Copia y creación de repositorios de paquetes en Oracle® Solaris 11.2

Referencia: E53760-02
Septiembre de 2014

Contenido

Lista de ejemplos

Uso de esta documentación

- **Descripción general**: describe cómo crear, copiar, facilitar, actualizar y mantener un repositorio de paquete de software con la función Image Packaging System (IPS) de Oracle Solaris.
- **Destinatarios**: administradores del sistema que instalan y gestionan software o que ayudan a otros a instalar y gestionar software.
- **Conocimiento necesario**: experiencia con la función utilidad de gestión de servicios (SMF) de Oracle Solaris y con la gestión de NFS y servidores web.

Biblioteca de documentación del producto

En la biblioteca de documentación, que se encuentra en http://www.oracle.com/pls/topic/lookup?ctx=E56339, se incluye información de última hora y problemas conocidos para este producto.

Acceso a My Oracle Support

Los clientes de Oracle disponen de asistencia a través de Internet en el portal My Oracle Support. Para obtener más información, visite http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info o, si tiene alguna discapacidad auditiva, visite http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs.

Comentarios

Envíenos comentarios acerca de esta documentación mediante http://www.oracle.com/goto/docfeedback.

♦ ♦ ♦ CAPÍTULO 1

Repositorios de empaquetado de Image Packaging System

El software Oracle Solaris 11 se distribuye en paquetes Image Packaging System (IPS). Los paquetes de IPS se almacenan en repositorios de paquetes de IPS que los editores IPS rellenan.

En esta guía, se describe cómo crear un repositorio de paquetes de software mediante la función Image Packaging System (IPS) de Oracle Solaris. Las herramientas de IPS permiten copiar fácilmente un repositorio existente o crear un repositorio de paquetes propio. También permiten actualizar fácilmente los paquetes del repositorio. Puede proporcionar una interfaz de archivo o una interfaz HTTP o HTTPS para los usuarios del repositorio. En esta guía también se describe cómo actualizar automáticamente el repositorio y cómo clonar un repositorio, y se muestra la configuración del servidor web Apache como, por ejemplo, el almacenamiento en caché, el equilibrio de carga y la configuración del acceso HTTPS.

Este capítulo proporciona:

- Los motivos para crear un repositorio de paquetes de IPS local para uso interno
- Prácticas recomendadas para la creación de repositorios de paquetes
- Requisitos del sistema para alojar un repositorio

Repositorios de IPS locales

Quizá desee crear un repositorio de IPS local por los siguientes motivos:

- **Rendimiento y seguridad.** No desea que los sistemas cliente recurran a Internet para recuperar paquetes de software nuevos ni para actualizar paquetes existentes.
- **Control de acceso.** Desea asegurarse de que el próximo año podrá realizar la misma instalación que está realizando hoy. Desea controlar fácilmente las versiones a las que se puede actualizar los sistemas.
- **Paquetes personalizados.** Desea entregar paquetes IPS personalizados.

Mejores prácticas para la creación y el uso de repositorios de paquetes IPS locales

Emplee las siguientes prácticas recomendadas para mantener la disponibilidad del repositorio y minimizar los errores.

Incluya todo el contenido de las actualizaciones de repositorio de asistencia (SRU).

Mantenga los repositorios locales actualizados con todas las actualizaciones de compatibilidad. Las actualizaciones de compatibilidad contienen actualizaciones de seguridad y otras correcciones importantes. Cada versión y actualización pequeñas del repositorio de paquetes del sistema operativo Oracle Solaris se publica como un conjunto completo de paquetes. Las SRU se publican como una actualización dispersa de sólo los paquetes cambiados.

- No agregue un subconjunto de paquetes desde una actualización de compatibilidad al repositorio. Agregue todo el contenido de la actualización de compatibilidad al repositorio local.
- No omita una actualización de compatibilidad. Acumule todas las actualizaciones de compatibilidad aplicables en cada repositorio.
- No elimine paquetes ofrecidos por un editor de Oracle.
- Utilice el servicio utilidad de gestión de servicios (SMF) `svc:/application/pkg/mirror` para actualizar automáticamente el repositorio maestro local desde el repositorio de asistencia de Oracle. Consulte Cómo copiar un repositorio de Internet de forma automática [23] para obtener instrucciones.

Los usuarios pueden actualizar a una versión anterior a la versión más reciente en el repositorio mediante la especificación de la versión de `todo` el paquete de incorporación a instalar. Consulte Capítulo 4, “Actualización de una imagen de Oracle Solaris” de “Agregación y actualización de software en Oracle Solaris 11.2 ”.

Verifique cada vez que actualice el repositorio.

Utilice el comando `pkgrepo verify` siempre que cambie el contenido o los valores de propiedad del repositorio. El comando `pkgrepo verify` verifica que los siguientes atributos del contenido de repositorio sean correctos:

- Sumas de comprobación de archivo.
- Permisos de archivo. Los archivos y directorios del repositorio y la ruta de acceso al repositorio se comprueban para garantizar que el usuario `pkg5srv` pueda leer el contenido del repositorio.
- Permisos del manifiesto de paquete.
- Contenido del manifiesto de paquete.
- Firmas de paquetes.

Cree repositorios en una ubicación compartida.

Una ubicación compartida es una ubicación que no está en ningún entorno de inicio (BE). Algunos ejemplos de ubicaciones compartidas son `/var/share` y `/export`. La creación de un repositorio en una ubicación compartida proporciona las siguientes ventajas:

- El repositorio está fácilmente disponible desde otros entornos de inicio existentes.
- Al crear un nuevo entorno de inicio mediante la actualización o la clonación de un entorno de inicio existente, no desperdicia espacio en varias copias de un repositorio.
- No pierde tiempo y recursos de E/S volviendo a aplicar las actualizaciones del repositorio que ya realizó en un entorno de inicio diferente.

Si utiliza zonas no globales, todas las ubicaciones de los editores configuradas en zonas no globales deben estar accesibles desde la zona global, incluso si ese editor no está configurado en la zona global.

Cree cada repositorio en su propio sistema de archivos ZFS.

El uso de un sistema de archivos ZFS independiente le permite hacer lo siguiente:

- Obtener un mayor rendimiento.
- Establecer las distintas características del sistema de archivos. Por ejemplo, establezca `atime` en `off` para un mejor rendimiento al actualizar el repositorio. La propiedad `atime` controla si la hora de acceso de los archivos se actualiza cuando los archivos se leen. Con la desactivación de esta propiedad, se evita generar tráfico de escritura al leer los archivos.
- Gestionar el uso de recursos. Especifique una cuota de disco adecuada para cada conjunto de datos de repositorio para asegurarse de que las actualizaciones grandes de repositorio no consuman todo el espacio en la agrupación. Esta práctica recomendada es especialmente importante si realiza actualizaciones automáticamente como se describe en Cómo copiar un repositorio de Internet de forma automática [23].
- Crear instantánea.

Realice una instantánea cada vez que se actualiza el repositorio.

Realice una instantánea del sistema de archivos de repositorio cada vez que actualiza el repositorio para obtener las siguientes ventajas:

- Revertir a una versión anterior del repositorio desde una instantánea.
- Actualizar el repositorio desde una instantánea para minimizar las interrupciones de usuario.

Proporcione alta disponibilidad.

- Mantener clones del repositorio en diferentes ubicaciones. Consulte “Mantenimiento de varios repositorios locales idénticos” [37] para obtener instrucciones.
- Configurar el servidor web para el almacenamiento en caché, el equilibrio de carga y cómo servir varios repositorios. Consulte Capítulo 5, Ejecución del servidor de depósitos detrás de un servidor web para obtener más información.

Proteja los repositorios locales.

Consulte “Configuración del acceso HTTPS al repositorio” [56] para obtener instrucciones.

Requisitos del sistema

El sistema que aloja los repositorios de paquetes de IPS puede ser un sistema basado en x86 o un sistema basado en SPARC.

Sistema operativo

Los servidores de repositorio que ejecutan Oracle Solaris 11 11/11 admiten todos los paquetes de actualización de Oracle Solaris 11.

Espacio en disco

Para alojar una copia del repositorio de la versión Oracle Solaris 11.2, el servidor del repositorio debe tener 16 GB de espacio libre.

Dado que la práctica recomendada consiste en mantener los repositorios locales actualizados con todas las actualizaciones de compatibilidad, planee utilizar entre 10 y 15 GB de espacio adicional cada año para actualizaciones de compatibilidad. El software adicional, como Oracle Solaris Studio u Oracle Solaris Cluster, por supuesto, requiere espacio adicional en el repositorio de paquetes.

Si un sistema aloja más de un repositorio de IPS, cada repositorio debe ser un sistema de archivos ZFS independiente, de manera que cada repositorio se puede deshacer o recuperar por separado.

Privilegios de gestión del repositorio

Utilice uno de los siguientes métodos para obtener el privilegio que necesita para crear y configurar repositorios de paquetes. Consulte “Protección de los usuarios y los procesos en Oracle Solaris 11.2 ” para obtener más información sobre los perfiles y los roles, incluido cómo determinar qué perfil o rol necesita.

Perfiles de derechos

Utilice el comando `profiles` para obtener una lista de los perfiles de derechos que se le han asignado. Los siguientes perfiles son útiles para mantener los repositorios de paquetes locales:

Administración del sistema de archivos ZFS

Este perfil de derechos permite ejecutar el comando `zfs`.

Instalación del software

Este perfil de derechos permite ejecutar el comando `pkg`.

Gestión de servicios

Este perfil de derechos permite ejecutar los comandos SMF como `svccfg`.

Funciones

Utilice el comando `roles` para obtener una lista de los roles que se le hayan asignado. Si tiene el rol `root`, puede utilizar el comando `su` con la contraseña `root` para asumir el rol `root`.

Comando sudo

En función de la política de seguridad de su sitio, es posible que pueda utilizar el comando `sudo` con su contraseña de usuario para ejecutar un comando con privilegios.

♦ ♦ ♦ CAPÍTULO 2

Copia de repositorios de paquetes de IPS

En este capítulo se describen dos formas de crear una copia del repositorio de paquetes IPS de Oracle Solaris: puede utilizar archivos de repositorio desde medios o desde un sitio de descarga de Oracle Solaris, o puede recuperar el contenido del repositorio desde Internet en forma manual o automática. En cualquier caso, cree primero un sistema de archivos ZFS independiente en una ubicación compartida para el repositorio de paquetes local. Después de crear el repositorio, verifique el repositorio y cree una instantánea.

En este capítulo también se proporciona información sobre el rendimiento y la resolución de problemas en relación con la copia de repositorios.

Consideraciones de rendimiento para la copia de repositorios

Si descarga archivos del repositorio desde el sitio de descarga de Oracle Solaris o si utiliza el comando `pkgrecv` que se muestra en “Copia de un repositorio de Internet” [21] para recuperar contenido del repositorio de una ubicación en Internet, tenga en cuenta la siguiente configuración para mejorar el rendimiento de la transferencia:

- Asegúrese de que la capacidad de la agrupación de almacenamiento de ZFS sea inferior al 80%. Utilice el comando `zpool list` para ver la capacidad de la agrupación.
- Si utiliza un proxy, compruebe el rendimiento del proxy.
- Cierre las aplicaciones que utilizan una gran cantidad de memoria.
- Asegúrese de tener suficiente espacio libre en el directorio temporal. Durante sus operaciones, el comando `pkgrecv` utiliza `$TMPDIR` como un directorio de almacenamiento temporal. Si `TMPDIR` no está definido, `pkgrecv` utiliza `/var/tmp` para este almacenamiento temporal. Asegúrese de que `$TMPDIR` o `/var/tmp` tenga suficiente espacio libre para el tamaño de la operación `pkgrecv` que está realizando.
- Si utiliza el comando `pkgrecv` para copiar un gran repositorio, considere utilizar la opción `--clone`. El uso de la opción `--clone` es más rápido y consume menos memoria. Consulte Cómo clonar un repositorio de paquetes IPS local [37].

- Si está utilizando el comando `pkgrecv` para crear o actualizar un gran repositorio, considere utilizar un SSD para el repositorio de destino. Puede mover el repositorio según sea necesario una vez que la recuperación del paquete termine.

Resolución de problemas de los repositorios de paquetes locales

Los siguientes métodos pueden evitar los problemas o ayudar a encontrar la causa de los problemas que pueden surgir:

- Verificar los archivos de origen del repositorio. Si utiliza archivos `.zip` para crear su repositorio, confirme que los archivos del sistema son correctos mediante las sumas de comprobación, como se describe en Cómo copiar un repositorio desde un archivo `zip` [17].
- Verifique el repositorio instalado. Utilice el comando `pkgrepo verify` para comprobar la instalación del repositorio.

  Los siguientes problemas de permisos son informados por `pkgrepo verify`:

  - Permisos de archivo. Para evitar problemas con los permisos de archivo y directorio para los repositorios basados en sistemas de archivos, asegúrese de que el usuario `pkg5srv` tiene permiso para leer el repositorio.
  - Permisos de directorio. Asegúrese de que todos los directorios en el repositorio tienen permiso de ejecución.

  Si el comando `pkgrepo verify` informa otros tipos de errores, intente usar el comando `pkgrepo fix` para corregir los errores. Consulte la página del comando man `pkgrepo`(1) para obtener más información.
- Compruebe el origen del editor. Asegúrese de definir el origen para cada editor de forma adecuada en cada imagen. Para actualizar paquetes instalados, instalar paquetes que dependan de paquetes instalados o instalar una zona no global, el repositorio que configura como el origen del editor debe contener al menos el mismo software que el que está instalado en la imagen en la que está configurando el editor. Consulte el paso 3 en Cómo permitir a los usuarios recuperar paquetes mediante una interfaz de archivo [27]. Consulte “Agregación y actualización de software en Oracle Solaris 11.2 ” para obtener más información sobre la configuración de editores y la resolución de problemas de instalación de paquetes.
- Compruebe la configuración del servidor web. Si configura un servidor web Apache para acceder a su repositorio, configure el servidor web para no decodificar las barras diagonales codificadas. Consulte las instrucciones en “Configuración necesaria de Apache” [50]. Descodificar barras diagonales codificadas puede producir errores de "no se ha encontrado el paquete".

- No cree un repositorio que sólo sea accesible desde una zona no global. Todas las ubicaciones de los editores configurados en zonas no globales deben ser accesibles desde la zona global, incluso si ese editor no está configurado en la zona global.

Copia de un repositorio desde un archivo

En esta sección se describe cómo realizar una copia local del repositorio de paquetes Oracle Solaris de uno o más archivos de repositorio. Los archivos de repositorio pueden estar en medios o disponibles en un sitio de descarga de Oracle Solaris. Los archivos de repositorio pueden ser archivos `zip` o archivos `iso`.

▼ Cómo copiar un repositorio desde un archivo `zip`

1. **Cree un sistema de archivos ZFS para el nuevo repositorio.**

   Cree el repositorio en una ubicación compartida. Establezca `atime` en `off` cuando cree el sistema de archivos de repositorio. Consulte “Mejores prácticas para la creación y el uso de repositorios de paquetes IPS locales” [10].

   ```
   $ zfs create -o atime=off rpool/export/IPSpkgrepos
   $ zfs create rpool/export/IPSpkgrepos/Solaris
   $ zfs get atime rpool/export/IPSpkgrepos/Solaris
   NAME                              PROPERTY  VALUE  SOURCE
   rpool/export/IPSpkgrepos/Solaris  atime     off    inherited from rpool/export/IPSpkgrepos
   ```

2. **Obtenga los archivos del repositorio de paquetes.**

   Descargue los archivos `.zip` del repositorio de paquetes IPS de Oracle Solaris desde la misma ubicación de donde descargó la imagen de instalación del sistema o encuentre el DVD del repositorio en el paquete de medios. Junto con los archivos `.zip`, descargue la secuencia de comandos `install-repo.ksh` y los archivos `.txt` (`README` y los archivos de la suma de comprobación).

   ```
   $ ls
   install-repo.ksh           sol-11_2-ga-repo-3of4.zip
   README-zipped-repo.txt     sol-11_2-ga-repo-4of4.zip
   sol-11_2-ga-repo-1of4.zip  sol-11_2-ga-repo.txt
   sol-11_2-ga-repo-2of4.zip
   ```

3. **Asegúrese de que el archivo de secuencia de comandos sea ejecutable.**

   ```
   $ chmod +x install-repo.ksh
   ```

4. **Ejecute la secuencia de comandos de instalación del repositorio.**

La secuencia de comandos del repositorio, `install-repo.ksh`, descomprime cada archivo del repositorio `.zip` en el directorio especificado. La secuencia de comandos, de manera opcional, realiza las siguientes tareas adicionales:

- Verifique las sumas de comprobación de los archivos `.zip` descargados. Si no especifica la opción `-c` para verificar las sumas de comprobación, compruebe las sumas de comprobación de forma manual antes de ejecutar la secuencia de comandos de instalación del repositorio. Ejecute el siguiente comando `digest` y compare la salida la suma de comprobación correspondiente para el archivo `.md5`:

  ```
  $ digest -a md5 file
  ```
- Agregue el contenido del repositorio a contenido existente si el destino especificado ya contiene un repositorio.
- Verifique el repositorio final. Si no especifica la opción `-v` para verificar el repositorio, utilice los subcomandos `info`, `list` y `verify` del comando `pkgrepo` para verificar el repositorio después de ejecutar la secuencia de comandos de la instalación del repositorio.
- Cree un archivo de imagen ISO para el montaje y la distribución. Si utiliza la opción `-I` para crear un archivo `.iso`, el archivo `.iso` y el archivo `README` que explican cómo utilizar el archivo `.iso` están en el directorio de destino especificado.

5. **Verifique el contenido del repositorio.**

   Si no ha especificado la opción `-v` en el paso anterior, utilice los subcomandos `info`, `list` y `verify` del comando `pkgrepo` para comprobar que el repositorio se haya copiado correctamente. Si el comando `pkgrepo verify` informa errores, intente usar el comando `pkgrepo fix` para corregir los errores. Consulte la página del comando man `pkgrepo`(1).

6. **Realice una instantánea del nuevo repositorio.**

   ```
   $ zfs snapshot rpool/export/IPSpkgrepos/Solaris@sol-11_2_0
   ```

ejemplo 2-1 Creación de un nuevo repositorio desde un archivo `zip`

En este ejemplo, no hay ningún repositorio hasta que los archivos `zip` se desempaqueten. La secuencia de comandos puede realizar las siguientes opciones:

`-s`	Opcional: Especifica la ruta de acceso completa al directorio donde se ubican los archivos `.zip`. De manera predeterminada: el directorio actual.
`-d`	Necesario. Especifica la ruta de acceso completa al directorio donde desea el repositorio.
`-i`	Opcional: Especifica los archivos que se deben utilizar para rellenar este repositorio. El directorio de origen puede contener varios conjuntos de archivos `.zip`. De manera predeterminada: la imagen disponible más nueva en el directorio de origen.

`-c` Opcional: Compara las sumas de comprobación de los archivos `.zip` con las sumas de comprobación en el archivo especificado. Si especifica `-c` sin argumentos, el archivo predeterminado utilizado es el archivo `.md5` para la imagen `-i` en el directorio de origen.

`-v` Opcional: Verifica el repositorio final.

`-I` Opcional: Crea una imagen ISO del repositorio en el directorio de origen. También deja un archivo log `mkiso.log` en el directorio de origen.

`-h` Opcional: Muestra un mensaje de uso.

```
$ ./install-repo.ksh -d /export/IPSpkgrepos/Solaris -c -v -I
Comparing checksums of downloaded files...done. Checksums match.
Uncompressing sol-11_2-ga-repo-1of4.zip...done.
Uncompressing sol-11_2-ga-repo-2of4.zip...done.
Uncompressing sol-11_2-ga-repo-3of4.zip...done.
Uncompressing sol-11_2-ga-repo-4of4.zip...done.
Repository can be found in /export/IPSpkgrepos/Solaris.
Initiating repository verification.
Building ISO image...done.
ISO image and instructions for using the ISO image are at:
/tank/downloads/sol-11_2-ga-repo.iso
/tank/downloads/README-repo-iso.txt
$ ls /export/IPSpkgrepos/Solaris
COPYRIGHT         NOTICES           pkg5.repository   publisher         README-iso.txt
```

La reconstrucción y la verificación del repositorio pueden tardar tiempo, pero el contenido del repositorio es recuperable después de obtener el mensaje "El repositorio se encuentra en".

ejemplo 2-2 Agregación a un repositorio existente desde un archivo `zip`

En este ejemplo, el contenido de los archivos `zip` del repositorio se agregan al contenido en un paquete de repositorio existente.

```
$ pkgrepo -s /export/IPSpkgrepos/Solaris info
PUBLISHER PACKAGES STATUS           UPDATED
solaris   4764     online           2014-03-18T05:30:57.221021Z
$ ./install-repo.ksh -d /export/IPSpkgrepos/Solaris -c -v -I
IPS repository exists at destination /export/IPSpkgrepos/Solaris
Current version: 0.175.2.0.0.35.0
Do you want to add to this repository? (y/n) y
Comparing checksums of downloaded files...done. Checksums match.
Uncompressing sol-11_2-ga-repo-1of4.zip...done.
Uncompressing sol-11_2-ga-repo-2of4.zip...done.
Uncompressing sol-11_2-ga-repo-3of4.zip...done.
Uncompressing sol-11_2-ga-repo-4of4.zip...done.
Repository can be found in /export/IPSpkgrepos/Solaris.
Initiating repository rebuild.
Initiating repository verification.
```

```
Building ISO image...done.
ISO image and instructions for using the ISO image are at:
/tank/downloads/sol-11_2-ga-repo.iso
/tank/downloads/README-repo-iso.txt
$ pkgrepo -s /export/IPSpkgrepos/Solaris info
PUBLISHER PACKAGES STATUS           UPDATED
solaris   4768     online           2014-06-02T18:11:55.640930Z
```

▼ Cómo copiar un repositorio desde un archivo `iso`

1. **Cree un sistema de archivos ZFS para el nuevo repositorio.**

   Cree el repositorio en una ubicación compartida. Establezca `atime` en `off` cuando cree el sistema de archivos de repositorio. Consulte “Mejores prácticas para la creación y el uso de repositorios de paquetes IPS locales” [10].

   ```
   $ zfs create -o atime=off rpool/export/IPSpkgrepos
   $ zfs create rpool/export/IPSpkgrepos/Solaris
   $ zfs get atime rpool/export/IPSpkgrepos/Solaris
   NAME                              PROPERTY  VALUE  SOURCE
   rpool/export/IPSpkgrepos/Solaris  atime     off    inherited from rpool/export/IPSpkgrepos
   ```

2. **Obtenga los archivos de imagen del repositorio de paquetes.**

   Cree un archivo `.iso` a partir de los archivos `.zip` del repositorio con la opción `-I`, como se describe en Ejemplo 2-1, “Creación de un nuevo repositorio desde un archivo `zip`”.

3. **Monte el archivo de imagen.**

   Monte el archivo `.iso` de repositorio para acceder al contenido.

   ```
   $ mount -F hsfs /path/sol-11_2-repo.iso /mnt
   ```

   Para evitar la necesidad de volver a montar la imagen `.iso` cada vez que el sistema del servidor del repositorio se reinicia, copie el contenido del archivo del repositorio como se describe en el siguiente paso.

4. **Copie el contenido del repositorio a la nueva ubicación.**

   Para aumentar el rendimiento de los accesos al repositorio y para evitar la necesidad de volver a montar la imagen `.iso` cada vez que el sistema se reinicia, copie los archivos de repositorio de `/mnt/repo/` a un sistema de archivos ZFS. Puede realizar esta copia con el comando `rsync` o con el comando `tar`

   - **Utilice el comando `rsync`.**

     Si utiliza el comando `rsync`, asegúrese de especificar `/mnt/repo/` (incluida la barra diagonal final) en vez de `/mnt/repo` para copiar los archivos y subdirectorios en el directorio `repo`. Consulte la página del comando man `rsync`(1).

```
$ rsync -aP /mnt/repo/ /export/IPSpkgrepos/Solaris
```

- **Utilice el comando `tar`.**

  Con el comando `tar`, como se muestra en el siguiente ejemplo, se puede copiar el repositorio mucho más rápido del sistema de archivos montado al sistema de archivos ZFS del repositorio.

  ```
  $ cd /mnt/repo; tar cf - . | (cd /export/IPSpkgrepos/Solaris; tar xfp -)
  ```

5. **Desmonte el archivo de imagen.**

   Asegúrese de no estar aun en el directorio `/mnt`.

   ```
   $ umount /mnt
   ```

6. **Verifique el contenido del nuevo repositorio.**

   Utilice los subcomandos `info`, `list` y `verify` del comando `pkgrepo` para comprobar que el repositorio se haya copiado correctamente. Si el comando `pkgrepo verify` informa errores, intente usar el comando `pkgrepo fix` para corregir los errores. Consulte la página del comando man `pkgrepo`(1).

7. **Realice una instantánea del nuevo repositorio.**

   ```
   $ zfs snapshot rpool/export/IPSpkgrepos/Solaris@sol-11_2_0
   ```

Copia de un repositorio de Internet

En esta sección, se describe cómo hacer una copia local del repositorio de paquetes de la versión Oracle Solaris copiando el repositorio desde una ubicación de Internet. El primer procedimiento muestra la emisión del comando de copia de la línea de comandos. El segundo procedimiento muestra el uso de un servicio SMF para copiar y actualizar un repositorio automáticamente.

▼ Cómo copiar un repositorio de Internet de forma explícita

1. **Cree un sistema de archivos ZFS para el nuevo repositorio.**

   Cree el repositorio en una ubicación compartida. Establezca `atime` en `off` cuando cree el sistema de archivos de repositorio. Consulte “Mejores prácticas para la creación y el uso de repositorios de paquetes IPS locales” [10].

```
$ zfs create -o atime=off rpool/export/IPSpkgrepos
$ zfs create rpool/export/IPSpkgrepos/Solaris
$ zfs get atime rpool/export/IPSpkgrepos/Solaris
NAME                               PROPERTY  VALUE  SOURCE
rpool/export/IPSpkgrepos/Solaris  atime     off    inherited from rpool/export/IPSpkgrepos
```

2. **Cree la infraestructura de repositorio necesaria.**

   Cree la infraestructura adecuada para el repositorio `pkg`(5) a fin de poder copiar el repositorio. Los archivos de imagen utilizados en el método anterior incluyen la infraestructura de repositorio, de modo que este paso no es necesario. Cuando copia el contenido del repositorio mediante el comando `pkgrecv` como se describe en este método, debe crear la infraestructura del repositorio y, a continuación, copiar el contenido del repositorio en esa infraestructura. Consulte las páginas del comando man `pkg`(5) y `pkgrepo`(1)

   ```
   $ pkgrepo create /export/IPSpkgrepos/Solaris
   ```

3. **Copie el contenido del repositorio a la nueva ubicación.**

   Utilice el comando `pkgrecv` para copiar el repositorio. Esta operación podría afectar el rendimiento de la red. El tiempo necesario para completar esta operación depende del ancho de banda de la red y de la velocidad de conexión. Consulte también “Consideraciones de rendimiento para la copia de repositorios” [15]. Si actualiza este repositorio más adelante, se transferirán solamente los cambios, y es posible que el proceso demore mucho menos.

   El comando siguiente recupera todas las versiones de todos los paquetes del repositorio de paquetes especificado por la opción `-s` al repositorio especificado por la opción `-d`. Si va a copiar desde un sitio seguro, asegúrese de que el certificado SSL y la clave requeridos estén instalados, y especifique las opciones del certificado y la clave.

   ```
   $ pkgrecv -s https://pkg.oracle.com/solaris/support -d /export/IPSpkgrepos/Solaris \
   --key /path-to-ssl_key --cert /path-to-ssl_cert '*'
   ```

   Consulte la página del comando man `pkgrecv`(1) para obtener información sobre las opciones `-m` y `--clone`. No debe utilizar la opción `-m latest` para este fin. Usar un repositorio demasiado disperso puede producir errores cuando los usuarios intentan actualizar sus imágenes.

4. **Verifique el contenido del nuevo repositorio.**

   Utilice los subcomandos `info`, `list` y `verify` del comando `pkgrepo` para comprobar que el repositorio se haya copiado correctamente. Si el comando `pkgrepo verify` informa errores, intente usar el comando `pkgrepo fix` para corregir los errores. Consulte la página del comando man `pkgrepo`(1).

5. **Realice una instantánea del nuevo repositorio.**

   ```
   $ zfs snapshot rpool/export/IPSpkgrepos/Solaris@sol-11_2_0
   ```

▼ Cómo copiar un repositorio de Internet de forma automática

De manera predeterminada, el servicio SMF `svc:/application/pkg/mirror` realiza una operación de `pkgrecv` periódica desde los orígenes del editor `solaris` definidos en esta imagen como `/var/share/pkg/repositories/solaris`. Esta operación `pkgrecv` inicia a las 2:30 a. m. un día de cada mes. Para cambiar este comportamiento predeterminado, configure el servicio como se describe en este procedimiento.

Al final de cada ejecución correcta de este servicio, los catálogos de repositorio se actualizan. No es necesario actualizar el repositorio para crear un índice de búsqueda.

Dado que este servicio se ejecuta de forma periódica, el repositorio se crea y también se mantiene actualizado. No necesita utilizar las instrucciones de actualización manual del repositorio que se muestran en este documento.

Otros sistemas pueden establecer su origen del editor `solaris` para este repositorio actualizado automáticamente o a un clon de este repositorio. Sólo un sistema debe tener un origen de editor en Internet y ejecutar el servicio de `mirror` para recibir automáticamente las actualizaciones.

1. **Configure los orígenes del editor.**

   De manera predeterminada, el servicio `mirror` transfiere paquetes desde el editor de `solaris` configurado en la imagen con raíz en /. Si bien no es posible especificar directamente los orígenes del editor en la configuración del servicio `mirror`, puede configurar la raíz de la imagen desde la cual recuperar esta información. En la raíz de la imagen, utilice `pkg set-publisher` para configurar los orígenes del editor para usar como fuentes de la transferencia `pkgrecv` para el repositorio de reflejo.

   a. **(Opcional) Configure la raíz de la imagen.**

      Si la configuración del editor que desea utilizar para servicio de reflejo es diferente de la configuración del editor que desea utilizar en esta imagen, cree una imagen de usuario en una ubicación compartida (no incluida en ningún entorno de inicio) y restablezca el valor de propiedad `config/ref_image` en el servicio `mirror` a esa nueva imagen, como se muestra en el ejemplo siguiente. El servicio `mirror` utilizará la configuración del editor de la imagen `config/ref_image`.

      ```
      $ svccfg -s pkg/mirror:default setprop config/ref_image = /var/share/pkg/
      mirror_svc_ref_image
      $ pkg image-create /var/share/pkg/mirror_svc_ref_image
      ```

   b. **(Opcional) Defina los editores.**

      Si desea actualizar el repositorio de reflejo con paquetes de otros editores además del editor `solaris`, restablezca el valor de la propiedad `config/publishers` en el servicio

`mirror`, como se muestra en el siguiente ejemplo que muestra cómo agregar los editores `ha-cluster` y `solarisstudio`.

```
$ svccfg -s pkg/mirror:default setprop config/publishers = solaris,ha-
cluster,solarisstudio
```

c. Configure los orígenes del editor.

Dado que este servicio se ejecuta de forma periódica, debe definir los orígenes del editor en un repositorio que proporcione actualizaciones regulares. Para los productos Oracle, es probable que desee establecer los orígenes del editor para que un repositorio de asistencia recupere las actualizaciones de repositorio de asistencia (SRU). En el ejemplo siguiente, se necesita la opción `-R` sólo si está configurando los editores en una raíz de imagen alternativa. Es posible que las opciones `-k` y `-c` no sean necesarias, en función de las URI de origen.

```
$ pkg -R /var/share/pkg/mirror_svc_ref_image set-publisher \
-g https://pkg.oracle.com/solaris/support/ -k ssl_key -c ssl_cert solaris
$ pkg -R /var/share/pkg/mirror_svc_ref_image set-publisher \
-g https://pkg.oracle.com/ha-cluster/support/ -k ssl_key -c ssl_cert ha-cluster
$ pkg -R /var/share/pkg/mirror_svc_ref_image set-publisher \
-g https://pkg.oracle.com/solarisstudio/support/ -k ssl_key -c ssl_cert solarisstudio
```

Utilice uno de los siguientes comandos para verificar los editores configurados en la nueva imagen:

```
$ pkg -R /var/share/pkg/mirror_svc_ref_image publisher
$ pkg -R /var/share/pkg/mirror_svc_ref_image publisher solaris ha-cluster
 solarisstudio
```

2. (Opcional) Configure otras propiedades del servicio de reflejo.

Es posible que desee modificar otras propiedades del servicio `mirror`, como la hora en que se ejecuta el servicio o la ubicación del repositorio de reflejo.

Es posible que desee cambiar la hora en que se ejecuta el servicio para que coincida con más exactitud con la hora que espera que se actualicen los orígenes del editor reflejados. Para cambiar la hora en que se ejecuta el servicio, modifique el valor de la propiedad `config/crontab_period`.

Para cambiar la ubicación del repositorio de reflejo, modifique el valor de la propiedad `config/repository`. Si cambia la ubicación del repositorio de reflejo, mantenga el repositorio en una ubicación compartida. Consulte “Mejores prácticas para la creación y el uso de repositorios de paquetes IPS locales” [10]. La ubicación predeterminada, `/var/share/pkg/repositories/solaris`, es una ubicación compartida no incluida en ningún entorno existente.

3. Active el servicio de imagen.

Utilice el comando `svcs mirror` para comprobar el estado del servicio `mirror`.

- **El servicio está desactivado y desea utilizar este servicio.**

  a. **Actualice la instancia de servicio si cambió la configuración.**

     Si ha cambiado la configuración del servicio `mirror`, como se muestra en los comandos `svccfg setprop` en los pasos anteriores, actualice el servicio para confirmar los valores cambiados en la instantánea en ejecución. Si la salida del comando `svcprop -p config mirror` no muestra los valores que desea, asegúrese de que la salida del comando `svccfg -s mirror:default listprop config` muestre los valores que desee. Utilice `svcadm refresh mirror:default` o `svccfg -s mirror:default refresh` para confirmar los valores cambiados en la instantánea en ejecución del servicio. Utilice el comando `svcprop -p config mirror` nuevamente para confirmar que el servicio está configurado en la forma en que desea que esté configurado.

  b. **Active la instancia de servicio.**

     Utilice el siguiente comando para activar el servicio de reflejo:

     ```
     $ svcadm enable mirror:default
     ```

     Utilice el comando `svcs mirror` para confirmar que el servicio `mirror` está en línea. El servicio se ejecutará en la hora definida en la propiedad `config/crontab_period`.

- **El servicio está en línea y desea ejecutarlo ahora.**

  Si el servicio está en línea, actualícelo para ejecutarlo inmediatamente. Debe ver el método `svc-pkg-mirror` y el comando `pkgrecv` que ejecuta el usuario `pkg5srv`.

- **El servicio está en línea y no desea utilizarlo.**

  Utilice el comando `svcadm Disable mirror` para desactivar este servicio. Es posible que desee ejecutar este servicio en un solo sistema para mantener un repositorio maestro. En otros sistemas, es probable que desee desactivar este servicio.

- **El servicio está en mantenimiento o está degradado.**

  Utilice el comando `svcs -xvL mirror` para obtener más información para diagnosticar y solucionar el problema.

4. **Verifique el contenido del repositorio.**

   Una vez que el servicio `mirror` finaliza la ejecución, utilice los subcomandos `info`, `list` y `verify` del comando `pkgrepo` para comprobar que el repositorio se haya copiado o se haya actualizado correctamente. Si el comando `pkgrepo verify` informa errores, intente usar el comando `pkgrepo fix` para corregir los errores. Consulte la página del comando man `pkgrepo`(1).

Compruebe el valor de la propiedad `config/crontab_period` del servicio `mirror` para consultar cuándo se ejecutará el servicio. Mientras el servicio se está ejecutando, el comando `svcs -p mirror` muestra el estado del servicio como `online*` y muestra los procesos iniciados por este servicio. Espere hasta que el servicio de estado se muestra `online` y no haya ningún proceso asociado con el servicio antes de verificar el repositorio.

5. **Realice una instantánea del nuevo repositorio.**

```
$ zfs snapshot rpool/VARSHARE/pkg/repositories/solaris@sol-11_2_0
```

Pasos siguientes

Es posible que no desee copiar el contenido de varios editores al mismo tiempo. En vez de configurar varios editores en una propiedad `config/publishers`, puede crear varias instancias del servicio `pkg/mirror`. Por ejemplo, la propiedad `config/publishers` puede configurarse en `solaris` para la instancia `default`, en `ha-cluster` para una nueva instancia `pkg/mirror:ha-cluster` y en `solarisstudio` para una nueva instancia `pkg/mirror:solarisstudio`. Del mismo modo, `config/crontab_period` puede configurarse de forma distinta para cada instancia. Puede almacenar el contenido de cada editor en un repositorio, como se muestra en este procedimiento, o puede definir un valor `config/repository` diferente para cada instancia `pkg/mirror`.

Véase también

Consulte “Gestión de los servicios del sistema en Oracle Solaris 11.2 ” para obtener más información sobre los comandos SMF.

♦ ♦ ♦ CAPÍTULO 3

Cómo proporcionar acceso al repositorio

En este capítulo se describe cómo permitir a los clientes recuperar paquetes en su repositorio local mediante una interfaz de archivo o mediante una interfaz HTTP. Se puede configurar un repositorio para ambos tipos de acceso.

Permitir a los usuarios recuperar paquetes mediante una interfaz de archivo

En esta sección, se describe cómo servir los paquetes del repositorio local desde un directorio de la red local.

▼ Cómo permitir a los usuarios recuperar paquetes mediante una interfaz de archivo

1. **Configure un recurso compartido NFS.**

   Para permitir que los clientes accedan al repositorio local mediante NFS, cree y publique un recurso compartido NFS.

   ```
   $ zfs share -o share.nfs=on rpool/export/IPSpkgrepos%ipsrepo
   ```

   Consulte la página del comando man `zfs_share`(1M) para obtener más información, como las propiedades adicionales de `share.nfs` que puede definir.

2. **Confirme que el recurso compartido se publica.**

   Lleve a cabo una de las siguientes pruebas para confirmar la publicación del recurso compartido:

   - **Busque el repositorio en la tabla del sistema de archivos compartido.**

     ```
     $ grep repo /etc/dfs/sharetab
     /export/IPSpkgrepos     ipsrepo nfs     sec=sys,rw
     ```

- **Determine si se puede acceder al repositorio desde un sistema remoto.**

```
$ dfshares solaris
RESOURCE                                  SERVER ACCESS  TRANSPORT
   solaris:/export/IPSpkgrepos           solaris  -       -
```

3. Configure el origen del editor.

Para permitir que los sistemas cliente obtengan paquetes del repositorio de archivos local, configure el origen del editor.

a. Determine el nombre del editor.

Utilice el siguiente comando para determinar los nombres de los editores en el repositorio:

```
$ pkgrepo info -s /export/IPSpkgrepos/Solaris
PUBLISHER PACKAGES STATUS           UPDATED
solaris   4768     online           2014-04-02T18:11:55.640930Z
```

b. Compruebe la idoneidad del origen del editor.

Para actualizar paquetes instalados, instalar paquetes que dependan de paquetes instalados o instalar una zona no global, el repositorio que configura como el origen del editor debe contener al menos el mismo software que el que está instalado en la imagen en la que está configurando el editor. El repositorio también puede contener software más antiguo o más reciente, pero debe contener el mismo software que el que está instalado en el repositorio.

El siguiente comando muestra que el repositorio especificado no es un origen de editor adecuado para esta imagen:

```
$ pkg list entire
NAME (PUBLISHER)      VERSION                   IFO
entire                0.5.11-0.175.2.0.0.36.0   i--
$ pkgrepo list -Hs http://pkg.oracle.com/solaris/release
 entire@0.5.11-0.175.2.0.0.36.0
pkgrepo list: The following pattern(s) did not match any packages:
        entire@0.5.11-0.175.2.0.0.36.0
```

El siguiente comando muestra que el repositorio especificado es un origen de editor adecuado para esta imagen:

```
$ pkgrepo list -Hs /export/IPSpkgrepos/Solaris entire@0.5.11-0.175.2.0.0.36.0
solaris       entire       0.5.11,5.11-0.175.2.0.0.36.0:20140401T190148Z
```

c. Configure el origen de editor.

Mediante la ubicación del repositorio y nombre del editor de los pasos anteriores, ejecute el siguiente comando para definir el origen del editor:

```
$ pkg set-publisher -G '*' -M '*' -g /export/IPSpkgrepos/Solaris/ solaris
```

`-G '*'`	Elimina todos los orígenes existentes para el editor de `solaris`.
`-M '*'`	Elimina todos los reflejos existentes para el editor de `solaris`.
`-g`	Agrega el URI del repositorio local recién creado como el nuevo origen del editor `solaris`.

Consulte "Configuración de editores" de "Agregación y actualización de software en Oracle Solaris 11.2 " para obtener más información acerca de la configuración de editores.

Si restablece el origen del editor en otras imágenes, vuelva a realizar la prueba de idoneidad: otras imágenes pueden tener una versión diferente de software instalado y es posible que no puedan utilizar este repositorio. Si restablece el origen del editor en imágenes en otros sistemas, utilice una ruta de acceso completa para el argumento `-g`.

Cómo permitir a los usuarios recuperar paquetes mediante una interfaz HTTP

En esta sección se describe cómo servir los paquetes de repositorio locales mediante el servidor de repositorios de paquete.

▼ Cómo permitir a los usuarios recuperar paquetes mediante una interfaz HTTP

El servidor de depósitos de paquete, `pkg.depotd`, proporciona acceso a la red a los datos incluidos en un repositorio de paquetes. El servicio SMF `svc:/application/pkg/server` invoca el daemon `pkg.depotd`. Para permitir que los clientes accedan al repositorio local mediante HTTP, este procedimiento muestra cómo configurar el servicio `pkg/server`. Puede configurar la instancia `default` del servicio. Este procedimiento muestra cómo crear y configurar una nueva instancia.

1. **Cree una instancia del servidor de depósitos.**
   Utilice el subcomando `add` para agregar una nueva instancia del servicio `pkg/server` denominado `solaris`.

   ```
   $ svccfg -s pkg/server add solaris
   ```

2. **Configure la ruta al repositorio.**
   Configure la ruta en la que esta instancia del servicio puede encontrar los datos del repositorio.

```
$ svccfg -s pkg/server:solaris setprop pkg/inst_root=/export/IPSpkgrepos/Solaris
```

3. **(Opcional) Establezca el número de puerto.**

   Establezca el número de puerto en el que la instancia del servidor de depósitos debe escuchar solicitudes de paquetes entrantes. De manera predeterminada, `pkg.depotd` recibe las conexiones en el puerto 80. Para cambiar el puerto, restablezca la propiedad `pkg/port`.

   ```
   $ svccfg -s pkg/server:solaris setprop pkg/port=81
   ```

4. **(Opcional) Establezca otras propiedades.**

   Para obtener una lista completa de las propiedades de `pkg/server`, consulte la página del comando man `pkg.depotd`(1M)

   Para configurar varias propiedades del servicio, utilice el siguiente comando para editar todas las propiedades a la vez. Recuerde eliminar desde el principio el marcador de comentario (#) en las líneas que cambie.

   ```
   $ svccfg -s pkg/server:solaris editprop
   ```

5. **Inicie el servicio de repositorio.**

   Reinicie el servicio de servidor de depósitos de paquetes.

   ```
   $ svcadm refresh pkg/server:solaris
   $ svcadm enable pkg/server:solaris
   ```

6. **Pruebe que el servidor del repositorio esté funcionando.**

   Para determinar si el servidor del repositorio está funcionando, abra una ventana del explorador en la ubicación `localhost`. De manera predeterminada, `pkg.depotd` recibe las conexiones en el puerto 80. Si ha cambiado el puerto, abra una ventana del explorador en la ubicación `localhost:`*`número_puerto`*.

7. **Configure el origen del editor.**

   Para permitir que los sistemas cliente obtengan paquetes del repositorio de archivos local, configure el origen del editor.

   a. **Determine el nombre del editor.**

      Utilice el siguiente comando para determinar los nombres de los editores en el repositorio:

      ```
      $ pkgrepo info -s /export/IPSpkgrepos/Solaris
      PUBLISHER PACKAGES STATUS           UPDATED
      solaris   4768     online           2014-04-02T18:11:55.640930Z
      ```

   b. **Compruebe la idoneidad de este origen del editor.**

      Para actualizar paquetes instalados, instalar paquetes que dependan de paquetes instalados o instalar una zona no global, el repositorio que configura como el origen del editor debe

contener al menos el mismo software que el que está instalado en la imagen en la que está configurando el editor. El repositorio también puede contener software más antiguo o más reciente, pero debe contener el mismo software que el que está instalado en el repositorio.

El siguiente comando muestra que el repositorio especificado no es un origen de editor adecuado para esta imagen:

```
$ pkg list entire
NAME (PUBLISHER)      VERSION                    IFO
entire                0.5.11-0.175.2.0.0.36.0    i--
$ pkgrepo list -Hs http://pkg.oracle.com/solaris/release
 entire@0.5.11-0.175.2.0.0.36.0
pkgrepo list: The following pattern(s) did not match any packages:
        entire@0.5.11-0.175.2.0.0.36.0
```

El siguiente comando muestra que el repositorio especificado es un origen de editor adecuado para esta imagen:

```
$ pkgrepo list -Hs http://localhost:81/ entire@0.5.11-0.175.2.0.0.36.0
solaris       entire       0.5.11,5.11-0.175.2.0.0.36.0:20140401T190148Z
```

c. Configure el origen de editor.

Configure el origen del editor con uno de los siguientes valores:

- La ubicación de `pkg/inst_root`.

  ```
  $ pkg set-publisher -G '*' -M '*' -g /export/IPSpkgrepos/Solaris/ solaris
  ```

- La ubicación de `pkg/port`.

  ```
  $ pkg set-publisher -G '*' -M '*' -g http://localhost:81/ solaris
  ```

`-G '*'` Elimina todos los orígenes existentes para el editor de `solaris`.

`-M '*'` Elimina todos los reflejos existentes para el editor de `solaris`.

`-g` Agrega el URI del repositorio local recién creado como el nuevo origen del editor `solaris`.

Consulte "Configuración de editores" de "Agregación y actualización de software en Oracle Solaris 11.2 " para obtener más información acerca de la configuración de editores.

Si restablece el origen del editor en otras imágenes, vuelva a realizar la prueba de idoneidad: otras imágenes pueden tener una versión diferente de software instalado y es posible que no puedan utilizar este repositorio.

Véase también

- "Cómo servir varios repositorios con acceso al servidor web" [45] describe cómo servir varios repositorios desde varias ubicaciones o desde una única ubicación.

- “Varios repositorios en un mismo dominio” [54] describe cómo ejecutar varios repositorios con un solo nombre de dominio con prefijos diferentes.
- “Configuración del acceso HTTPS al repositorio” [56] describe cómo configurar el acceso seguro al repositorio.

♦ ♦ ♦ CAPÍTULO 4

Mantenimiento del repositorio de paquetes de IPS local

En este capítulo, se explica cómo actualizar los paquetes en un repositorio de IPS, cómo definir o actualizar las propiedades de un repositorio, y cómo agregar paquetes a un repositorio desde una segunda fuente.

Actualización del repositorio local

Los procedimientos que se muestran en esta sección ilustran las siguientes prácticas recomendadas para la actualización de repositorios de paquetes IPS:

- Mantenga los repositorios actualizados con todas las actualizaciones de compatibilidad para cada versión. Las actualizaciones de compatibilidad contienen actualizaciones de seguridad y otras correcciones importantes.
  - No intente seleccionar correcciones particulares para aplicar desde una actualización de compatibilidad. No agregue un subconjunto de paquetes desde una actualización de compatibilidad al repositorio. Agregue todo el contenido de la actualización de compatibilidad al repositorio local. El comportamiento predeterminado del comando `pkgrecv` es recuperar todas las versiones de todos los paquetes.
  - No omita una actualización de compatibilidad. Acumule todas las actualizaciones de compatibilidad aplicables en cada repositorio.

  Los usuarios pueden actualizar a una versión anterior a la versión más reciente en el repositorio mediante la especificación de la versión de `todo` el paquete de incorporación a instalar. Consulte Capítulo 4, “Actualización de una imagen de Oracle Solaris” de “Agregación y actualización de software en Oracle Solaris 11.2 ”.
- Actualice una copia del repositorio. Esta práctica ayuda a garantizar que los sistemas no accedan al repositorio mientras el repositorio se está actualizando. Cree una instantánea del repositorio antes de actualizarlo, clone la instantánea, realice la actualización y reemplace el repositorio original con el repositorio actualizado.

Si va a mantener varias copias de repositorios de paquetes con el mismo contenido, utilice el siguiente procedimiento para actualizar uno de los repositorios idénticos. Consulte

"Mantenimiento de varios repositorios locales idénticos" [37] para conocer el procedimiento para actualizar los repositorios adicionales desde este repositorio maestro.

▼ Cómo actualizar un repositorio de paquetes IPS local

Nota - No es necesario realizar este procedimiento si utiliza el `svc:/application/pkg/mirror` servicio SMF para actualizar periódicamente el repositorio. Consulte Cómo copiar un repositorio de Internet de forma automática [23] para obtener instrucciones sobre cómo utilizar el servicio de `reflejo`.

1. Realice una instantánea de ZFS del repositorio del paquete.

Asegúrese de que dispone de una instantánea actual del repositorio que se va a actualizar.

```
$ zfs list -t all -r rpool/export/IPSpkgrepos/Solaris
NAME                                          USED  AVAIL  REFER  MOUNTPOINT
rpool/export/IPSpkgrepos/Solaris             17.6G  78.4G    34K  /export/IPSpkgrepos/Solaris
rpool/export/IPSpkgrepos/Solaris@initial         0      -  17.6G  -
```

Si ya dispone de una instantánea del repositorio, utilice el comando `zfs diff` para comprobar si la instantánea es la misma que el conjunto de datos del repositorio.

```
$ zfs diff rpool/export/IPSpkgrepos/Solaris@initial
$
```

Si el comando `zfs diff` no produce ninguna salida, la instantánea es la misma que su conjunto de datos principal, y puede utilizarla instantánea para la actualización.

Si el comando `zfs diff` genera una salida, o si no dispone de una instantánea del repositorio, realice una nueva instantánea como se muestra en el paso 6 de Paso 6 in Cómo copiar un repositorio de Internet de forma explícita [21]. Utilice esta nueva instantánea para la actualización.

2. Realice una clonación ZFS del repositorio del paquete.

Clone la instantánea para crear una copia del repositorio que puede actualizar.

```
$ zfs clone rpool/export/IPSpkgrepos/Solaris@initial rpool/export/IPSpkgrepos/Solaris_tmp
$ zfs list -r rpool/export/IPSpkgrepos/Solaris/
NAME                                          USED  AVAIL  REFER  MOUNTPOINT
rpool/export/IPSpkgrepos/Solaris             17.6G  78.4G    34K  /export/IPSpkgrepos/Solaris
rpool/export/IPSpkgrepos/Solaris@initial         0      -  17.6G  -
rpool/export/IPSpkgrepos/Solaris_tmp           76K  78.4G  17.6G  /export/IPSpkgrepos/
Solaris_tmp
```

3. Actualice la clonación ZFS del repositorio del paquete.

De la misma manera que creó el repositorio original desde un archivo o desde una ubicación HTTP, puede actualizar un repositorio desde un archivo o desde una ubicación HTTP.

- **Actualice desde un archivo `zip`.**

  Consulte Ejemplo 2-2, "Agregación a un repositorio existente desde un archivo `zip`". Si el destino especificado ya contiene un repositorio de paquetes, el contenido del archivo `zip` se agrega al contenido del repositorio existente.

- **Actualice desde un archivo ISO.**

  a. **Monte la imagen ISO.**

  ```
  $ mount -F hsfs ./sol-11_2-incr-repo.iso /mnt
  ```

  b. **Copie el contenido del archivo ISO al clon del repositorio.**

  Utilice `rsync` o `tar` como se muestra en Cómo copiar un repositorio desde un archivo `iso` [20].

  ```
  $ rsync -aP /mnt/repo/ /export/IPSpkgrepos/Solaris_tmp
  ```

  c. **Desmonte la imagen ISO.**

- **Actualice desde un repositorio.**

  Copie el contenido desde otro repositorio al clon del repositorio. Si va a copiar desde un sitio seguro, asegúrese de que el certificado SSL y la clave requeridos estén instalados, y especifique las opciones del certificado y la clave.

  ```
  $ pkgrecv -s https://pkg.oracle.com/solaris/support \
  -d /export/IPSpkgrepos/Solaris_tmp \
  --key /path-to-ssl_key --cert /path-to-ssl_cert '*'
  ```

  Consulte la página del comando man `pkgrecv`(1) para obtener más información sobre el comando `pkgrecv`. Solamente se actualizan los paquetes que han cambiado, de modo que el tiempo para actualizar el repositorio puede ser mucho menor que el tiempo para rellenar el repositorio original. Consulte las sugerencias de rendimiento en "Consideraciones de rendimiento para la copia de repositorios" [15].

  Si la operación `pkgrecv` se interrumpe, siga las instrucciones en "Reanudación de una recepción de paquete interrumpida" [36].

4. **Sustituya el repositorio que está funcionando por el clon actualizado.**

```
$ svcadm disable -st pkg/server:solaris
$ zfs promote rpool/export/IPSpkgrepos/Solaris_tmp
$ zfs rename rpool/export/IPSpkgrepos/Solaris rpool/export/IPSpkgrepos/Solaris_old
$ zfs rename rpool/export/IPSpkgrepos/Solaris_tmp rpool/export/IPSpkgrepos/Solaris
```

Consulte la página del comando man svcadm(1M) para obtener más información sobre el comando svcadm.

5. **Verifique el repositorio actualizado.**

   Utilice el comando pkgrepo verify para verificar el repositorio actualizado. Consulte la página del comando man pkgrepo(1) para obtener más información sobre los comandos pkgrepo verify y pkgrepo fix.

6. **Catalogue los paquetes nuevos y actualice los índices de búsqueda.**

   Catalogue los paquetes nuevos encontrados en el repositorio recientemente actualizado y actualice todos los índices de búsqueda.

   ```
   $ pkgrepo refresh -s rpool/export/IPSpkgrepos/Solaris
   ```

7. **Realice una instantánea ZFS del clon recientemente actualizado del repositorio de paquetes.**

   ```
   $ zfs snapshot rpool/export/IPSpkgrepos/Solaris@S11U2SRU1
   ```

8. **Reinicie el servicio de la SMF.**

   Si está proporcionado el repositorio a través de una interfaz HTTP, reinicie el servicio SMF. Asegúrese de especificar la instancia de servicio correspondiente al reiniciar el servicio.

   ```
   $ svcadm restart pkg/server:solaris
   ```

9. **Elimine el repositorio antiguo.**

   Cuando crea que el repositorio actualizado funciona correctamente, puede eliminar el repositorio antiguo.

   ```
   $ zfs destroy rpool/export/IPSpkgrepos/Solaris_old
   ```

Reanudación de una recepción de paquete interrumpida

Si se interrumpe la operación pkgrecv, utilice la opción -c para recuperar el contenido que ya se haya descargado y reanude la descarga del contenido. El valor de *cache_dir* se proporciona en un mensaje informativo cuando se interrumpe la transferencia, como se muestra en el siguiente ejemplo:

```
PROCESS                    ITEMS       GET (MB)        SEND (MB)
...
pkgrecv: http protocol error: code: 503 reason: Service Unavailable
URL: 'https://pkg.oracle.com/solaris/support/file/file_hash
```

```
pkgrecv: Cached files were preserved in the following directory:
        /var/tmp/pkgrecv-fOGaIg
Use pkgrecv -c to resume the interrupted download.
$ pkgrecv -c /var/tmp/pkgrecv-fOGaIg \
-s https://pkg.oracle.com/solaris/support -d /export/IPSpkgrepos/Solaris_tmp \
--key /path/to/ssl_key --cert /path/to/ssl_cert '*'
Processing packages for publisher solaris ...
Retrieving and evaluating 156 package(s)...
```

Mantenimiento de varios repositorios locales idénticos

Es posible que desee mantener varias copias de repositorios de paquetes con el mismo contenido para cumplir los siguientes objetivos:

- Aumentar la disponibilidad del repositorio mediante el mantenimiento de copias en nodos distintos.
- Mejorar el rendimiento de los accesos al repositorio si tiene muchos usuarios o los usuarios se distribuyen en una gran distancia.

Utilice el procedimiento Cómo actualizar un repositorio de paquetes IPS local [34] para actualizar uno de los repositorios de paquetes. A continuación, utilice el procedimiento Cómo clonar un repositorio de paquetes IPS local [37] para actualizar repositorios idénticos adicionales desde el repositorio que actualizó en primer lugar. Estos dos procedimientos son muy similares, con una diferencia importante en el modo en que se utiliza el comando `pkgrecv`. La operación `pkgrecv` que aparece en el procedimiento de clonación copia los archivos de repositorio de origen exactamente, con los siguientes efectos:

- Los registros de hora de los catálogos de los repositorios clonados son exactamente iguales a los registros de hora de los catálogos del repositorio de origen. Si los repositorios tienen equilibrio de carga, los catálogos de todos los repositorios deben ser exactamente iguales para evitar problemas cuando el equilibrio de carga cambia clientes de un nodo a otro. Consulte “Configuración del equilibrio de carga” [54] para obtener información sobre el equilibrio de carga.
- Los paquetes que están en el repositorio de destino pero no en el repositorio de origen se eliminan del repositorio de destino. No utilice un repositorio disperso como origen para una operación de clonación a menos que su objetivo sea crear una copia exacta de ese repositorio disperso.

▼ Cómo clonar un repositorio de paquetes IPS local

Consulte Cómo actualizar un repositorio de paquetes IPS local [34] para obtener detalles de estos pasos.

1. **Copie el repositorio de destino.**

   Asegúrese de que dispone de instantáneas actual del repositorio de destino. Realice una clonación ZFS de esta instantánea.

2. **Actualice la copia del repositorio de destino.**

   Utilice el comando `pkgrecv` para clonar el repositorio de paquetes local actualizado previamente a la copia del repositorio de destino. Consulte la página del comando man `pkgrecv`(1) para obtener más información sobre la operación de clonación `pkgrecv`.

   ```
   $ pkgrecv -s /net/host1/export/IPSpkgrepos/Solaris \
   -d /net/host2/export/IPSpkgrepos/Solaris_tmp --clone
   ```

3. **Sustituya el repositorio de destino que está funcionando por el clon actualizado.**

4. **Verifique el repositorio actualizado.**

   Utilice el comando `pkgrepo verify` para verificar el repositorio de destino actualizado.

5. **Realice una instantánea del repositorio recientemente actualizado.**

6. **Reinicie el servicio de la SMF.**

   Si está proporcionado el repositorio a través de una interfaz HTTP, reinicie el servicio SMF. Asegúrese de especificar la instancia de servicio correspondiente al reiniciar el servicio.

7. **Elimine el repositorio antiguo.**

   Cuando crea que el repositorio actualizado funciona correctamente, elimine el repositorio antiguo.

Véase también Si está proporcionado el repositorio a través de una interfaz HTTP, consulte la siguiente documentación relacionada:

- “Cómo servir varios repositorios con acceso al servidor web” [45] describe cómo servir varios repositorios con varios daemons `pkg.depotd` que se ejecutan en puertos diferentes.
- “Varios repositorios en un mismo dominio” [54] describe cómo ejecutar varios repositorios con un solo nombre de dominio con prefijos diferentes.

Comprobación y definición de las propiedades del repositorio

En esta sección, se describe cómo mostrar información acerca de un repositorio IPS y cómo cambiar los valores de propiedades del repositorio.

Visualización de propiedades que se aplican a todo el repositorio

El siguiente comando muestra una lista con los editores de paquetes que el repositorio local conoce. La columna de ESTADO indica si los datos de paquetes del editor se están procesando.

```
$ pkgrepo info -s /export/IPSpkgrepos/Solaris
PUBLISHER PACKAGES STATUS           UPDATED
solaris   4506     online           2013-07-11T23:32:46.379726Z
```

El siguiente comando muestra información de propiedad que se aplica a todo el repositorio. Consulte la página del comando man pkgrepo(1) para obtener una lista completa de las propiedades de repositorio y sus descripciones, incluidas especificaciones de sus valores.

```
$ pkgrepo get -s /export/IPSpkgrepos/Solaris
SECTION    PROPERTY                    VALUE
publisher  prefix                      solaris
repository check-certificate-revocation False
repository signature-required-names    ()
repository trust-anchor-directory      /etc/certs/CA/
repository version                     4
```

`publisher/prefix`

El nombre del editor predeterminado. Si bien un repositorio puede contener paquetes de varios editores, sólo uno de los editores se puede establecer como el editor predeterminado. Se utiliza el nombre de este editor predeterminado para los siguientes fines:

- Para identificar un paquete cuando no hay ningún editor especificado en el paquete FMRI en el comando `pkg`
- Para asignar un editor a un paquete cuando el paquete se ha publicado en el repositorio (con el comando `pkgsend`(1)) y no hay ningún editor especificado en el manifiesto del paquete

`repository/check-certificate-revocation`

Un indicador para comprobar el certificado. Cuando se establece en `True`, el comando `pkgrepo verify` intenta determinar si el certificado se ha revocado desde que se emitió. Este valor debe coincidir con el valor de la propiedad de imagen `check-certificate-revocation` descrita en "Propiedades adicionales de imágenes" de "Agregación y actualización de software en Oracle Solaris 11.2 " y en la página del comando man pkg(1).

`repository/signature-required-names`

Una lista de nombres que se deben ver como nombres comunes de certificados durante la validación de firmas de un paquete. El comando `pkgrepo verify` utiliza esta lista. Este valor debe coincidir con el valor de la propiedad de imagen `signature-required-names` descrita en "Propiedades de imágenes para paquetes firmados" de "Agregación y actualización de software en Oracle Solaris 11.2 " y en la página del comando man pkg(1).

`repository/trust-anchor-directory`

La ruta absoluta del directorio que contiene los anclajes de confianza para los paquetes en este repositorio. El valor predeterminado es `/etc/certs/CA/`. Este valor debe coincidir con el valor de la propiedad de imagen `trust-anchor-directory` descrita en "Propiedades adicionales de imágenes" de "Agregación y actualización de software en Oracle Solaris 11.2 " y en la página del comando man `pkg`(1).

`repository/version`

La versión de formato del repositorio. No se puede definir este valor con el comando `pkgrepo set` que se muestra en "Modificación de valores de propiedad del repositorio" [42]. Este valor se puede definir con el comando `pkgrepo create`. Los repositorios de la versión 4 se crean de manera predeterminada. Los repositorios de la versión 4 admiten el almacenamiento de paquetes para varios editores.

Visualización de las propiedades del editor del repositorio

El siguiente comando muestra información sobre las propiedades del editor `solaris` en el repositorio local. Los paréntesis indican que un valor puede ser una lista de valores.

```
$ pkgrepo get -p solaris -s /export/IPSpkgrepos/Solaris
PUBLISHER SECTION    PROPERTY         VALUE
solaris   publisher  alias
solaris   publisher  prefix           solaris
solaris   repository collection-type  core
solaris   repository description      ""
solaris   repository legal-uris       ()
solaris   repository mirrors          ()
solaris   repository name             ""
solaris   repository origins          ()
solaris   repository refresh-seconds  ""
solaris   repository registration-uri ""
solaris   repository related-uris     ()
```

`publisher/prefix`

El nombre del editor especificado en la opción `-p`. Si no se especifica una opción `-p`, este valor es el nombre del editor predeterminado para este repositorio, como se describe en la sección anterior.

`repository/collection-type`

El tipo de paquetes en este repositorio. Si el valor es `core`, este repositorio contiene todas las dependencias declaradas por los paquetes en el repositorio. Si el valor es `supplemental`, este repositorio no contiene todas las dependencias declaradas por los paquetes en el repositorio.

`repository/description`

La finalidad y el contenido de este repositorio. Si este repositorio está disponible desde una interfaz HTTP, este valor se muestra en la sección Acerca de cerca de la parte superior de la página principal.

`repository/legal-uris`

Una lista de las ubicaciones de los documentos que proporcionan información legal sobre el repositorio.

`repository/mirrors`

Una lista de ubicaciones de repositorios que contienen el mismo contenido de paquete que este repositorio.

`repository/name`

El nombre de este repositorio. Si este repositorio está disponible desde una interfaz HTTP, este valor aparece en la parte superior de la página principal y en el título de la ventana.

`repository/origins`

Una lista de ubicaciones de repositorios que contienen el mismo contenido y metadatos de paquete que este repositorio.

`repository/refresh-seconds`

El número de segundos que los clientes deben esperar entre comprobaciones para que se actualicen los datos del paquete en este repositorio.

`repository/registration-uri`

La ubicación de un recurso que se debe utilizar para obtener credenciales para acceder al repositorio.

`repository/related-uris`

Una lista de ubicaciones de repositorios que contienen otros paquetes que pueden ser de interés.

El siguiente comando muestra información sobre la propiedad especificada *section/property* en el repositorio `pkg.oracle.com`.

```
$ pkgrepo get -p solaris -s http://pkg.oracle.com/solaris/release \
repository/name repository/description
PUBLISHER SECTION    PROPERTY    VALUE
solaris   repository description This\ repository\ serves\ the\ Oracle\ Solaris\ 11\ Package\
 repository.
solaris   repository name        Oracle\ Solaris\ 11\ Package\ Repository
```

Modificación de valores de propiedad del repositorio

"Visualización de las propiedades del editor del repositorio" [40] muestra que los valores de propiedad de la descripción del repositorio y del nombre del repositorio no se establecen para el editor de `solaris` en el repositorio local. Si este repositorio está disponible desde una interfaz HTTP y usted utiliza un explorador para ver el contenido de este repositorio, aparece un nombre predeterminado y ninguna descripción. Después de definir estos valores, el valor del editor `repository/name` se muestra cerca de la parte superior de la página como el título de la página y el valor del editor `repository/description` se muestra en la sección Acerca de, justo debajo del nombre. Debe utilizar la opción `-p` para especificar al menos un editor al definir estos valores. Si este repositorio contiene contenido de más de un editor, puede definir valores diferentes para cada editor, o puede especificar `-p all`.

```
$ pkgrepo set -p solaris -s /export/IPSpkgrepos/Solaris \
repository/description="Local copy of the Oracle Solaris 11 repository." \
repository/name="Oracle Solaris 11"
$ pkgrepo get -p solaris -s /export/IPSpkgrepos/Solaris repository/name repository/
description
PUBLISHER SECTION    PROPERTY         VALUE
solaris   repository description      Local\ copy\ of\ the\ Oracle\ Solaris\ 11\ repository.
solaris   repository name             Oracle\ Solaris\ 11
```

Personalización del repositorio local

Puede utilizar el comando `pkgrecv` para agregar paquetes y sus datos de editor al repositorio. Puede utilizar el comando `pkgrepo` para eliminar paquetes y editores desde el repositorio.

Agregación de paquetes al repositorio

Puede agregar editores a un repositorio. Por ejemplo, puede mantener paquetes `solaris`, `ha-cluster` y `solarisstudio` en un repositorio.

Si agrega paquetes personalizados, publique esos paquetes con un nombre de editor personalizado. No publique paquetes personalizados como un editor existente, por ejemplo, `solaris`. Si publica paquetes que no tienen un editor especificado, los paquetes se agregarán al editor predeterminado para el repositorio. Publique paquetes personalizados a un repositorio de prueba con el editor predeterminado. Luego, utilice el comando `pkgrecv` para agregar los paquetes y la información de su editor al repositorio de producción. Consulte "Publish the

Package" de "Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.2 " para obtener instrucciones.

En el siguiente ejemplo, los datos del editor de `isvpub` y todos los paquetes del archivo de paquetes `ISVproducts.p5p` se agregan al repositorio local. Un *archivo de paquetes* es un archivo que contiene información del editor y uno o más paquetes proporcionados por ese editor. Consulte "Deliver as a Package Archive File" de "Packaging and Delivering Software With the Image Packaging System in Oracle Solaris 11.2 ". La mayoría de las operaciones de `pkgrepo` no están disponibles para archivos de paquetes. Un archivo de paquetes contiene paquetes, pero no contiene configuración de repositorio. Sin embargo, los comandos `pkgrepo list` y `pkgrepo contents` funcionan con archivos de paquetes. El comando `pkgrepo contents` se analiza en "Examen de paquetes en el repositorio" [44].

En la salida `pkgrepo list`, el editor se muestra porque no es el editor mejor clasificado en el orden de búsqueda de esta imagen.

```
$ pkgrepo -s /tmp/ISVproducts.p5p list
PUBLISHER NAME                                    O VERSION
isvpub    isvtool                                   1.1,5.11:20131120T021902Z
isvpub    isvtool                                   1.0,5.11:20131120T010105Z
```

El siguiente comando `pkgrecv` recupera todos los paquetes del repositorio de origen. Si enumera nombres de paquetes para recuperar o especifica un patrón distinto de `'*'`, debe especificar la opción `-r` para asegurarse de recuperar todos los paquetes de dependencia necesarios.

```
$ pkgrecv -s /tmp/ISVproducts.p5p -d /export/IPSpkgrepos/Solaris '*'
Processing packages for publisher isvpub ...
Retrieving and evaluating 2 package(s)...
PROCESS        ITEMS      GET (MB)      SEND (MB)
Completed        2/2       0.0/0.0          0.0/0
```

Después de cambiar el contenido de un repositorio, actualice el repositorio y reinicie cualquier instancia de servicio de servidor de repositorio de paquetes configurada para este repositorio.

```
$ pkgrepo -s /export/IPSpkgrepos/Solaris refresh -p isvpub
Initiating repository refresh.
$ svcadm refresh pkg/server:solaris
$ svcadm restart pkg/server:solaris
```

El siguiente comando `pkgrepo info` muestra un paquete porque se han recuperado dos paquetes que son diferentes versiones del mismo paquete. El comando `pkgrepo list` muestra ambos paquetes.

```
$ pkgrepo -s /export/IPSpkgrepos/Solaris info
PUBLISHER PACKAGES STATUS           UPDATED
solaris   4768     online           2014-01-02T19:19:06.983979Z
isvpub    1        online           2014-03-20T23:24:37.196773Z
$ pkgrepo -s /export/IPSpkgrepos/Solaris list -p isvpub
PUBLISHER NAME                                    O VERSION
```

```
isvpub    isvtool                                  1.1,5.11:20131120T021902Z
isvpub    isvtool                                  1.0,5.11:20131120T010105Z
```

Agregue la nueva ubicación de repositorio para el editor `isvpub` mediante el comando `pkg set-publisher`.

Si este repositorio está disponible desde una interfaz HTTP y usted utiliza un explorador para ver el contenido de este repositorio, aparece este paquete nuevo especificando el editor en la ubicación. Por ejemplo, puede especificar `http://localhost:81/isvpub/`.

Examen de paquetes en el repositorio

Además de los comandos `pkgrepo info` y `pkgrepo list` que se muestran “Agregación de paquetes al repositorio” [42], puede utilizar el comando `pkgrepo contents` para examinar el contenido de paquetes en el repositorio.

Para un único paquete, la salida del comando `pkgrepo contents` es la misma que la salida del comando `pkg contents -m`. El comando `pkgrepo contents` muestra la salida para cada paquete coincidente en el repositorio especificado, mientras que el comando `pkg contents` muestra la salida sólo para versiones de paquetes coincidentes que se pueden instalar en esta imagen. Si especifica la opción `-t`, el comando `pkgrepo contents` sólo muestra las acciones especificadas.

En el ejemplo siguiente, no es necesario especificar la versión del paquete porque sólo existe una versión de este paquete en el repositorio especificado. Este paquete contiene acciones `depend` para proporcionar el conjunto de paquetes Oracle Solaris necesarios para la instalación y operación de Oracle Database 12.

```
$ pkgrepo -s http://pkg.oracle.com/solaris/release/ \
contents -t depend oracle-rdbms-server-12cR1-preinstall
depend fmri=x11/library/libxi type=group
depend fmri=x11/library/libxtst type=group
depend fmri=x11/session/xauth type=group
depend fmri=compress/unzip type=require
depend fmri=developer/assembler type=require
depend fmri=developer/build/make type=require
```

Eliminación de paquetes del repositorio

No elimine paquetes ofrecidos por un editor de Oracle. “Agregación y actualización de software en Oracle Solaris 11.2 ” muestra métodos para instalar sólo los paquetes que desee y evitar instalar los paquetes que no desee.

Puede utilizar el comando `pkgrepo remove` para eliminar los paquetes que no fueron entregados por un editor de Oracle. Puede utilizar el comando `pkgrepo remove-publisher` para eliminar un editor y todos los paquetes proporcionados por ese editor. Consulte la página del comando man `pkgrepo`(1) para obtener detalles. Estas operaciones deben realizarse en una copia del repositorio, como se describe en Cómo actualizar un repositorio de paquetes IPS local [34].

Cómo servir varios repositorios con acceso al servidor web

Los procedimientos de esta sección muestran cómo extender la información proporcionada en “Cómo permitir a los usuarios recuperar paquetes mediante una interfaz HTTP” [29] a fin de admitir el servicio de varios repositorios.

Los siguientes métodos son dos formas diferentes para servir varios repositorios de paquetes IPS mediante el acceso HTTP. Para ambos métodos, comience por la creación de instancias adicionales del servicio `pkg/server` con las rutas únicas de repositorio.

- Varias ubicaciones. Los usuarios acceden a cada repositorio mediante la visualización de páginas en ubicaciones independientes.
- Única ubicación. Los usuarios acceden a todos los repositorios desde una ubicación.

Además de proporcionar acceso a varios repositorios, recuerde que un solo repositorio puede proporcionar paquetes de varios editores, como se muestra en “Agregación de paquetes al repositorio” [42].

▼ Cómo servir varios repositorios desde ubicaciones distintas

En este ejemplo, el repositorio `SolarisStudio` existe además del repositorio `Solaris`. El repositorio `Solaris` es accesible desde `http://localhost/` con el puerto 81, como se especifica en la instancia `solaris` del servicio `pkg/server`. Consulte “Cómo permitir a los usuarios recuperar paquetes mediante una interfaz HTTP” [29].

1. **Cree una nueva instancia del servidor de repositorio.**

   Utilice el subcomando `add` del comando `svccfg` para agregar una nueva instancia del servicio `pkg/server`.

   ```
   $ svccfg -s pkg/server add studio
   ```

2. **Compruebe que haya agregado la nueva instancia.**

   ```
   $ svcs pkg/server
   STATE  STIME    FMRI
   ```

```
online 14:54:16 svc:/application/pkg/server:default
online 14:54:20 svc:/application/pkg/server:studio
online 14:54:20 svc:/application/pkg/server:solaris
```

3. **Establezca la ruta al repositorio.**

   Configure la ruta en la que esta instancia del servicio puede encontrar los datos del repositorio.

   ```
   $ svccfg -s pkg/server:studio setprop pkg/inst_root=/export/IPSpkgrepos/SolarisStudio
   ```

4. **(Opcional) Establezca el número de puerto para la nueva instancia.**

   ```
   $ svccfg -s pkg/server:studio setprop pkg/port=82
   ```

5. **(Opcional) Establezca la base del proxy Apache.**

   Consulte “Configuración de un proxy con prefijo simple” [53] para ver un ejemplo de la configuración de `pkg/proxy_base`.

6. **Defina el nombre y la descripción del repositorio.**

   Asegúrese de que el nombre y la descripción del repositorio estén configurados como se muestra en “Modificación de valores de propiedad del repositorio” [42].

7. **Inicie el servicio de repositorio.**

   Reinicie el servicio de servidor de depósitos de paquetes.

   ```
   $ svcadm refresh pkg/server:studio
   $ svcadm enable pkg/server:studio
   ```

8. **Pruebe que el servidor del repositorio esté funcionando.**

   Abra una ventana del explorador en la ubicación `http://localhost:82/`.

   Si no definió el número de puerto, el valor predeterminado es 80. Vea el repositorio en `http://localhost:80/` o `http://localhost/`.

   Si el número de puerto también está siendo usado por otra instancia de `pkg/server`, agregue el nombre del editor a la ubicación para ver los nuevos paquetes. Por ejemplo, vea el repositorio en `http://localhost:81/solarisstudio/`.

9. **Configure el origen del editor.**

   Configure el origen del editor con uno de los siguientes valores:

   - La ubicación de `pkg/inst_root`.

     ```
     $ pkg set-publisher -G '*' -M '*' -g /export/IPSpkgrepos/SolarisStudio/ \
     solarisstudio
     ```

   - La ubicación de `pkg/port`.

     ```
     $ pkg set-publisher -G '*' -M '*' -g http://localhost:82/ solarisstudio
     ```

Véase también Consulte "Varios repositorios en un mismo dominio" [54] para obtener información sobre cómo ejecutar varios repositorios con un nombre de dominio con prefijos diferentes como y `http://pkg.example.com/solaris` y `http://pkg.example.com/studio`.

▼ Cómo servir varios repositorios desde una ubicación única

Muchos de los pasos de este procedimiento son los mismos que los pasos del procedimiento anterior. Consulte el procedimiento anterior para obtener detalles.

1. **Cree una nueva instancia del servidor de repositorio.**

2. **Establezca la ruta al repositorio.**

   Cada instancia de `pkg/server` gestionada mediante una instancia `pkg/depot` particular debe tener un único valor `pkg/inst_root`.

3. **Compruebe la propiedad `readonly` para la nueva instancia.**

   El valor predeterminado de la propiedad `pkg/readonly` es `true`. Si este valor se ha cambiado, restablezca el valor a `true`.

   ```
   $ svcprop -p pkg/readonly pkg/server:studio
   true
   ```

4. **Establezca la propiedad `standalone` para la nueva instancia.**

   De manera predeterminada, el valor de la propiedad `pkg/standalone` es `true`. Cualquier instancia de `pkg/server` cuya propiedad `pkg/standalone` está establecida en `false` se puede servir desde la misma ubicación mediante una instancia del servicio `pkg/depot`.

   ```
   $ svccfg -s pkg/server:studio
   svc:/application/pkg/server:studio> setprop pkg/standalone=false
   svc:/application/pkg/server:studio> refresh
   svc:/application/pkg/server:studio> select solaris
   svc:/application/pkg/server:solaris> setprop pkg/standalone=false
   svc:/application/pkg/server:solaris> refresh
   svc:/application/pkg/server:solaris> exit
   $
   ```

   Asegúrese de que el valor de la propiedad `pkg/inst_root` sea único para cada instancia de `pkg/server` cuya propiedad `pkg/standalone` esté establecida en `false`.

5. **(Opcional) Establezca el número de puerto para la instancia de `pkg/depot`.**

   De manera predeterminada, el número de puerto del servicio `svc:/application/pkg/depot:default` es 80. Este número de puerto puede ser el mismo que el número de puerto

para cualquiera de las instancias de `pkg/server` que se gestionarán mediante esta instancia de `pkg/depot`. Para cambiar el número de puerto, establezca la propiedad `config/port` de `pkg/depot:default`.

6. **Reinicie la instancia de `pkg/depot`.**

```
$ svcadm refresh pkg/depot:default
$ svcadm restart pkg/depot:default
```

7. **Pruebe que el servidor del repositorio esté funcionando.**

Cuando los usuarios abren la ubicación `http://localhost:80/`, ven el repositorio `http://localhost/solaris` con el editor `solaris` y ven el repositorio `http://localhost/studio` enumerado con el editor `solarisstudio`.

Si un repositorio proporciona paquetes para varios editores, todos los editores se muestran en la lista. Por ejemplo, los usuarios pueden ver el repositorio `http://localhost/solaris` enumerado con los editores `solaris` y `isvpub`.

8. **Configure el origen del editor.**

Configure el origen del editor con uno de los siguientes valores:

- La ubicación de `pkg/inst_root` única.

```
$ pkg set-publisher -G '*' -M '*' -g /export/IPSpkgrepos/SolarisStudio/ \
solarisstudio
```

- La ubicación definida por el valor de `config/port` más el nombre de instancia `pkg/server`.

```
$ pkg set-publisher -G '*' -M '*' -g http://localhost:80/studio/ solarisstudio
```

Pasos siguientes

Si cambia el contenido de un repositorio gestionado por una instancia de `pkg/depot`, como se describe en “Actualización del repositorio local” [33] y “Personalización del repositorio local” [42], realice ambos pasos siguientes:

- Ejecute `pkgrepo refresh` en el repositorio.
- Ejecute `svcadm restart` en la instancia `pkg/depot`.

Puede crear instancias adicionales del servicio `pkg/depot` donde cada instancia aloja uno o más repositorios.

Para generar una configuración independiente en lugar de configurar las instancias de servicio `pkg/server` y `pkg/depot`, consulte la página del comando man `pkg.depot-config`(1M).independiente

♦ ♦ ♦ CAPÍTULO 5

Ejecución del servidor de depósitos detrás de un servidor web

La ejecución del servidor de depósitos detrás de una instancia del servidor web apache proporciona las siguientes ventajas:

- Permite alojar varios repositorios con un mismo nombre de dominio. El servidor de depósitos `pkg`(5) permite proporcionar acceso a un repositorio en la red local o en Internet con facilidad. Sin embargo, el servidor de depósitos no admite que se sirvan varios repositorios en un solo nombre de dominio o prefijos sofisticados. Para alojar varios repositorios en un solo nombre de dominio, ejecute el servidor de depósitos detrás de un proxy web.
- Mejora el rendimiento y la disponibilidad. La ejecución del servidor de depósitos detrás de un proxy web puede mejorar el rendimiento y la disponibilidad del servidor mediante la activación del equilibrio de la carga en varios depósitos y del almacenamiento del contenido.
- Permite proporcionar un servidor del repositorio seguro. Ejecute el servidor de depósitos detrás de una instancia de Apache activada para el protocolo de capa de conexión segura (SSL) que admita los certificados de cliente.

Configuración Apache para el servidor de depósitos

Los ejemplos de este capítulo utilizan el servidor web Apache como software de proxy. Active el servidor web Apache al activar el servicio `svc:/network/http:apache22`. Consulte la Documentación del servidor HTTP Apache versión 2.2 para obtener información adicional.

Debe poder aplicar los principios que se muestran en estos ejemplos para cualquier software de servidor proxy.

El sistema operativo Oracle Solaris 11.2 incluye el servidor web Apache en el paquete `web/server/apache-22`, que entrega un archivo `httpd.conf` básico en `/etc/apache2/2.2`. En general, puede utilizar el siguiente comando para buscar el archivo `httpd.conf`:

```
$ pkg search -Hl -o path ':file:path:*httpd.conf'
```

```
etc/apache2/2.2/httpd.conf
etc/apache2/2.2/original/httpd.conf
```

Configuración necesaria de Apache

Si ejecuta el servidor de depósitos de paquetes detrás de una instancia del servidor web Apache, incluya el siguiente valor en su archivo `httpd.conf` para no decodificar las barras diagonales codificadas:

```
AllowEncodedSlashes NoDecode
```

Los nombres de paquetes pueden contener barras diagonales codificadas para URL porque las barras diagonales se utilizan para expresar la jerárquica de los nombres de paquetes. Por ejemplo, el nombre de paquete `pkg://solaris/developer/build/make` se convierte en `http://pkg.oracle.com/solaris/release/manifest/0/developer%2Fbuild%2Fmake` para el servidor web. Para evitar que estas barras diagonales se interpreten como delimitadores de directorio, establezca que Apache no descodifique las barras codificadas `%2F`.

Omitir esta configuración puede provocar errores `404 Not Found` y afectar muy negativamente el rendimiento de la consulta.

Valores de configuración Apache genéricos recomendados

Los siguientes valores afectan el rendimiento y la seguridad.

Reduzca el tamaño sin cifrar de los metadatos.

Los clientes HTTP pueden indicar al servidor que aceptan datos comprimidos en una solicitud HTTP. La activación del filtro DEFLATE de Apache puede reducir drásticamente el tamaño en línea de los metadatos, como catálogos y manifiestos. Los metadatos como catálogos y manifiestos suelen comprimirse un 90%.

```
AddOutputFilterByType DEFLATE text/html application/javascript text/css text/plain
```

Permita más solicitudes canalizadas.

Aumente el valor `MaxKeepAliveRequests` para permitir que los clientes extraigan un mayor número de solicitudes canalizadas sin cerrar la conexión.

```
MaxKeepAliveRequests 10000
```

Establezca el tiempo de espera máximo para la respuesta.

El tiempo de espera del proxy determina cuánto tiempo Apache espera la respuesta del depósito en segundo plano. Para la mayoría de las operaciones, 30 segundos es suficiente.

Las búsquedas que arrojan una gran cantidad de resultados pueden llevar bastante más tiempo. Quizá desee asignar un valor de tiempo de espera mayor para dichas búsquedas.

```
ProxyTimeout 30
```

Desactive el proxy de reenvío.

Asegúrese de que el proxy de reenvío esté desactivado.

```
ProxyRequests Off
```

Configuración del almacenamiento en caché para el servidor de depósitos

Se requiere una mínima configuración para configurar el servidor de depósitos detrás de un proxy de almacenamiento en caché. Con la excepción del archivo de los atributos del catálogo (consulte “Consideraciones de caché para el archivo de atributos del catálogo” [52]) y resultados de búsqueda de repositorio (consulte “Consideraciones de caché para la búsqueda” [52]), todos los archivos servidos son únicos y, por lo tanto, el almacenamiento en caché indefinidamente es seguro si es necesario. Asimismo, todas las respuestas contienen los encabezados HTTP adecuados para garantizar que los archivos de la memoria caché no caduquen por error.

Consulte Caching Guide (Guía de almacenamiento en caché) de Apache para obtener más información sobre la configuración Apache como proxy de almacenamiento en caché.

Utilice la directiva `CacheRoot` para especificar el directorio que va a contener los archivos almacenados en caché. Asegúrese de que el directorio especificado se pueda escribir en el proceso de Apache. No se muestra ningún mensaje de error explícito si Apache no puede escribir en este directorio.

```
CacheRoot /tank/proxycache
```

Apache permite activar el almacenamiento en caché para directorios específicos. Quizá desee que su servidor del repositorio almacene en caché todo el contenido del servidor, como se muestra en la siguiente directiva.

```
CacheEnable disk /
```

Utilice la directiva `CacheMaxFileSize` para establecer el tamaño máximo de los archivos que se van a almacenar en caché. Puede que el valor predeterminado de 1 MB de Apache sea demasiado pequeño para la mayoría de los repositorios. La siguiente directiva establece 1 GB como tamaño máximo para el archivo almacenado en caché.

```
CacheMaxFileSize 1000000000
```

Ajustar la estructura del directorio de la memoria caché en disco para obtener el mejor rendimiento con el sistema de archivos subyacente. En un conjunto de datos ZFS, el

rendimiento se ve más afectado si hay varios niveles de directorios que si hay un gran número de archivos en un mismo directorio. Por lo tanto, configure un solo nivel de directorio con un gran número de archivos en cada directorio. Utilice las directivas `CacheDirLevels` y `CacheDirLength` para controlar la estructura de directorios. Establezca `CacheDirLevels` en 1. Establezca `CacheDirLength` en un valor que proporcione un buen equilibrio entre el número de directorios y el número de archivos por directorio. El valor de 2 establecido a continuación genera 4096 directorios. Consulte la documentación Disk-based Caching (Almacenamiento en caché basado en disco) para obtener más información.

```
CacheDirLevels 1
CacheDirLength 2
```

Consideraciones de caché para el archivo de atributos del catálogo

En el archivo de atributos del catálogo del repositorio (`catalog.attrs`) contiene el estado actual del catálogo del repositorio. Puede que el tamaño de este archivo justifique el almacenamiento en caché. Sin embargo, este archivo caduca si el catálogo del repositorio en segundo plano ha cambiado. Puede utilizar uno de los dos métodos siguientes para solucionar este problema.

- No almacene este archivo en caché. Esta solución también funciona mejor si el servidor de repositorios se ejecuta en un entorno de alto ancho de banda en el que el tráfico adicional no es una consideración importante. El siguiente archivo `httpd.conf` parcial muestra cómo especificar que no se almacene en caché el archivo `catalog.attrs`:

  ```
  <LocationMatch ".*/catalog.attrs">
          Header set Cache-Control no-cache
  </LocationMatch>
  ```

- Quite este archivo de la memoria caché cuando se actualiza el catálogo del repositorio en segundo plano.

Consideraciones de caché para la búsqueda

La búsqueda de un repositorio de paquetes genera respuestas personalizadas que se basan en la solicitud. Por lo tanto, los resultados de la búsqueda no son apropiados para almacenarlos en caché. El servidor de repositorios establece los encabezados HTTP apropiados a fin de asegurar que los resultados de búsqueda no caduquen en una memoria caché. Sin embargo, el ahorro de ancho de banda derivado del uso del almacenamiento en caché es reducido. El siguiente archivo `httpd.conf` parcial muestra cómo especificar que no se almacenen en caché los resultados de búsqueda.

```
<LocationMatch ".*/search/\d/.*">
        Header set Cache-Control no-cache
</LocationMatch>
```

Configuración de un proxy con prefijo simple

En este ejemplo, se muestra la configuración básica para un servidor de depósitos sin equilibro de carga. En este ejemplo, se conecta `http://pkg.example.com/myrepo` con `internal.example.com:10000`.

Consulte “Cómo servir varios repositorios con acceso al servidor web” [45] para obtener instrucciones sobre la definición de otras propiedades que necesita y que no están descritas en este ejemplo.

Configure el servidor de depósitos con una configuración `pkg/proxy_base` que mencione la URL en la que se puede acceder al servidor de depósitos. Utilice los comandos siguientes para establecer la configuración `pkg/proxy_base`:

```
$ svccfg -s pkg/server add repo
$ svccfg -s pkg/server:repo setprop pkg/proxy_base = astring: http://pkg.example.com/
myrepo
$ svcadm refresh pkg/server:repo
$ svcadm enable pkg/server:repo
```

El cliente `pkg`(5) abre 20 conexiones paralelas al servidor de depósitos cuando realiza operaciones de red. Asegúrese de que el número de subprocesos de depósitos coincida con las conexiones esperadas para el servidor en cualquier momento. Utilice los siguientes comandos para definir el número de subprocesos por depósito:

```
$ svccfg -s pkg/server:repo setprop pkg/threads = 200
$ svcadm refresh pkg/server:repo
$ svcadm restart pkg/server:repo
```

Utilice `nocanon` para suprimir la canonización de direcciones URL. Este valor es importante para que las búsquedas funcionen bien. Además, limite el número de conexiones en segundo plano al número de subprocesos que el servidor de depósitos proporciona. El siguiente archivo `httpd.conf` parcial muestra cómo aplicar un proxy en un servidor de depósitos:

```
Redirect /myrepo http://pkg.example.com/myrepo/
ProxyPass /myrepo/ http://internal.example.com:10000/ nocanon max=200
```

Para obtener información sobre el proxy SSL en el nivel del núcleo de Oracle Solaris y el uso de SSL para cifrar y acelerar las comunicaciones del servidor web, consulte Capítulo 3, “Servidores web y el protocolo de capa de sockets seguros” de “Protección de la red en Oracle Solaris 11.2 ”.

Varios repositorios en un mismo dominio

La principal razón para ejecutar el servidor de depósitos detrás de un proxy es que permite ejecutar varios repositorios en un mismo nombre de dominio con diferentes prefijos. El ejemplo de “Configuración de un proxy con prefijo simple” [53] puede ampliarse fácilmente para admitir varios repositorios.

En este ejemplo, tres prefijos diferentes de un nombre de dominio están conectados con tres repositorios de paquetes diferentes:

- `http://pkg.example.com/repo_one` está conectado con `internal.example.com:10000`
- `http://pkg.example.com/repo_two` está conectado con `internal.example.com:20000`
- `http://pkg.example.com/xyz/repo_three` está conectado con `internal.example.com:30000`

El servidor de depósitos `pkg`(5) es un servicio que se gestiona mediante SMF. Por lo tanto, para ejecutar varios servidores de depósitos en el mismo host, simplemente cree una nueva instancia de servicio:

```
$ svccfg -s pkg/server add repo1
$ svccfg -s pkg/server:repo1 setprop pkg/property=value
$ ...
```

Como en el ejemplo anterior, cada servidor de depósitos se ejecuta con 200 subprocesos.

```
Redirect /repo_one http://pkg.example.com/repo_one/
ProxyPass /repo_one/ http://internal.example.com:10000/ nocanon max=200

Redirect /repo_two http://pkg.example.com/repo_two/
ProxyPass /repo_two/ http://internal.example.com:20000/ nocanon max=200

Redirect /xyz/repo_three http://pkg.example.com/xyz/repo_three/
ProxyPass /xyz/repo_three/ http://internal.example.com:30000/ nocanon max=200
```

Configuración del equilibrio de carga

Quizá desee ejecutar servidores de depósitos detrás de un equilibrador de carga de Apache. Una ventaja del equilibrio de carga es aumentar la disponibilidad del repositorio. Esta sección muestra dos ejemplos del equilibrio de carga.

Si los repositorios tienen equilibrio de carga, los catálogos de todos los repositorios deben ser exactamente iguales para evitar problemas cuando el equilibrio de carga cambia clientes de un nodo a otro. Para asegurarse de que los catálogos sean iguales, clone los repositorios que participan en el equilibrio de carga como se describe en “Mantenimiento de varios repositorios locales idénticos” [37].

Un servidor del repositorio con equilibrio de carga

En este ejemplo, se conecta `http://pkg.example.com/myrepo` con `internal1.example.com:10000` and `internal2.example.com:10000`.

Configure el servidor de depósitos con una configuración de `proxy_base` adecuada, como se muestra en “Configuración de un proxy con prefijo simple” [53].

Limite el número de conexiones en segundo plano al número de subprocesos que cada depósito esté ejecutando dividido por el número de depósitos de la configuración del equilibrador de carga. De lo contrario, Apache abre más conexiones con un depósito que las que están disponibles, y las conexiones se detienen, lo cual puede reducir el rendimiento. Especifique el número máximo de conexiones paralelas a cada depósito con el parámetro `max=`. El siguiente ejemplo muestra dos depósitos que ejecutan 200 subprocesos cada uno. Consulte “Configuración de un proxy con prefijo simple” [53] para ver un ejemplo de cómo definir el número de subprocesos de los depósitos.

```
<Proxy balancer://pkg-example-com-myrepo>
        # depot on internal1
        BalancerMember http://internal1.example.com:10000 retry=5 max=100

        # depot on internal2
        BalancerMember http://internal2.example.com:10000 retry=5 max=100
</Proxy>

Redirect /myrepo http://pkg.example.com/myrepo/
ProxyPass /myrepo/ balancer://pkg-example-com-myrepo/ nocanon
```

Un servidor del repositorio con equilibrio de carga y otro sin equilibrio de carga

Este ejemplo incluye todas las directivas que debe agregar al archivo `httpd.conf` para un servidor del repositorio que aloja una configuración de servidor de depósito con carga equilibrada y con carga no equilibrada.

En este ejemplo, dos prefijos diferentes de un mismo nombre de dominio están conectados a tres repositorios de paquetes diferentes:

- `http://pkg.example.com/repo_one` está conectado con `internal1.example.com:10000` y `internal2.example.com:10000`
- `http://pkg.example.com/repo_two` está conectado con `internal1.example.com:20000`

```
AddOutputFilterByType DEFLATE text/html application/javascript text/css text/plain

AllowEncodedSlashes NoDecode
```

```
MaxKeepAliveRequests 10000

ProxyTimeout 30

ProxyRequests Off

<Proxy balancer://pkg-example-com-repo_one>
        # depot on internal1
        BalancerMember http://internal1.example.com:10000 retry=5 max=100

        # depot on internal2
        BalancerMember http://internal2.example.com:10000 retry=5 max=100
</Proxy>

Redirect /repo_one http://pkg.example.com/repo_one/
ProxyPass /repo_one/ balancer://pkg-example-com-repo_one/ nocanon
Redirect /repo_two http://pkg.example.com/repo_two/
ProxyPass /repo_two/ http://internal.example.com:20000/ nocanon max=200
```

Configuración del acceso HTTPS al repositorio

Cualquier cliente puede descargar paquetes desde un repositorio que está configurado para servir paquetes en HTTP. En algunos casos, debe restringir el acceso. Una forma para restringir el acceso al repositorio es ejecutar el servidor de depósitos detrás de una instancia de Apache activada para SSL que admita certificados de cliente.

El uso de SSL proporciona las siguientes ventajas:

- Asegura la transferencia cifrada de datos de paquete entre el cliente y el servidor
- Permite otorgar acceso a los repositorios sobre la base del certificado que el cliente presenta al servidor

Para configurar un servidor del repositorio seguro, debe crear una cadena de certificado personalizada:

1. Cree una autoridad de certificación (CA), que es el principio de la cadena de certificados.
2. Emita certificados desde esta CA para los clientes que pueden acceder a este repositorio.

Se almacena una copia del CA en el servidor del repositorio. Siempre que un cliente presenta un certificado al servidor, se coteja el certificado de cliente con la CA en el servidor para determinar si otorgar acceso o no.

En esta sección se describen los siguientes pasos para crear la cadena de certificados y configurar el front-end de Apache para verificar los certificados de cliente:

- Crear un almacén de claves
- Creación de una autoridad de certificación para certificados de cliente

- Agregación de configuración de SSL al archivo de configuración de Apache
- Creación de una autoridad de certificación de servidor autofirmado
- Creación de un almacén de claves PKCS12

Para obtener información sobre los privilegios del servidor web Apache en Oracle Solaris, consulte "Bloqueo de recursos utilizando privilegios ampliados" de "Protección de los usuarios y los procesos en Oracle Solaris 11.2 ".

Creación de un almacén de claves

Para gestionar certificados y claves, cree un almacén de claves. El almacén de claves se almacena en la CA, la clave de CA y certificados y claves de cliente.

La herramienta utilizada para la gestión del almacén de claves es `pktool`. Consulte la página del comando man `pktool`(1) para obtener más información.

La ubicación predeterminada del almacén de claves para `pktool` es `/var/user/`*username*, donde *username* es el nombre del usuario del sistema actual. La ubicación predeterminada de este almacén de claves puede causar problemas cuando un almacén de claves es gestionado por varios usuarios. Además, la gestión del repositorio de paquetes IPS debe tener un almacén de claves dedicado para no confundir los certificados. Para definir una ubicación personalizada para el almacén de claves `pktool` para el repositorio de paquetes IPS, defina la variable de entorno `SOFTTOKEN_DIR`. Restablezca la variable `SOFTTOKEN_DIR` según sea necesario para gestionar varios almacenes de claves.

Utilice los siguientes comandos para crear un directorio para el almacén de claves. Establezca el propietario, el grupo y los permisos de forma adecuada si varios usuarios necesitan gestionar el almacén de claves.

```
$ mkdir /path-to-keystore
$ export SOFTTOKEN_DIR=/path-to-keystore
```

El acceso al almacén de claves está protegido por una frase de contraseña que se debe introducir cada vez que se invoca el comando `pktool`. La frase de contraseña predeterminada para un almacén de claves que se acaba de crear es `changeme`. Asegúrese de cambiar la frase de contraseña `changeme` por una frase de contraseña más segura.

Utilice el siguiente comando para establecer la frase de contraseña (PIN) para el almacén de claves:

```
$ pktool setpin
Enter token passphrase: changeme
Create new passphrase:
Re-enter new passphrase:
Passphrase changed.
```

```
$ ls /path-to-keystore
pkcs11_softtoken
```

Creación de una autoridad de certificación para certificados de cliente

La CA es el certificado de nivel superior en la cadena de certificados. La CA es necesaria para generar certificados de cliente y validar los certificados presentados por los clientes para acceder al repositorio.

Las CA de terceros están gestionadas por pocas empresas de confianza, como VeriSign. Esta gestión de confianza permite a los clientes verificar la identidad de un servidor con una de sus CA. El ejemplo de esta sección no incluye verificar la identidad del servidor del repositorio. Este ejemplo sólo muestra verificación de los certificados de cliente. Por lo tanto, en este ejemplo se utiliza un certificado autofirmado para crear la CA y no se utiliza ninguna CA de terceros.

La CA requiere un nombre común (CN). Si ejecuta un solo repositorio, puede que desee definir el CN con el nombre de su organización (por ejemplo, "Entrega de software de Oracle"). Si tiene varios repositorios, cada repositorio debe tener su propia CA. En este caso, configure el CN con un nombre que identifique de forma única el repositorio para el que crea la CA. Por ejemplo, si tiene un repositorio de la versión y un repositorio de asistencia, sólo los certificados de la CA de la versión podrán acceder al repositorio de la versión, y sólo los certificados de la CA de asistencia podrán acceder al repositorio de asistencia.

Para identificar el certificado en el almacén de claves, defina una etiqueta descriptiva para el certificado. Una buena práctica es definir la etiqueta del certificado en *CN*`_ca`, donde *CN* es el CN del certificado.

Utilice el siguiente comando para crear el certificado de CA, donde *name* es el CN del certificado y *CAlabel* es la etiqueta del certificado:

```
$ pktool gencert label=CAlabel subject="CN=name" serial=0x01
```

La CA se almacenará en el almacén de claves. Utilice el siguiente comando para mostrar el contenido de su almacén de claves:

```
$ pktool list
```

Deberá extraer el certificado de CA del almacén de claves cuando configure Apache como se describe en “Agregación de configuración de SSL al archivo de configuración de Apache” [61]. Utilice el siguiente comando para extraer el certificado de CA a un archivo denominado `ca_file.pem`:

```
$ pktool export objtype=cert label=CAlabel outformat=pem \
outfile=ca_file.pem
```

Creación de certificados de cliente utilizados para acceder al repositorio

Después de generar la CA, puede generar certificados de cliente.

Generación de una solicitud de firma de certificado

Para generar un certificado de cliente, genere una solicitud de firma de certificado (CSR). La CSR contiene toda la información necesaria para pasar de forma segura al servidor.

Si sólo desea comprobar si el cliente posee un certificado válido emitido por usted, no necesita volver a codificar ninguna información. Cuando el cliente presenta su certificado al servidor, el servidor valida el certificado con la CA y verifica si el certificado de cliente fue generado por usted. Sin embargo, SSL requiere un `subject` para la CSR. Si no necesita transferir ninguna otra información al servidor, puede configurar `subject` con el país en el que se ha emitido el certificado. Por ejemplo, podría establecer el asunto `subject` como `C=US`.

Una buena práctica es codificar el nombre de usuario del cliente en el certificado para permitir al servidor identificar el cliente. El nombre de usuario es el nombre del usuario al que se concede acceso al repositorio. Puede utilizar el CN para este fin. Especifique una etiqueta para esta CSR para poder buscar y extraer la clave para el certificado final como se describe en “Extracción de la clave de certificado” [60].

Utilice el siguiente comando para generar la CSR:

```
$ pktool gencsr subject="C=US,CN=username" label=label format=pem \
outcsr=cert.csr
```

Utilice el siguiente comando OpenSSL para inspeccionar la CSR en el archivo `cert.csr`.

```
$ openssl req -text -in cert.csr
```

Firma de la CSR

La CSR debe estar firmada por la CA para crear un certificado. Para firmar la CSR, proporcione la siguiente información:

- Defina el `issuer` del certificado con la misma cadena que utilizó para el `subject` cuando creó la CA con el comando `gencert`, como se muestra en “Creación de una autoridad de certificación para certificados de cliente” [58].
- Defina un número de serie hexadecimal. En este ejemplo, el número de serie de la CA se ha especificado como `0x01`, de modo que al certificado de cliente se le debe asignar el número

de serie `0x02`. Incremente el número de serie para cada nuevo certificado de cliente que genere.

Cada CA y sus certificados de cliente descendientes tienen su propio conjunto de números de serie. Si tiene varias CA configuradas en el almacén de claves, tenga cuidado de definir correctamente los números de serie del certificado de cliente.

- Defina el `signkey` con la etiqueta de la CA en el almacén de claves.
- Defina `outcert` con el nombre del archivo de certificado. Una buena práctica es nombrar el certificado y la clave como el repositorio al que se va a acceder.

Utilice el siguiente comando para firmar la CSR:

```
$ pktool signcsr signkey=CAlabel csr=cert.csr  \
serial=0x02 outcert=reponame.crt.pem issuer="CN=name"
```

El certificado se crea en el archivo *reponame*`.crt.pem`. Utilice el siguiente comando OpenSSL para inspeccionar el certificado:

```
$ openssl x509 -text -in reponame.crt.pem
```

Extracción de la clave de certificado

Extraiga la clave para este certificado del almacén de claves. Establezca `label` con el mismo valor de la etiqueta al ejecutar `gencsr` para generar la CSR en “Generación de una solicitud de firma de certificado” [59]. Utilice el siguiente comando para exportar la clave del almacén de claves:

```
$ pktool export objtype=key label=label outformat=pem \
outfile=reponame.key.pem
```

Transfiera el certificado y la clave a los sistemas cliente que necesitan acceder al repositorio protegido por SSL.

Activación de los sistemas cliente para acceder al repositorio protegido

Para acceder al repositorio protegido por SSL, los sistemas cliente deben tener una copia del certificado y la clave y deben especificar el certificado y la clave en la configuración del editor.

Copie el certificado (*reponame*`.crt.pem`) y la clave (*reponame*`.key.pem`) para cada sistema cliente. Por ejemplo, puede copiarlos en el directorio `/var/pkg/ssl` en cada cliente.

Utilice el siguiente comando para especificar la clave y el certificado generados en la configuración del editor:

```
$ pkg set-publisher -k reponame.key.pem -c reponame.crt.pem \
-p https://repolocation
```

Tenga en cuenta que la autenticación SSL sólo se admite para las URI del repositorio HTTPS. No se admite la autenticación SSL para las URI del repositorio de archivo.

Agregación de configuración de SSL al archivo de configuración de Apache

Para utilizar la autenticación basada en un certificado de cliente para su repositorio, defina primero una configuración Apache para el servidor de depósitos genérica como se describe en “Configuración Apache para el servidor de depósitos” [49]. Luego, agregue la siguiente configuración de SSL al final de su archivo `httpd.conf`:

```
# Let Apache listen on the standard HTTPS port
Listen 443

# VirtualHost configuration for request on port 443
<VirtualHost 0.0.0.0:443>
        # DNS domain name of the server, needs to match your server certificate
        ServerName pkg-sec.example.com

        # enable SSL
        SSLEngine On

        # Location of the server certificate and key.
        # You either have to get one from a certificate signing authority like
        # VeriSign or create your own CA for testing purposes (see "Creating a
        # Self-Signed CA for Testing Purposes")
        SSLCertificateFile /path/to/server.crt
        SSLCertificateKeyFile /path/to/server.key

        # Intermediate CA certificate file. Required if your server certificate
        # is not signed by a top-level CA directly but an intermediate authority
        # Comment out this section if you are using a test certificate or your
        # server certificate doesn't require it.
        # For more info:
        # http://httpd.apache.org/docs/2.2/mod/mod_ssl.html#sslcertificatechainfile
        SSLCertificateChainFile /path/to/ca_intermediate.pem

        # CA certs for client verification.
        # This is where the CA certificate created in step 3 needs to go.
        # If you have multiple CAs for multiple repos, just concatenate the
        # CA certificate files
        SSLCACertificateFile /path/to/ca_cert.pem

        # If the client presents a certificate, verify it here. If it doesn't,
        # ignore.
        # This is required to be able to use client-certificate based and
```

```
        # anonymous SSL traffic on the same VirtualHost.
        # This statement could also go into the <Location> tags but putting it
        # here avoids re-negotiation which can cause security issues with older
        # servers/clients:
        # http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2009-3555
        SSLVerifyClient optional

        <Location /repo>
                SSLVerifyDepth 1
                # This is the SSL requirement for this location.
                # Requirements can be made based on various information encoded
                # in the certificate. Two variants are the most useful for use
                # with IPS repositories:
                # a) SSLRequire ( %{SSL_CLIENT_I_DN_CN} =~ m/reponame/ )
                #    only allow access if the CN in the client certificate matches
                #    "reponame", useful for different certificates for different
                #    repos
                #
                # b) SSLRequire ( %{SSL_CLIENT_VERIFY} eq "SUCCESS" )
                #    grant access if clients certificate is signed by one of the
                #    CAs specified in SSLCACertificateFile
                SSLRequire ( %{SSL_CLIENT_VERIFY} eq "SUCCESS" )

                # proxy request to depot running at internal.example.com:12345
                ProxyPass http://internal.example.com:12345 nocanon max=500
        </Location>
</VirtualHost>
```

Creación de una autoridad de certificación de servidor autofirmado

Con fines de prueba, puede utilizar una autoridad de certificación de un servidor autofirmado (CA) en lugar de una CA de terceros. Los pasos para crear un servidor autofirmado CA para Apache son muy similares a los pasos para crear una CA para los certificados del cliente como se describe en “Creación de una autoridad de certificación para certificados de cliente” [58].

Utilice el siguiente comando para crear una CA del servidor. Configure `subject` con el nombre del servidor DNS.

```
$ pktool gencert label=apacheCA subject="CN=apachetest" \
serial=0x01
```

Utilice el comando siguiente para crear una CSR para una CA de servidor. Si el servidor es accesible con varios nombres o desea que esté disponible en su dirección IP directamente, utilice la directiva `subjectAltNames` como se describe en Subject Alternative Name en la documentación de OpenSSL.

```
$ pktool gencsr label=apache subject="CN=pkg-sec.internal.example.com" \
altname="IP=192.168.1.1,DNS=pkg-sec.internal.example.com" \
```

```
format=pem outcsr=apache.csr
```

Utilice el siguiente comando para firmar la CSR. Utilice `server.crt` para `SSLCertificateFile`.

```
$ pktool signcsr signkey=apacheCA csr=apache.csr serial=0x02 \
outcert=server.crt issuer="CN=apachetest"
```

Utilice el siguiente comando para extraer la clave. Utilice `server.key` para `SSLCertificateKeyFile`.

```
$ pktool export objtype=key label=apache outformat=pem \
outfile=server.key
```

Para asegurarse de que su cliente acepte esta clave de servicio, agregue el certificado de CA (`apacheCA`) al directorio de CA aceptadas en el sistema del cliente y reinicie el servicio `ca-certificates` para crear los enlaces necesarios para OpenSSL.

Utilice el siguiente comando para extraer el certificado de CA:

```
$ pktool export label=apacheCA objtype=cert outformat=pem \
outfile=test_server_ca.pem
```

Copie el certificado de CA en el directorio del certificado de CA de la máquina cliente:

```
$ cp /path-to/test_server_ca.pem /etc/certs/CA/
```

Reinicie el servicio de certificados de CA:

```
$ svcadm refresh ca-certificates
```

Antes de continuar, asegúrese de que el nuevo certificado de CA se ha vinculado. Después de actualizar, el servicio `ca-certificate` vuelve a crear los enlaces en el directorio `/etc/openssl/certs`. Ejecute el siguiente comando para comprobar si el nueva certificado de CA se ha vinculado:

```
$ ls -l /etc/openssl/certs | grep test_server_ca.pem
lrwxrwxrwx   1 root     root          40 May  1 09:51 e89d96e0.0 -> ../../certs/CA/
test_server_ca.pem
```

Puede que el valor hash, `e89d96e0.0`, sea diferente para usted porque está basado en el asunto de su certificado.

Creación de un almacén de claves PKCS12 para acceder a un repositorio seguro con Firefox

Los certificados PEM creados en “Creación de certificados de cliente utilizados para acceder al repositorio” [59] funcionarán para acceder al repositorio seguro con el cliente `pkg`.

Sin embargo, para acceder a la interfaz de usuario del explorador (BUI), debe convertir el certificado y la clave a un formato que Firefox pueda importar. Firefox acepta almacenes de claves PKCS12.

Utilice el siguiente comando OpenSSL para crear el almacén de claves PKCS12 para Firefox:

```
$ openssl pkcs12 -export -in /path-to/certificate.pem \
-inkey /path-to/key.pem -out name.p12
```

Para importar el almacén de claves PKCS12 que acaba de crear, seleccione los siguientes menús y botones de Firefox: Editar > Preferencias > Avanzado > Cifrado > Ver certificados > Autoridades > Importar.

Importe un certificado a la vez.

Ejemplo completo de repositorios seguros

En este ejemplo, se configuran tres repositorios seguros denominados `repo1`, `repo2` y `repo3`. Los repositorios `repo1` y `repo2` están configurados con certificados dedicados. Por lo tanto, los certificados para `repo1` no funcionarán en `repo2` y los certificados para `repo2` no funcionarán en `repo1`. El repositorio `repo3` está configurado para aceptar cualquier certificado.

El ejemplo supone que usted dispone de un certificado de servidor adecuado para la instancia Apache ya disponible. Si no tiene un certificado de servidor para la instancia Apache, consulte las instrucciones para crear un certificado de prueba en “Creación de una autoridad de certificación de servidor autofirmado” [62].

Los tres repositorios se configuran en `https://pkg-sec.example.com/repo1`, `https://pkg-sec.example.com/repo2` y `https://pkg-sec.example.com/repo3`. Estos repositorios apuntan a servidores de depósitos configurados en `http://internal.example.com` en puertos 10001, 10002 y 10003 respectivamente. Asegúrese de que la variable del entorno `SOFTTOKEN_DIR` esté definida correctamente como se describe en “Creación de un almacén de claves” [57].

▼ Cómo configurar repositorios seguros

1. **Cree un certificado de CA para `repo1`.**

```
$ pktool gencert label=repo1_ca subject="CN=repo1" serial=0x01
$ pktool export objtype=cert label=repo1_ca outformat=pem \
outfile=repo1_ca.pem
```

2. **Cree un certificado de CA para `repo2`.**

```
$ pktool gencert label=repo2_ca subject="CN=repo2" serial=0x01
$ pktool export objtype=cert label=repo2_ca outformat=pem \
outfile=repo2_ca.pem
```

3. **Cree un archivo de certificado CA combinado.**

```
$ cat repo1_ca.pem > repo_cas.pem
$ cat repo2_ca.pem >> repo_cas.pem
$ cp repo_cas.pem /path-to-certs
```

4. **Cree un par de certificado de cliente/clave para permitir que el usuario `myuser` acceda al repositorio `repo1`.**

```
$ pktool gencsr subject="C=US,CN=myuser" label=repo1_0001 format=pem \
outcsr=repo1_myuser.csr
$ pktool signcsr signkey=repo1_ca csr=repo1_myuser.csr  \
serial=0x02 outcert=repo1_myuser.crt.pem issuer="CN=repo1"
$ pktool export objtype=key label=repo1_0001 outformat=pem \
outfile=repo1_myuser.key.pem
$ cp repo1_myuser.key.pem /path-to-certs
$ cp repo1_myuser.crt.pem /path-to-certs
```

5. **Cree un par de certificado de cliente/clave para permitir que el usuario `myuser` acceda al repositorio `repo2`.**

```
$ pktool gencsr subject="C=US,CN=myuser" label=repo2_0001 format=pem \
outcsr=repo2_myuser.csr
$ pktool signcsr signkey=repo2_ca csr=repo2_myuser.csr  \
serial=0x02 outcert=repo2_myuser.crt.pem issuer="CN=repo2"
$ pktool export objtype=key label=repo2_0001 outformat=pem \
outfile=repo2_myuser.key.pem
$ cp repo2_myuser.key.pem /path-to-certs
$ cp repo2_myuser.crt.pem /path-to-certs
```

6. **Configure Apache.**

   Agregue la siguiente configuración de SSL al final de su archivo `httpd.conf`:

```
# Let Apache listen on the standard HTTPS port
Listen 443

<VirtualHost 0.0.0.0:443>
        # DNS domain name of the server
        ServerName pkg-sec.example.com

        # enable SSL
        SSLEngine On

        # Location of the server certificate and key.
        # You either have to get one from a certificate signing authority like
        # VeriSign or create your own CA for testing purposes (see "Creating a
        # Self-Signed CA for Testing Purposes")
        SSLCertificateFile /path/to/server.crt
        SSLCertificateKeyFile /path/to/server.key

        # Intermediate CA certificate file. Required if your server certificate
        # is not signed by a top-level CA directly but an intermediate authority.
```

```
        # Comment out this section if you don't need one or if you are using a
        # test certificate
        SSLCertificateChainFile /path/to/ca_intermediate.pem

        # CA certs for client verification.
        # This is where the CA certificate created in step 3 needs to go.
        # If you have multiple CAs for multiple repos, just concatenate the
        # CA certificate files
        SSLCACertificateFile /path/to/certs/repo_cas.pem

        # If the client presents a certificate, verify it here. If it doesn't,
        # ignore.
        # This is required to be able to use client-certificate based and
        # anonymous SSL traffic on the same VirtualHost.
        SSLVerifyClient optional

        <Location /repo1>
                SSLVerifyDepth 1
                SSLRequire ( %{SSL_CLIENT_I_DN_CN} =~ m/repo1/ )
                # proxy request to depot running at internal.example.com:10001
                ProxyPass http://internal.example.com:10001 nocanon max=500
        </Location>

        <Location /repo2>
                SSLVerifyDepth 1
                SSLRequire ( %{SSL_CLIENT_I_DN_CN} =~ m/repo2/ )
                # proxy request to depot running at internal.example.com:10002
                ProxyPass http://internal.example.com:10002 nocanon max=500
        </Location>

        <Location /repo3>
                SSLVerifyDepth 1
                SSLRequire ( %{SSL_CLIENT_VERIFY} eq "SUCCESS" )
                # proxy request to depot running at internal.example.com:10003
                ProxyPass http://internal.example.com:10003 nocanon max=500
        </Location>

</VirtualHost>
```

7. **Pruebe el acceso a `repo1`.**

```
$ pkg set-publisher -k /path-to-certs/repo1_myuser.key.pem \
-c /path-to-certs/repo1_myuser.crt.pem \
-p https://pkg-sec.example.com/repo1/
```

8. **Pruebe el acceso a `repo2`.**

```
$ pkg set-publisher -k /path-to-certs/repo2_myuser.key.pem \
-c /path-to-certs/repo2_myuser.crt.pem \
-p https://pkg-sec.example.com/repo2/
```

9. **Pruebe el acceso a `repo3`.**

   Utilice el certificado `repo1` para probar el acceso a `repo3`.

```
$ pkg set-publisher -k /path-to-certs/repo1_myuser.key.pem \
-c /path-to-certs/repo1_myuser.crt.pem \
-p https://pkg-sec.example.com/repo3/
```

Utilice el certificado `repo2` para probar el acceso a `repo3`.

```
$ pkg set-publisher -k /path-to-certs/repo2_myuser.key.pem \
-c /path-to-certs/repo2_myuser.crt.pem \
-p https://pkg-sec.example.com/repo3/
```

Índice

S

V

Z