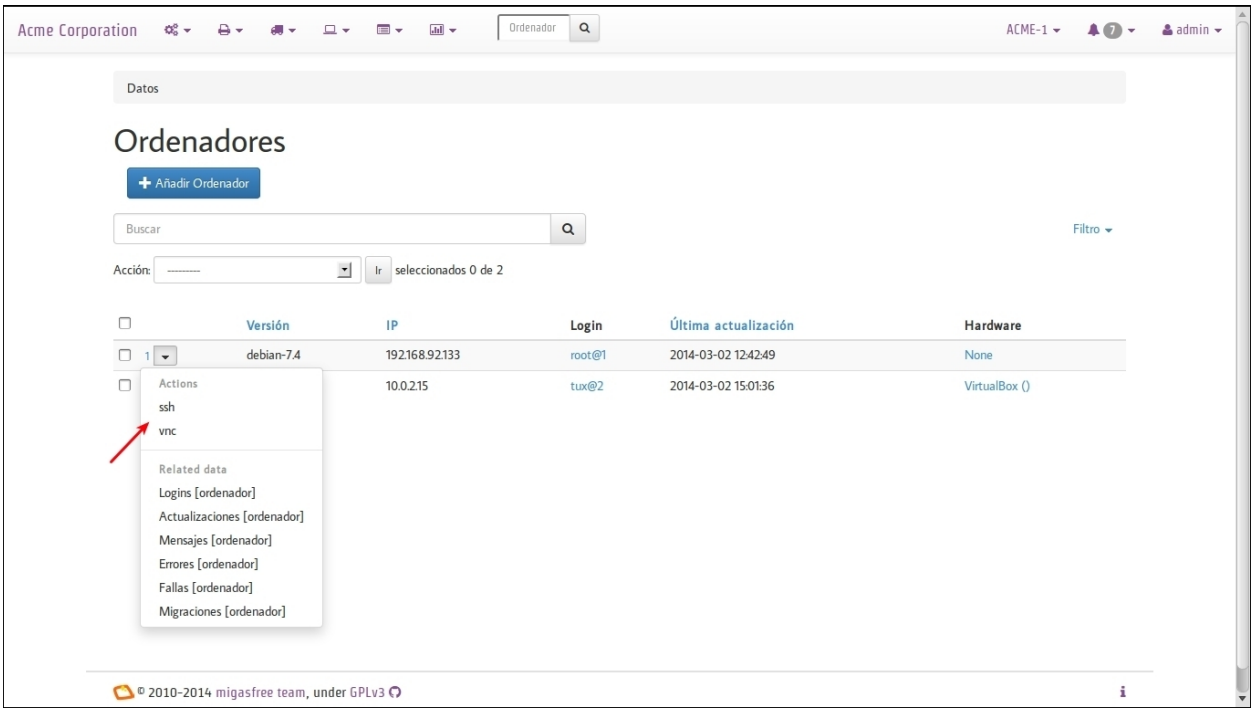


Fig. 16.3: MIGASFREE_REMOTE_ADMIN_LINK

```
    network.protocol-handler.expose.vnc false
```

Luego crea un fichero ejecutable para asociarlo al protocolo vnc para que lanze `vinagre` contra la ip del ordenador:

```
    #!/bin/bash
    URL=${1#vnc://}
    vinagre $URL
```

16.1.9 MIGASFREE_HW_PERIOD

Valor por defecto: 30

Periodo en días para el envío del hardware de los ordenadores al servidor. Si han pasado más días de los especificados se envia de nuevo toda la información del harware al servidor.

Ejemplo:

```
    MIGASFREE_HW_PERIOD = 1 # Cada día
```

16.1.10 MIGASFREE_INVALID_UUID

Valor por defecto =

```
    ["03000200-0400-0500-0006-000700080008", # ASROCK
       "00000000-0000-0000-0000-000000000000",
       "FFFFFFFF-FFFF-FFFF-FFFF-FFFFFFFFFFFF",
       "00000000-0000-0000-0000-FFFFFFFFFFFF"
       ]
```

Es una lista con UUIDs inválidos.

Algunas placas base pueden no tener asignado un UUID único, pudiéndose darse el caso que varios equipos tengan un mismo UUID. Esto provoca que en el servidor se comparta el mismo registro de `ordenador`.

Para evitarlo, es preciso añadir estos UUIDs en este ajuste.

Cuando un ordenador tiene un UUID inválido, el servidor toma y asigna el `nombre del ordenador` como UUID.

Puedes hacer UUIDs inválidos añadiendo en `/etc/migasfree-server/settings.py` la siguiente instrucción:

```
MIGASFREE_INVALID_UUID.extend( my_invalid_UUIDs )
```

donde `my_invalid_UUIDs` es una lista de UUIDs invalidos.

Ejemplo:

```
MIGASFREE_INVALID_UUID.extend( ["00000000-FFFF-FFFF-FFFF-FFFFFFFFFFFF",] )
```

16.1.11 MIGASFREE_NOTIFY_NEW_COMPUTER

Valor por defecto = False

Si se asigna a `True` el sistema añadirá una `Notificación` cuando un cliente migasfree se registra en el servidor por primera vez.

16.1.12 MIGASFREE_NOTIFY_CHANGE_UUID

Valor por defecto = False

Si se establece a True se creará una `Notificación` cuando un equipo cambia de UUID.

Esto puede ocurrir en contadas ocasiones y está relacionado con antiguos clientes de migasfree, UUIDs inválidos, o con cambios de placa base en el ordenador.

16.1.13 MIGASFREE_NOTIFY_CHANGE_NAME

Valor por defecto = False

Si se establece a True se creará una `Notificación` cuando se detecta que un ordenador ha cambiado de nombre.

Este ajuste puede resultar útil para detectar UUIDs no únicos.

16.1.14 MIGASFREE_NOTIFY_CHANGE_IP

Valor por defecto = False

Si se establece a True se creará una `Notificación` cuando un ordenador cambia de ip.

Este ajuste puede resultar útil para detectar UUIDs no únicos.

Note: No actives este ajuste si tienes ordenadores con IP dinámica, ya que se crearán demasiadas notificaciones irrelevantes.

16.2 Ajustes de Django

Los ajustes de Django tambien pueden ser modificados para adaptar el funcionamiento del servidor añadiendo el ajuste en el fichero `/etc/migasfree-server/settings.py`.

El más importante de este tipo de ajustes es:

16.2.1 DATABASES

Valor por defecto:

```
    DATABASES = {
        'default': {
            'ENGINE': 'django.db.backends.postgresql_psycopg2',
            'NAME': 'migasfree',
            'USER': 'migasfree',
            'PASSWORD': 'migasfree',
            'HOST': '',
            'PORT': '',
        }
    }
```

CHAPTER 17

Ajustes del cliente migasfree

> Uno no debe adaptarse al cambio, sino crearlo.
>
> —Jorge González Moore

Los ajustes de configuración del cliente migasfree se encuentran en el fichero `/etc/migasfree.conf`

17.1 Sección [client]

17.1.1 Server

Valor por defecto: localhost

Nombre del Servidor migasfree contra el que se van a realizar las actualizaciones.

Ejemplo:

```
    Server = 192.168.1.10
```

17.1.2 Version

Valor por defecto: Se basa en la función de python `platform.linux_distribution()`

Nombre de la versión migasfree. Sería el equivalente al nombre que le quieras dar a tu Distribución personalizada. Es muy *recomendable* que configures este ajuste, ya que para algunas Distribuciones la función `platform.linux_distribution()` puede producir versiones diferentes al aumentar de release (CentOS sería un ejemplo de esto).

Ejemplo:

```
    Version = MiDistro-1
```

17.1.3 Computer_Name

Valor por defecto: Se obtiene de la función de python `platform.node()`

Nombre del ordenador que se mostrará en migasfree. Si por calquier circunstancia se necesita que el nombre del ordenador no sea el `hostname` puedes configurar este ajuste para modificarlo.

Ejemplo:

```
Computer_Name = PC15403
```

17.1.4 Debug

Valor por defecto: False

Si se establece a True, la salida de los comandos del cliente mostrará información útil para la depuración.

Ejemplo:

```
Debug = True
```

17.1.5 GUI_Verbose

Valor por defecto: True

Indica si aparecen más o menos mensajes en el Intefaz Gráfico de Usuario. Si se asigna a False, sólo se mostrarán el primer y último mensaje.

Ejemplo:

```
GUI_Verbose = False
```

17.1.6 Auto_Update_Packages

Valor por defecto: True

Determina si al ejecutar `migasfree --update` se instalan las nuevas versiones de los paquetes ya instalados.

Si se establece este ajuste a **False** las actualizaciones de paquetes no se producirán al ejecutar `migasfree` con objeto de que sea el usuario quien decida cuándo quiere realizarlas (siguiendo p.e. la configuración de un gestor de actualizaciones tipo `update-manager` de Gnome figura 17.1 o ejecutando una actualización desde el front-end del sistema de paquetería).

Este ajuste no afectará en ningún caso a los paquetes a instalar y/o a los paquetes a desinstalar que hubiera definidos en los repositorios de migasfree.

Ejemplo:

```
Auto_Update_Packages = False
```

17.1.7 SSL_Cert

Valor por defecto: No establecido.

Ruta al fichero de certificado SSL de servidor en el cliente.

Si este fichero de certificado no existe se utilizará igualmente https para la privacidad, pero la autenticación entre extremos no estará garantizada. En este caso aparece en consola el siguiente mensaje:

```
Certificate does not exist and authentication is not guaranteed
```

Ejemplo:

```
SSL_Cert = "/path/to/ssl/cert"
```

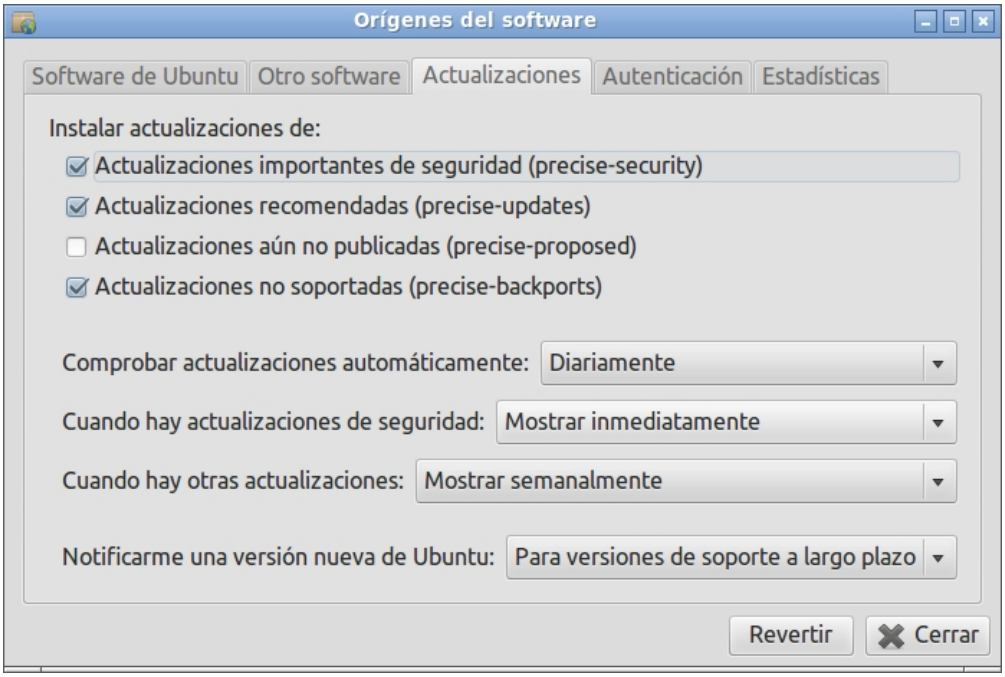


Fig. 17.1: Configuración del Gestor de Actualizaciones.

17.1.8 Proxy

Valor por defecto: No establecido.

Configuración del proxy.

Ejemplo:

```
Proxy = 192.168.1.100:8080
```

17.1.9 Package_Proxy_Cache

Valor por defecto: No establecido.

Permite especificar la dirección de un sistema cache de repositorios como podría ser `apt-cacher`.

Ejemplo:

```
Package_Proxy_Cache = 192.168.1.101:1234
```

Consulta el apartado `Servicio de caché de paquetes` del capítulo *Migasfree en producción*.

17.2 Sección [packager]

Esta sección se utiliza cuando se suben paquetes al servidor mediante el comando `migasfree-upload`. Se te pedirá la información que no hayas especificado en estos ajustes.

17.2.1 User

Valor por defecto: No establecido.

Usuario con permisos para subir paquetes al servidor migasfree. Por defecto la base de dato del servidor `migasfree` incluye el usuario `packager` con los permisos adecuados para almacenar paquetes en el servidor.

Ejemplo:

```
    User = packager
```

17.2.2 Password

Valor por defecto: No establecido.

Contraseña del usuario.

Ejemplo:

```
    Password = packager
```

17.2.3 Version

Valor por defecto: No establecido.

Indica el nombre de la versión migasfree a la que se van a subir los paquetes.

Ejemplo:

```
    Version = AZLinux-12
```

17.2.4 Store

Valor por defecto: No establecido.

Almacén en el servidor migasfree donde se guardarán los paquetes. Corresponde al nombre de una carpeta en el servidor donde se situará el Paquete o Conjunto de Paquetes. Puedes ver la lista de Almacenes disponibles accediendo a `Liberación - Almacenes` en la web del servidor migasfree. Si asignas un Almacén que no existe se creará automáticamente al subir el primer paquete.

Ejemplo:

```
    Store = Acme # Sitúa en /var/migasfree/repo/<Version>/STORES/Acme los paquetes.
```

17.3 Variables de entorno

Mediante el uso de variables de entorno podemos modificar tambien la configuración del cliente migasfree.

`MIGASFREE_CONF`

Por defecto el fichero de configuración del cliente migasfree se encuentra en `/etc/migasfree.conf` pero mediante la variable de entorno `MIGASFREE_CONF` podemos indicar al cliente que use otro fichero. Esto puede serte útil si tienes que subir paquetes mediante el comando‘‘migasfree-upload‘‘ a distintos servidores migasfree desde la consola.

Ejemplo:

```
export MIGASFREE_CONF='/etc/migasfree.conf.serverA'
migasfree-upload -f <mipaquete>
```

Además, todos los ajustes del fichero de configuración del cliente migasfree también pueden ser asignados mediante variables de entorno, siendo éstas variables prioritarias frente a los ajustes del fichero de configuración:

MIGASFREE_CLIENT_SERVER

MIGASFREE_CLIENT_VERSION

MIGASFREE_CLIENT_COMPUTER_NAME

MIGASFREE_CLIENT_DEBUG

MIGASFREE_CLIENT_GUI_VERBOSE

MIGASFREE_CLIENT_AUTO_UPDATE_PACKAGES

MIGASFREE_PROXY

MIGASFREE_CLIENT_PACKAGE_PROXY_CACHE

MIGASFREE_PACKAGER_USER

MIGASFREE_PACKAGER_PASSWORD

MIGASFREE_PACKAGER_VERSION

MIGASFREE_PACKAGER_STORE

Como ejemplo de uso de las variables de entorno imagina un escenario en el cual tienes un servidor migasfree y muchos centros en los que en cada uno de ellos hay un servicio de caché de paquetes para minimizar el tráfico de internet. Para configurar cada equipo deberías tener un paquete de configuración del cliente migasfree por cada centro, pero si tienes muchos centros esto puede resultar costoso. Una solución podría ser tener un sólo paquete de configuración del cliente migasfree para todos los centros y en la postinstalación del paquete crear las variables de entorno necesarias en función de la etiqueta del centro.

```
# Codigo de ejemplo postinst acme-migasfree-client

TAGS=`migasfree-tags -g`
for CENTRO in $TAGS
do
  if [ $CENTRO = "CTR-DELEGACION-BARCELONA" ]; then
    echo "MIGASFREE_CLIENT_PACKAGE_PROXY_CACHE='192.168.96.6:3142'" > /etc/profile.d/migasfree.sh
  fi
  if [ $CENTRO = "CTR-DELEGACION-MADRID" ]; then
    echo ""MIGASFREE_CLIENT_PACKAGE_PROXY_CACHE='192.168.80.4:3142'" > /etc/profile.d/migasfree.sh
  fi
done
```

Part VI

Empaquetado

CHAPTER 18

Empaquetando migasfree

> Nadie es como otro. Ni mejor ni peor. Es otro. Y si dos están de acuerdo, es por un malentendido
>
> —Jean-Paul Sartre

El proceso consiste básicamente en bajarte el fuente del proyecto y ejecutar el comando `bin/create-package`

Note: Si quieres la versión de desarrollo puedes bajar `master.zip` en vez de `latest.zip`

18.1 Creación del paquete migasfree-server (.deb)

Abre una terminal como root y baja el código fuente de migasfree:

```
wget https://github.com/migasfree/migasfree/archive/latest.zip
```

Necesitaremos tener instalado unzip para descomprimir el fichero zip:

```
apt-get install unzip
```

Descomprimimos el fichero latest.zip:

```
unzip latest.zip
rm latest.zip
```

Ahora tendrás una carpeta llamada migasfree-latest.

Creamos a continuacion el paquete migasfree-server. Para ello necesitamos tener instalado el paquete python-stdeb:

```
apt-get install python-stdeb
```

Nos situamos en la carpeta bin del proyecto y ejecutamos el script create-package:

```
cd migasfree-latest/bin
./create-package
cd ../..
```

Ahora en la carpeta deb_dist tenemos el paquete deb que instalamos:

```
dpkg -i migasfree-latest/deb_dist/migasfree-server_*_all.deb
```

Por problemas de dependencias seguramente se dejará sin configurar el servidor de migasfree. Para instalar las dependencias que faltan haremos:

```
apt-get -f install
```

18.2 Creación del paquete migasfree-client (.deb)

Abre una terminal como root y baja el código fuente del cliente migasfree:

```
wget https://github.com/migasfree/migasfree-client/archive/latest.zip
```

Necesitaremos tener instalado unzip para descomprimir el fichero zip:

```
apt-get install unzip
```

Descomprimimos el fichero latest.zip:

```
unzip latest.zip
rm latest.zip
```

Ahora tendrás una carpeta llamada migasfree-client-latest.

Creamos a continuación el paquete migasfree-client. Para ello necesitamos tener instalado el paquete python-stdeb:

```
apt-get install python-stdeb
```

Nos situamos en la carpeta bin del proyecto y ejecutamos el script create-package:

```
cd migasfree-client-latest/bin
./create-package
cd ../..
```

Ahora en la carpeta deb_dist tenemos el paquete deb que instalamos:

```
dpkg -i migasfree-client-latest/deb_dist/migasfree-client_*_all.deb
```

Por problemas de dependencias seguramente se dejará sin configurar el cliente de migasfree. Para instalar las dependencias que faltan haremos:

```
apt-get -f install
```

18.3 Otras Distribuciones a las implementadas

Si al ejecutar `./create-package` te aparece:

> **Error:** Computer distro is not available. Aborting package creation.

entonces, consigue el nombre de tu Distribución:

```
_DISTRO=$(python -c "import platform; print platform.linux_distribution()[0].strip()")
```

y crea un nuevo fichero cuyo nombre sea `$_DISTRO` en el directorio `setup.cfg.d` para paquetería rpm:

```
touch ../setup.cfg.d/$_DISTRO #
```

ó en el directorio `stdeb.cfg.d` para paquetería deb:

```
touch ../stdeb.cfg.d/$_DISTRO
```

Finalmente copia dentro de este fichero el contenido de otro fichero de una Distribución similar y modifica las dependencias necesarias.

Una vez realizado este proceso vuelve a ejecutar `./create-package`

CHAPTER 19

Empaquetando proyectos python

La diferencia entre el pasado, el presente y el futuro es sólo una ilusión persistente.

—Albert Einstein

Es muy posible que en la distribución en la que has instalado el servidor migasfree no encuentres los paquetes que se requieren para instalar el servidor, bien porque simplemente no están disponibles o bien porque la versión disponible no es suficientemente reciente como para ejecutar el servidor.

En este capítulo vas a empaquetar `django` como ejemplo y el método que usarás es válido para la mayoría de proyectos escritos en python.

19.1 Creación del paquete `django` en distros basadas en paquetería `apt`

Descarga el código del proyecto django cuya versión necesites y descomprímelo:

```
wget https://www.djangoproject.com/download/1.6.2/tarball/ -O Django-1.6.2.tar.gz
tar xzvf Django-1.6.2.tar.gz
cd Django-1.6.2
```

Asegúrate que tienes instalado el paquete python-stdeb:

```
apt-get install python-stdeb
```

Ahora crea el paquete:

```
python setup.py --command-packages=stdeb.command bdist_deb
```

En la carpeta `deb_dist` tendrás el paquete deb.

Más información en https://wiki.debian.org/Python/Packaging

19.2 Creación del paquete `django` en distros basadas en paquetería `rpm`

Descarga el código del proyecto django cuya versión necesites y descomprímelo:

```
wget https://www.djangoproject.com/download/1.6.2/tarball/ -O Django-1.6.2.tar.gz
tar xzvf Django-1.6.2.tar.gz
cd Django-1.6.2
```

Ahora crea el paquete:

```
    python setup.py bdist_rpm
```

En la carpeta `dist` tendrás el paquete rpm.

Para más información puedes consultar http://docs.python.org/2.0/dist/creating-rpms.html

Part VII

Anexos

CHAPTER 20

Bibliografía

Software Configuration Management, Bersoff, Henderson & Siegel, Prentice-Hall, 1980

[IEEE828-98] IEEE Std 828-1998, IEEE Standard for Software Configuration Management Plans, IEEE, 1998.

Eduardo Romero Moreno, Migración Escritorio Software Libre, 2011

CHAPTER 21

Referencias

ConfigPackages

New debian policy

rpm.org

Metrica v3

CHAPTER 22

Glosario de términos

Auditoría. Análisis cuyo objetivo es revisar y evaluar la gestión efectuada.

Alerta. Aviso para que el usuario preste su atención a una determinada situación.

Atributo. Valor concreto que toma una propiedad al ser ejecutada en un equipo.

Cambio. Actividad que modifica un ECS, generando una nueva versión del ECS.

ECS. Elemento de configuración software. Objeto software sometido a la GCS.

Falla. Hecho negativo que se produce en un equipo cliente

GCS. Gestión de la configuración software. Proceso de la Ingeniería del Software que identifica, hace seguimiento y controla cada uno de los cambios que se producen en los sistemas.

Liberación. Actividad de situar una versión del ECS en un repositorio para que los clientes del ECS puedan acceder a él e instalarlo.

Metadatos. Información sobre los datos.

Paquete. Contenedor que encapsula un conjunto de ECS liberados junto a sus metadatos.

Paquete huérfano. Paquete que no está en ningún repositorio.

PMS. Package Management System - Gestor de paquetes. Programa que permite aplicar en un equipo, los cambios que han sido liberados en los repositorios.

Propiedad. Código que devolverá una determinada característica de los equipos o de los usuarios.

Repositorio. Almacén de ECS.

Repositorio migasfree. Añade a un repositorio la capacidad de establecer cuándo y quién tiene acceso al repositorio.

Systems Management System. Gestor de sistemas. Software que permite la administración de escritorios de manera centralizada.

CHAPTER 23

API

> La belleza es el acuerdo entre el contenido y la forma.
>
> —Henrik Johan Ibsen

El objetivo de este capítulo es describir la API de migasfree.

Migasfree tiene 2 tipos de API:

- Pública:
  - Son accesibles públicamente.
  - Utilizan el metodo GET de HTTP para el envío de parámetros
- Privada:
  - Comparten entre sí la misma manera de llamar a las funciones.
  - Envían un fichero en el parámetro `message` del método POST HTTP, cuyo nombre de fichero debe usar el siguiente formato:

```
<COMPUTER_NAME>.<UUID>.<API_FUNCTION>
```

  - En el contenido del fichero `message` se introduce, en formato json lo que denominamos el `API Private Data` con los parámetros de entrada de las funciones.
  - En la mayoría de los casos este `API Private Data` es firmado usando la KEY de la `Version` ó del `Empaquetador` añadiendo esta firma al final del fichero:

```
``API Private Data`` + sign(``API Private Data``)
```

  - En el menor de los casos estas funciones en vez de firma requieren el uso de un usuario y contraseña que simplemente se introduce en el propio `API Private Data`.
  - Los valores devueltos por la funciones se reciben en formato json convenientemente firmados por el servidor, siguiendo la estructura:

```
{"<API_FUNCTION>.return": ``API Private Data`` } +
    sign({"<API_FUNCTION>.return": ``API Private Data`` })
```

23.1 get_versions

Pertenece a la API pública.

Devuelve una lista de diccionarios de las plataformas con sus `versiones` correspondientes.

23.1.1 Parámetros de entrada

- Ninguno

23.1.2 Salida

- Lista de diccionarios de plataformas:
  - **plafform**: Nombre de la plataforma.
  - **versions**: Lista de diccionarios de versiones:
    - **name**: Nombre de la version.

Veamos un ejemplo. Si en un navegador web accedemos a la siguiente dirección:

```
http://miservidor/get_versions
```

Obtendremos una cadena de texto en formato json parecida a esta:

```
[{"platform": "Linux", "versions": [{"name": "ACME-1"}, {"name": "debian-7.0"}]}]
```

23.2 get_computer_info

Pertenece a la API pública.

Obtiene un diccionario con información relevante del ordenador consultado.

23.2.1 Parámetros de entrada

- **uuid**: Como parámetro de método GET en la petición HTTP debe indicarse el identificador único de la placa base del ordenador.

23.2.2 Salida

- Diccionario de datos del equipo:
  - **search**: Valor del primer campo indicado en la lista `MIGASFREE_COMPUTER_SEARCH_FIELDS` de los *Ajustes del servidor migasfree* y que sirve para facilitar la identificación del equipo en vez de usar el uuid.
  - **name**: Nombre del ordenador.
  - **tags**: Lista de cadenas de texto con los nombres de las `Etiquetas` asignadas al ordenador.
  - **available_tags**: Diccionario de Propiedades de tipo `Etiqueta`.
    - **<Propiedad>**: Lista de cadenas de texto con el nombre de las etiquetas.
  - **helpdesk**: Cadena de texto MIGASFREE_HELP_DESK de los *Ajustes del servidor migasfree*
  - **id**: Número identificador del ordenador en la tabla de `Ordenadores`
  - **uuid**: Identificador único de la placa base del ordenador

Veamos un ejemplo accediendo a:

```
http://miservidor/get_computer_info/?uuid=E9E66900-CBD4-9A47-B2EC-6ED0367A3AFB
```

obtendríamos algo parecido a esto:

```
{"search": 2, "name": "debian-client", "tags": [], "available_tags": {},
"helpdesk": "Put here how you want to be found", "id": 2,
"uuid": "E9E66900-CBD4-9A47-B2EC-6ED0367A3AFB"}
```

23.3 computer_label

Pertenece a la API pública.

Obtiene una página html que muestra la etiqueta que debe pegarse físicamente en el equipo para facilitar su identificación aún estando éste apagado.

Es utilizada por el comando `migasfree-label`.

23.3.1 Parámetros de entrada

- **uuid**: Como parámetro de método GET en la petición HTTP debe indicarse el identificador único de la placa base del ordenador.

23.3.2 Salida

- La página **html** de la etiqueta:

Por ejemplo al ejecutar:

```
http://miservidor/computer_label/?uuid=E9E66900-CBD4-9A47-B2EC-6ED0367A3AFB
```

Podemos obtener algo como:

Fig. 23.1: Comando migasfree-label.

23.4 register_computer

Pertenece a la API Privada.

Necesita usuario y contraseña con permisos de lectura/escritura en `Ordenadores`, y en `Plataformas` y `Versiones` si `MIGASFREE_AUTOREGISTER` está activo. Ver *Ajustes del servidor migasfree*

Esta función realiza lo siguiente:

- Registra el `Ordenador` en el servidor.
- Añade la `Plataforma` y/o `Version` del ordenador si no existen, siempre y cuando `MIGASFREE_AUTOREGISTER` esté activo.
- Añade las correspondientes `Notificaciones`
- Como resultado se obtendrán las KEYS de la `Version` que usarán las funciones de la API pública que las requieren.

23.4.1 API Private Data Input

- **username**: Nombre del usuario
- **pasword**: Contraseña
- **platform**: Plataforma del ordenador.
- **version**: Version del ordenador.
- **pms**: Sistema de paqeuetería.
- **ip**: Dirección ip.

23.4.2 API Private Data Output

- **migasfree-server.pub**: KEY pública del servidor,
- **migasfree-client.pri**: KEY privada de la versión.
- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.5 get_key_packager

Pertenece a la API Privada.

Necesita usuario y contraseña con permisos de lectura/escritura en `Paquetes`.

Obtiene la `KEY` que permitirá subir paquetes al servidor con el comando de cliente `migasfree-upload`.

23.5.1 API Private Data Input

- **username**: Nombre del usuario.
- **password**: Contraseña del usuario.

23.5.2 API Private Data Output

- **migasfree-server.pub**: KEY pública del servidor,
- **migasfree-packager.pri**: KEY privada del Empaquetador.
- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.6 upload_server_package

Pertenece a la API Privada.

Necesita firmar con KEY de Empaquetador.

Sube un paquete al servidor.

El fichero del paquete debe enviarse en `HttpRequest.FILES["package"]`

23.6.1 API Private Data Input

- **version**: Versión.
- **store**: Ubicación donde se almacena el paquete.
- **source**: Valor booleano que indica si el paquete es el binario ó el fuente .

23.6.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.7 upload_server_set

Pertenece a la API Privada.

Necesita firmar con KEY de Empaquetador.

Sube un paquete de un `Conjunto de Paquetes` al servidor .

El fichero del paquete debe enviarse en `HttpRequest.FILES["package"]`

23.7.1 API Private Data Input

- **version**: Versión.
- **store**: Ubicación donde se almacena el paquete.
- **packageset**: `Conjunto de Paquetes` en el que está incluido el paquete.

23.7.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.8 create_repositories_of_packageset

Pertenece a la API Privada.

Necesita firmar con KEY de Empaquetador.

Se utiliza para forzar la creación de los metadatos de los `Repositorios` en donde está asignado el `Conjunto de Paquetes` especificado.

Se usa despues de subir todos los paquetes de un `Conjunto de Paquetes`.

23.8.1 API Private Data Input

- **packageset**: El nombre del `Conjunto de Paquetes`.
- **version**: La `Version` del `Conjunto de Paquetes`.

23.8.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.9 upload_computer_message

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Envia un mensaje de texto al servidor informando que proceso esta realizando el cliente. Es utilizado por `migasfree --update`

23.9.1 API Private Data Input

El mensaje de texto que se quiere enviar al servidor.

23.9.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.10 get_properties

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Obtiene las Propiedades activas en el servidor migasfree.

23.10.1 API Private Data Input

No requiere.

23.10.2 API Private Data Output

- **properties**: Lista de diccionarios con las Propiedades:
  - **prefix**: Prefijo de la propiedad
  - **function** Instrucciones de la Propiedad
  - **language** Lenguaje en que está programado la propiedad.
- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.11 upload_computer_info

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Dados los datos del ordenador Obtiene del servidor diferente información con lo que el cliente debe hacer para realizar una actuliazación.

23.11.1 API Private Data Input

- **computer**: Diccionario con información relativa al `Ordenador`
  - **hostname**: Nombre del ordenador.
  - **ip**: Dirección ip del ordenador.
  - **platform**: Plataforma.
  - **version**: Nombre de la Version.
  - **user**: Cuenta del usuario que esta logueado en la sesión gráfica.
  - **user_fullname**: Nombre completo del usuario
- **attributes**: Lista de diccionarios con los `Atributos` conseguidos al ejecutar cada una de las `Propiedades`
  - **<ATTRIBUTES_NAME>**: Valor del Atributo

23.11.2 API Private Data Output

- **faultsdef**: Lista de diccionarios de `Definiciones de Fallas`
  - **name**: Nombre de la falla.
  - **function**: Instrucciones de la falla.
  - **language**: Lenguaje en que está escrita la falla.
- **repositories**: Lista de diccionarios de repositorios que deben configurarse en el cliente y que han sido seleccionados por el servidor en función de los atributos de entrada y la fecha actual.
  - **name**:
- **packages**: Diccionario de paquetes.
  - **install**: Lista de cadenas de texto con los paquetes a instalar.
  - **remove**: Lista de cadenas de texto con los paquetes a desinstalar.
  - **base**: `True` si es el ordenador de referencia.
  - **hardware_capture**: `True` si el ordenador tiene que enviar el hardware.
  - **devices**: #TODO
- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.12 upload_computer_faults

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Sube el resultado de las `Fallas`.

23.12.1 API Private Data Input

Diccionario con las Fallas:

- **<PROPIEDAD>**: Texto de la salida estándar al ejecutar la `FALLA`

23.12.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.13 upload_computer_hardware

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Sube el hardware del `Ordenador`.

23.13.1 API Private Data Input

Salida en formato json del comando `lshw`.

23.13.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.14 upload_computer_software_base_diff

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Sube la diferencia respecto al ordenador base

23.14.1 API Private Data Input

Texto con la lista de paquetes respecto al ordenador base separados por retornos de carro.

23.14.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.15 upload_computer_software_base

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Lo utiliza el `Ordenador` de referencia para informar de los paquetes que tiene instalados

23.15.1 API Private Data Input

Texto con la lista de paquetes instalados separados por retornos de carro.

23.15.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.16 upload_computer_software_history

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Informa de cambio en el software.

23.16.1 API Private Data Input

Texto con el cambio de paquetes producidos en el `Ordenador`. Sigue el formato:

```
# [<FECHA DESDE>, <FECHA_HASTA]
<ACTION><PACKAGE> ,
<ACTION><PACKAGE> , ...
```

dónde ACTION puede ser (-) para indicar desintalado y (+) para indicar paquete instalado.

23.16.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.17 get_computer_software

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Obtiene el conjunto de paquetes del `Ordenador` de referencia.

23.17.1 API Private Data Input

No requiere.

23.17.2 API Private Data Output

- Texto con la lista de paquetes del `Ordenador` de referencia separados por retorno de carro
- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.18 upload_computer_errors

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Sube los errores producidos en el cliente.

23.18.1 API Private Data Input

Texto con el errores que han producido en el cliente.

23.18.2 API Private Data Output

- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.19 get_computer_tags

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Obtiene las etiquetas del `Ordenador` y las disponibles en el sistema.

23.19.1 API Private Data Input

No requerido

23.19.2 API Private Data Output

- **selected**: Lista de textos con las `Etiquetas` asignadas al ordenador.
- **available**: Diccionario de Propiedades
  - **<PROPERTY>**: Lista de textos con las `Etiquetas` disponibles por cada `Propiedad` de tipo ``tag`
- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.20 set_computer_tags

Pertenece a la API Privada.

Necesita firmar con KEY de Version.

Asigna las etiquetas al ordenador y como resultado se obtiene los paquetes que deben instalarse y desinstalarse en función de las etiquetas que anteriormente tuviera asignadas el equipo.

23.20.1 API Private Data Input

- **tags**: Lista de Etiquetas a asignar al `Ordenador`

23.20.2 API Private Data Output

- **packages**: Diccionario con la listas de paquetes.
  - **preinstall**: Lista de nombres de paquetes separados por espacios obtenidos del campo `default preinstall packages`
  - **install**: Lista de nombres de paquetes separados por espacios obtenidos del campo `default install packages`
  - **remove**:Lista de nombres de paquetes separados por espacios obtenidos del campo `default remove packages`
- **errmfs**: Diccionario con el posible error devuelto.
  - **code**: Código del error. Un valor de cero indica que no ha habido error.
  - **info**: Texto desciptivo del error.

23.21 get_device

Pendiente de implementar.

23.22 get_assist_devices

Pendiente de implementar.

23.23 install_device

Pendiente de implementar.

23.24 remove_device

Pendiente de implementar.

CHAPTER 24

GNU Free Documentation License

Version 1.3, 3 November 2008

0. PREAMBLE

The purpose of this License is to make a manual, textbook, or other functional and useful document "free" in the sense of freedom: to assure everyone the effective freedom to copy and redistribute it, with or without modifying it, either commercially or noncommercially. Secondarily, this License preserves for the author and publisher a way to get credit for their work, while not being considered responsible for modifications made by others.

This License is a kind of "copyleft", which means that derivative works of the document must themselves be free in the same sense. It complements the GNU General Public License, which is a copyleft license designed for free software.

We have designed this License in order to use it for manuals for free software, because free software needs free documentation: a free program should come with manuals providing the same freedoms that the software does. But this License is not limited to software manuals; it can be used for any textual work, regardless of subject matter or whether it is published as a printed book. We recommend this License principally for works whose purpose is instruction or reference.

1. APPLICABILITY AND DEFINITIONS

This License applies to any manual or other work, in any medium, that contains a notice placed by the copyright holder saying it can be distributed under the terms of this License. Such a notice grants a world-wide, royalty-free license, unlimited in duration, to use that work under the conditions stated herein. The "Document", below, refers to any such manual or work. Any member of the public is a licensee, and is addressed as "you". You accept the license if you copy, modify or distribute the work in a way requiring permission under copyright law.

A "Modified Version" of the Document means any work containing the Document or a portion of it, either copied verbatim, or with modifications and/or translated into another language.

A "Secondary Section" is a named appendix or a front-matter section of the Document that deals exclusively with the relationship of the publishers or authors of the Document to the Document's overall subject (or to related matters) and contains nothing that could fall directly within that overall subject. (Thus, if the Document is in part a textbook of mathematics, a Secondary Section may not explain any mathematics.) The relationship could be a matter of historical connection with the subject or with related matters, or of legal, commercial, philosophical, ethical or political position regarding them.

The "Invariant Sections" are certain Secondary Sections whose titles are designated, as being those of Invariant Sections, in the notice that says that the Document is released under this License. If a section does not fit the above definition of Secondary then it is not allowed to be designated as Invariant. The Document may contain zero Invariant Sections. If the Document does not identify any Invariant Sections then there are none.

The "Cover Texts" are certain short passages of text that are listed, as Front-Cover Texts or Back-Cover Texts, in the notice that says that the Document is released under this License. A Front-Cover Text may be at most 5 words, and a Back-Cover Text may be at most 25 words.

A "Transparent" copy of the Document means a machine-readable copy, represented in a format whose specification is available to the general public, that is suitable for revising the document straightforwardly with generic text editors or (for images composed of pixels) generic paint programs or (for drawings) some widely available drawing editor, and that is suitable for input to text formatters or for automatic translation to a variety of formats suitable for input to text formatters. A copy made in an otherwise Transparent file format whose markup, or absence of markup, has been arranged to thwart or discourage subsequent modification by readers is not Transparent. An image format is not Transparent if used for any substantial amount of text. A copy that is not "Transparent" is called "Opaque".

Examples of suitable formats for Transparent copies include plain ASCII without markup, Texinfo input format, LaTeX input format, SGML or XML using a publicly available DTD, and standard-conforming simple HTML, PostScript or PDF designed for human modification. Examples of transparent image formats include PNG, XCF and JPG. Opaque formats include proprietary formats that can be read and edited only by proprietary word processors, SGML or XML for which the DTD and/or processing tools are not generally available, and the machine-generated HTML, PostScript or PDF produced by some word processors for output purposes only.

The "Title Page" means, for a printed book, the title page itself, plus such following pages as are needed to hold, legibly, the material this License requires to appear in the title page. For works in formats which do not have any title page as such, "Title Page" means the text near the most prominent appearance of the work's title, preceding the beginning of the body of the text.

The "publisher" means any person or entity that distributes copies of the Document to the public.

A section "Entitled XYZ" means a named subunit of the Document whose title either is precisely XYZ or contains XYZ in parentheses following text that translates XYZ in another language. (Here XYZ stands for a specific section name mentioned below, such as "Acknowledgements", "Dedications", "Endorsements", or "History".) To "Preserve the Title" of such a section when you modify the Document means that it remains a section "Entitled XYZ" according to this definition.

The Document may include Warranty Disclaimers next to the notice which states that this License applies to the Document. These Warranty Disclaimers are considered to be included by reference in this License, but only as regards disclaiming warranties: any other implication that these Warranty Disclaimers may have is void and has no effect on the meaning of this License.

2. VERBATIM COPYING

You may copy and distribute the Document in any medium, either commercially or noncommercially, provided that this License, the copyright notices, and the license notice saying this License applies to the Document are reproduced in all copies, and that you add no other conditions whatsoever to those of this License. You may not use technical measures to obstruct or control the reading or further copying of the copies you make or distribute. However, you may accept compensation in exchange for copies. If you distribute a large enough number of copies you must also follow the conditions in section 3.

You may also lend copies, under the same conditions stated above, and you may publicly display copies.

3. COPYING IN QUANTITY

If you publish printed copies (or copies in media that commonly have printed covers) of the Document, numbering more than 100, and the Document's license notice requires Cover Texts, you must enclose the copies in covers that carry, clearly and legibly, all these Cover Texts: Front-Cover Texts on the front cover, and Back-Cover Texts on the back cover. Both covers must also clearly and legibly identify you as the publisher of these copies. The front cover must present the full title with all words of the title equally prominent and visible. You may add other material on the covers in addition. Copying with changes limited to the covers, as long as they preserve the title of the Document and satisfy these conditions, can be treated as verbatim copying in other respects.

If the required texts for either cover are too voluminous to fit legibly, you should put the first ones listed (as many as fit reasonably) on the actual cover, and continue the rest onto adjacent pages.

If you publish or distribute Opaque copies of the Document numbering more than 100, you must either include a machine-readable Transparent copy along with each Opaque copy, or state in or with each Opaque copy a computer-network location from which the general network-using public has access to download using public-standard network protocols a complete Transparent copy of the Document, free of added material. If you use the latter option, you must take reasonably prudent steps, when you begin distribution of Opaque copies in quantity, to ensure that this Transparent copy will remain thus accessible at the stated location until at least one year after the last time you distribute an Opaque copy (directly or through your agents or retailers) of that edition to the public.

It is requested, but not required, that you contact the authors of the Document well before redistributing any large number of copies, to give them a chance to provide you with an updated version of the Document.

4. MODIFICATIONS

You may copy and distribute a Modified Version of the Document under the conditions of sections 2 and 3 above, provided that you release the Modified Version under precisely this License, with the Modified Version filling the role of the Document, thus licensing distribution and modification of the Modified Version to whoever possesses a copy of it. In addition, you must do these things in the Modified Version:

1. Use in the Title Page (and on the covers, if any) a title distinct from that of the Document, and from those of previous versions (which should, if there were any, be listed in the History section of the Document). You may use the same title as a previous version if the original publisher of that version gives permission.
2. List on the Title Page, as authors, one or more persons or entities responsible for authorship of the modifications in the Modified Version, together with at least five of the principal authors of the Document (all of its principal authors, if it has fewer than five), unless they release you from this requirement.
3. State on the Title page the name of the publisher of the Modified Version, as the publisher.
4. Preserve all the copyright notices of the Document.
5. Add an appropriate copyright notice for your modifications adjacent to the other copyright notices.
6. Include, immediately after the copyright notices, a license notice giving the public permission to use the Modified Version under the terms of this License, in the form shown in the Addendum below.
7. Preserve in that license notice the full lists of Invariant Sections and required Cover Texts given in the Document's license notice.
8. Include an unaltered copy of this License.
9. Preserve the section Entitled "History", Preserve its Title, and add to it an item stating at least the title, year, new authors, and publisher of the Modified Version as given on the Title Page. If there is no section Entitled "History" in the Document, create one stating the title, year, authors, and publisher of the Document as given on its Title Page, then add an item describing the Modified Version as stated in the previous sentence.
10. Preserve the network location, if any, given in the Document for public access to a Transparent copy of the Document, and likewise the network locations given in the Document for previous versions it was based on. These may be placed in the "History" section. You may omit a network location for a work that was published at least four years before the Document itself, or if the original publisher of the version it refers to gives permission.
11. For any section Entitled "Acknowledgements" or "Dedications", Preserve the Title of the section, and preserve in the section all the substance and tone of each of the contributor acknowledgements and/or dedications given therein.
12. Preserve all the Invariant Sections of the Document, unaltered in their text and in their titles. Section numbers or the equivalent are not considered part of the section titles.
13. Delete any section Entitled "Endorsements". Such a section may not be included in the Modified Version.
14. Do not retitle any existing section to be Entitled "Endorsements" or to conflict in title with any Invariant Section.
15. Preserve any Warranty Disclaimers.

If the Modified Version includes new front-matter sections or appendices that qualify as Secondary Sections and contain no material copied from the Document, you may at your option designate some or all of these sections as invariant. To do this, add their titles to the list of Invariant Sections in the Modified Version's license notice. These titles must be distinct from any other section titles.

You may add a section Entitled "Endorsements", provided it contains nothing but endorsements of your Modified Version by various parties–for example, statements of peer review or that the text has been approved by an organization as the authoritative definition of a standard.

You may add a passage of up to five words as a Front-Cover Text, and a passage of up to 25 words as a Back-Cover Text, to the end of the list of Cover Texts in the Modified Version. Only one passage of Front-Cover Text and one of Back-Cover Text may be added by (or through arrangements made by) any one entity. If the Document already includes a cover text for the same cover, previously added by you or by arrangement made by the same entity you are acting on behalf of, you may not add another; but you may replace the old one, on explicit permission from the previous publisher that added the old one.

The author(s) and publisher(s) of the Document do not by this License give permission to use their names for publicity for or to assert or imply endorsement of any Modified Version.

5. COMBINING DOCUMENTS

You may combine the Document with other documents released under this License, under the terms defined in section 4 above for modified versions, provided that you include in the combination all of the Invariant Sections of all of the original documents, unmodified, and list them all as Invariant Sections of your combined work in its license notice, and that you preserve all their Warranty Disclaimers.

The combined work need only contain one copy of this License, and multiple identical Invariant Sections may be replaced with a single copy. If there are multiple Invariant Sections with the same name but different contents, make the title of each such section unique by adding at the end of it, in parentheses, the name of the original author or publisher of that section if known, or else a unique number. Make the same adjustment to the section titles in the list of Invariant Sections in the license notice of the combined work.

In the combination, you must combine any sections Entitled "History" in the various original documents, forming one section Entitled "History"; likewise combine any sections Entitled "Acknowledgements", and any sections Entitled "Dedications". You must delete all sections Entitled "Endorsements".

6. COLLECTIONS OF DOCUMENTS

You may make a collection consisting of the Document and other documents released under this License, and replace the individual copies of this License in the various documents with a single copy that is included in the collection, provided that you follow the rules of this License for verbatim copying of each of the documents in all other respects.

You may extract a single document from such a collection, and distribute it individually under this License, provided you insert a copy of this License into the extracted document, and follow this License in all other respects regarding verbatim copying of that document.

7. AGGREGATION WITH INDEPENDENT WORKS

A compilation of the Document or its derivatives with other separate and independent documents or works, in or on a volume of a storage or distribution medium, is called an "aggregate" if the copyright resulting from the compilation is not used to limit the legal rights of the compilation's users beyond what the individual works permit. When the Document is included in an aggregate, this License does not apply to the other works in the aggregate which are not themselves derivative works of the Document.

If the Cover Text requirement of section 3 is applicable to these copies of the Document, then if the Document is less than one half of the entire aggregate, the Document's Cover Texts may be placed on covers that bracket the Document within the aggregate, or the electronic equivalent of covers if the Document is in electronic form. Otherwise they must appear on printed covers that bracket the whole aggregate.

8. TRANSLATION

Translation is considered a kind of modification, so you may distribute translations of the Document under the terms of section 4. Replacing Invariant Sections with translations requires special permission from their copyright holders, but you may include translations of some or all Invariant Sections in addition to the original versions of these Invariant Sections. You may include a translation of this License, and all the license notices in the Document, and any Warranty Disclaimers, provided that you also include the original English version of this License and the original versions of those notices and disclaimers. In case of a disagreement between the translation and the original version of this License or a notice or disclaimer, the original version will prevail.

If a section in the Document is Entitled "Acknowledgements", "Dedications", or "History", the requirement (section 4) to Preserve its Title (section 1) will typically require changing the actual title.

9. TERMINATION

You may not copy, modify, sublicense, or distribute the Document except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, or distribute it is void, and will automatically terminate your rights under this License.

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, receipt of a copy of some or all of the same material does not give you any rights to use it.

10. FUTURE REVISIONS OF THIS LICENSE

The Free Software Foundation may publish new, revised versions of the GNU Free Documentation License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns. See http://www.gnu.org/copyleft/.

Each version of the License is given a distinguishing version number. If the Document specifies that a particular numbered version of this License "or any later version" applies to it, you have the option of following the terms and conditions either of that specified version or of any later version that has been published (not as a draft) by the Free Software Foundation. If the Document does not specify a version number of this License, you may choose any version ever published (not as a draft) by the Free Software Foundation. If the Document specifies that a proxy can decide which future versions of this License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Document.

11. RELICENSING

"Massive Multiauthor Collaboration Site" (or "MMC Site") means any World Wide Web server that publishes copyrightable works and also provides prominent facilities for anybody to edit those works. A public wiki that anybody can edit is an example of such a server. A "Massive Multiauthor Collaboration" (or "MMC") contained in the site means any set of copyrightable works thus published on the MMC site.

"CC-BY-SA" means the Creative Commons Attribution-Share Alike 3.0 license published by Creative Commons Corporation, a not-for-profit corporation with a principal place of business in San Francisco, California, as well as future copyleft versions of that license published by that same organization.

"Incorporate" means to publish or republish a Document, in whole or in part, as part of another Document.

An MMC is "eligible for relicensing" if it is licensed under this License, and if all works that were first published under this License somewhere other than this MMC, and subsequently incorporated in whole or in part into the MMC, (1) had no cover texts or invariant sections, and (2) were thus incorporated prior to November 1, 2008.

The operator of an MMC Site may republish an MMC contained in the site under CC-BY-SA on the same site at any time before August 1, 2009, provided the MMC is eligible for relicensing.

ADDENDUM: How to use this License for your documents

To use this License in a document you have written, include a copy of the License in the document and put the following copyright and license notices just after the title page:

> Copyright (c) YEAR YOUR NAME. Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

If you have Invariant Sections, Front-Cover Texts and Back-Cover Texts, replace the "with...Texts." line with this:

> with the Invariant Sections being LIST THEIR TITLES, with the Front-Cover Texts being LIST, and with the Back-Cover Texts being LIST.

If you have Invariant Sections without Cover Texts, or some other combination of the three, merge those two alternatives to suit the situation.

If your document contains nontrivial examples of program code, we recommend releasing these examples in parallel under your choice of free software license, such as the GNU General Public License, to permit their use in free software.