Universitat Autònoma de Barcelona

etse

Escola Tècnica Superior d'Enginyeria

PID 351
Automatización de la gestión del ciclo de vida de entornos Linux y Solaris

Número de Proyecto:	351
Versión :	2.0
Autor:	Felipe Herrera Martínez
Fecha del Documento:	18/02/2008

“Quiero agradecer enormemente a Antonio Portero la paciencia e insistencia demostrada a lo largo de los años, que ha sido igual o superior a la de mis padres y amigos, para que realizara el Proyecto Final de Carrera, tras 5 años de haber finalizado todas las asignaturas previas al mismo.

También me gustaría dedicar este proyecto a mi esposa Nuria, que ha sido mi punto de apoyo tanto en mi carrera profesional como en mi vida, y sobretodo me gustaría dedicárselo a mi hijo Joan, que nacerá entre las semanas en que yo haga entrega de esta memoria del proyecto y pueda presentarlo ante el tribunal. **”**

Tabla de Contenidos

0. Control

0.1. Revisiones

Versión	Autor	Revisión	Descripción	Date
1.0A	Felipe Herrera	Antonio Portero	Primera Revisión de la planificación	18/02/2008
2.0	Felipe Herrera	Antonio Portero	Segunda Revisión del proyecto	26/04/2008

0.2. Figuras y tablas

1. Introducción

El propósito de este documento es definir el proyecto "Automatización de la gestión del ciclo de vida de entornos Linux y Solaris" realizado por Felipe Herrera Martínez bajo el número de proyecto PID 351 y con la supervisión de Antonio Portero como director de proyecto.

En este documento se detallarán los diferentes aspectos que se han considerado claves para la ejecución del proyecto en el entorno real del cliente y que seguirán la siguiente lista:

Necesidades y requisitos del cliente para la Automatización de la gestión del ciclo de vida de los diferentes entornos.
Posibilidades y estudio de herramientas que Sun Microsystems dispone para hacer frente a esta automatización.
Estudio de viabilidad.
Arquitectura propuesta para la solución.
Planificación temporal y Valoración económica.

Este proyecto ha formado parte de uno de mis logros Profesionales en Sun Microsystems. Gracias al empeño y a la lucha por introducir y presentar Sun Connection Satellite y xVM Ops Center, como productos para la automatización y ayuda a la gestión de actualizaciones y despliegue de aplicaciones en la Península Ibérica, he logrado que se me reconociese el trabajo realizado y actualmente forme parte de un grupo de ámbito Internacional, con sede en San Francisco, como *Senior Sales Engineer* y responsable técnico de esta línea de productos en la Península Ibérica.

A su vez, este proyecto, es un proyecto referente para Sun Ibérica por lo que respecta a la automatización y la gestión masiva de parcheos con Herramientas autónomas, por lo que el éxito del mismo supone un éxito para la compañía y referencia a tener en cuenta para el resto de clientes de la Península Ibérica.

2. Definición del Proyecto

El objeto del proyecto es la evaluación e implantación de una herramienta capaz de asegurar la completa automatización de procesos de despliegue de parches, paquetes y aplicaciones, así como el control de la configuración siguiendo la metodología *ITIL (®) (IT Infrastructure Library),* de todos los sistemas Linux y Solaris del cliente, teniendo en cuenta los Objetivos planteados por el cliente, así como el entorno del mismo, tal y como se detallará más adelante, proponiendo entre varias herramientas la mejor para estos objetivos y éste entorno.

2.1. Objetivos del cliente

El objeto del proyecto es la evaluación e implantación de una herramienta capaz de asegurar la completa automatización de procesos de despliegue de parches, paquetes y aplicaciones, así como el control de la configuración siguiendo la metodología *ITIL* de todos los sistemas Linux y Solaris del cliente.

Durante los últimos años se ha comprobado como la complejidad de los Sistemas que forman los Centros de Cómputo del cliente, el número que lo forman, la demanda de Servicio y la gran dependencia que existe de estos para el Negocio ha ido creciendo.

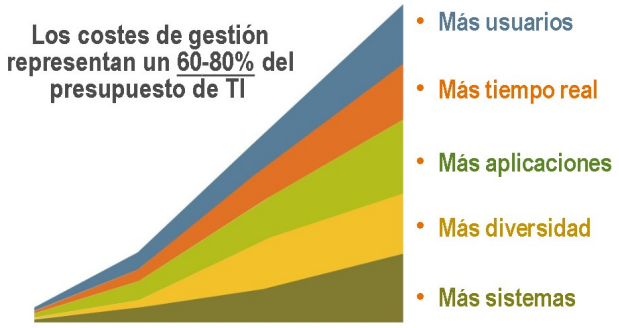

figura 1 – Proceso de crecimiento de la Complejidad de los Centros de Cómputo según IDC.

En clientes que cumplen este patrón y de gran envergadura, gestionar el parcheado de los Sistemas se hace cada vez más complicado y costoso, a la vez que necesario, ya que requieren que sus plataformas sean 100% fiables y aunque dispongan del personal suficiente y preparado para realizar esta operación, la mayoría de Sistemas deben ser actualizados en horario no laboral, debido a que estos sistemas dan un Servicio concreto y en muchos casos crítico para el negocio del cliente. Esta peculiaridad supone incrementar los costes necesarios para gestionar estos sistemas, por lo tanto la adopción de esta herramienta en el cliente debe aportar beneficios claros de ahorro de costes en el parcheado y despliegue de aplicaciones en sus plataformas, así como la disminución del riesgo de operación debido a la completa automatización del proceso de parcheado y despliegue.

Para poder asumir de forma viable los objetivos descritos durante la implantación de dicha herramienta en la infraestructura global del cliente, formada por aproximadamente unos 500 Sistemas Open Unix y divididos entre Sistemas Red Hat Linux y Sistemas Solaris, se han debido someter a estudio las necesidades de automatización de procesos que habitualmente se realizaban de forma manual, teniendo en cuenta las condiciones marco definidas por el cliente y las limitaciones tanto temporales como económicas que existían para hacer frente a este proyecto.

El no afrontar adecuadamente estos objetivos puede tener implicaciones no deseadas en los Sistemas que integran el Cómputo de Datos, afectando a su correcto funcionamiento debido a parcheos insuficientes o fallos en la operación durante la realización de los mismos, lo que implicaría un mayor tiempo de parada de los Sistemas y consecuentemente degradación del Servicio y pérdida de parte del Negocio. Implantar una herramienta capaz de automatizar la actualización y control de los Sistemas, permite:

- Reducir el coste de la mano de obra necesaria para realizar esta tarea. así como la experiencia necearia para realizarla.
- Reducir el riesgo de pérdida de Negocio debido a una actualización insuficiente de los Sistemas, que provoquen fallos inesperados en los mismos.
- Aumentar la capacidad de Servicio y consecuentemente de Negocio en cada uno de los Sistemas gestionando, obteniendo el máximo partido y rendimiento de los mismo.

2.2. Entorno del cliente

El entorno del cliente en el que nos vamos a centrar durante este proyecto, se encuentra integrado como un subconjunto dentro del marco de sistemas del Departamento de Explotación. En concreto, este subconjunto es el que forma la *Infraestructura Global Unix* y está formado por unos 500 sistemas, divididos entre Sistemas *Red Hat Linux y Sistemas Solaris*. Cada uno de ellos ofrece un Servicio y tienen una criticidad definida por el propio cliente, dependiendo de la importancia de su existencia y el valor que aportan al Negocio.

Dentro de estos 500 sistemas, como más tarde se detallará, hemos escogido una pequeña muestra de unos 25 sistemas, para poder adoptar de forma viable el plan de integración de una Herramienta que posibilite la automatización de procesos de parcheo y despliegue de aplicaciones en la *Infraestructura Global Unix* del cliente.

Todos los sistemas de la Infraestructura cliente tienen dos redes diferenciadas físicamente:

Red de Producción. Esta red se utiliza única y exclusivamente para dar Servicio, es la más crítica, no solamente a nivel de seguridad, ya que es la red pública por donde hay que prever ciertos ataques malintencionados , sino también debido a que toda la explotación de los sistemas se realiza por ésta. Si el sistema fuese un Servidor de Aplicaciones o Servidor Web, sería por ésta por donde se atendería cualquier petición por parte de una aplicación cliente.

Red de Gestión. Esta red se utiliza para todas las tareas de administración y gestión necesaria para mantener los sistemas, es desde ésta por donde se monitorizan los sistemas y se conecta el personal de Administración y control de Explotación del Servicio.

Cada servidor tiene una dirección IP en cada una de estas dos redes, cada una de ellas se asigna por una Red Virtual (*VLAN*) diferente, por lo que no es posible que los sistemas accedan entre ellos a través de estas dos redes diferenciadas.
En la Red de Gestión, existe un método de publicación, que permite asignar una dirección IP a un sistema y que este sea visible por el resto de sistemas a través de las diferentes redes Virtuales existentes dentro de la Red de Gestión, sin necesidad de realizar modificaciones en los sistemas de Seguridad, de control o de gestión de rutas que el cliente dispone para esta red.

El Servidor de la Herramienta de Gestión escogida para realizar la automatización de las actualizaciones, estará en esta red de publicación con acceso a Internet, ya que es uno de los requisitos y necesidades del producto para su correcto funcionamiento, dependiendo de la herramienta que finalmente se escoja para el proyecto.

En el caso de ciertos sistemas, en concreto los Sistemas Operativos Solaris 10 con Zonas (ver Apéndice 5), actualmente hay algunos de ellos que no tienen acceso a la Red de Gestión, ya que sólo la Zona Global tiene acceso a la misma, esto a priori puede ser un problema por la dificultad de no poder acceder el resto de Zonas locales a la Red de Gestión y consecuentemente al Servidor que automatice los procesos de actualización. No obstante, en esta primera fase del proyecto se trabajará con aquellos sistemas cuyas Zonas tengan acceso a la Red de Gestión, ya que el cliente se encuentra en plena fase de cambio de su infraestructura de Red y estandarizará la situación para que todos los sistemas tengan acceso a ésta.

El cliente actualmente dispone de aproximadamente unos 500 sistemas sobre los que se requiere implantar este proceso de automatización y que forman la Infraestructura Global Unix. Estos sistemas se podrían dividir, dependiendo de la necesidad de la división: por la tipología o criticidad del Servicio que ofrecen, por el cliente final al que ofrecen el servicio, por la Arquitectura Hardware o bien por la naturaleza o Distribuidor del Sistema Operativo.

Nosotros nos vamos a centrar en la Naturaleza del Sistema Operativo, para agrupar estos sistemas y poder implantar mejor los procesos de automatización de la gestión del ciclo de vida de cada uno de estos grupos, ya que independientemente del servicio o criticidad de cada uno de los sistemas, es la distinción primordial y la que nos hará desarrollar un proceso u otro a la hora de automatizar el parcheo o actualización de cada sistema.

Teniendo en cuenta los diferentes Sistemas Operativos que dispone el cliente en el entorno que va a ser objeto del proyecto hemos considerado adecuado diferenciar tres grupos:

Grupo 1 : Este grupo de sistemas estaría formado por equipos cuyos Sistemas Operativos sean *Red Hat Enterprise Linux AS 3, 4 y 5*.

Grupo 2 : Este grupo de sistemas estaría formado por equipos cuyos Sistemas Operativos sean *Solaris 8,9 y 10*.

Grupo 3 : Este grupo de sistemas estaría formado por equipos cuyos Sistemas Operativos sean *Solaris 10 con zonas* (ver Apéndice 5)

2.3. Planificación y estudio de viabilidad del Proyecto

Para poder asegurar una adopción viable del proyecto y conseguir los beneficios deseados por el cliente se han tenido que tener en cuenta varios factores primordiales, estos factores primordiales fueron planteados por el cliente como requisitos indispensables para poder alcanzar los Objetivos de este proyecto, tal y como se plantea a continuación.

2.3.1. Requisitos propuestos por el cliente a Sun Microsystems

Para poder abordar este proyecto, el cliente nos propuso una serie de requisitos que se debían cumplir, los requisitos principales fueron:

Se debe aportar una solución de automatización del proceso de parcheado, tal que aporte máxima repetitividad, sostenibilidad y eficiencia a la misma y que permita asumir el parcheado de más de 20 sistemas por semana y de forma autónoma.
Actualmente el cliente se encuentra con la problemática de que muchos de sus sistemas se encuentran desactualizados y las tareas de administración diarias imposibilitan que el personal de operaciones haga frente a la actualización y parcheo de los sistemas que forman la Infraestructura Global de forma asidua.
La situación de desactualización conlleva a la aparición de agujeros de seguridad en los sistemas, debido a parcheos insuficientes, P*roblemas* o *Incidentes* conocidos y resueltos con la aplicación de nuevos parches o paquetes de producto y por lo tanto una degradación en el Servicio que ofrecen los sistemas.

La herramienta, a parte de automatizar el proceso de actualización, debe permitir controlar el nivel de versiones de parches y paquetes de los diferentes sistemas del cliente. Muchas de las tareas habituales de gestión sobre las plataformas requieren que éstas tengan el mismo nivel de parches o versiones de paquetes instalados, en concreto el cliente migra con cierta asiduidad Servicios de unos sistemas a otros y por motivos de seguridad se desea controlar que sistemas cumplen el mismo nivel de versiones para tener posibles sistemas candidatos sobre los que hacer la migración.

La herramienta debe permitir una Marcha atrás segura en caso de problemas a la hora de restaurar el servicio, ya que tras la actualización del Sistema, éste puede que no se comporte según lo deseado, y dada esta situación, se debe llevar el sistema a la situación previa a su modificación.

La solución debe estar funcionando un mes después del inicio del proyecto, ya que este proyecto tiene una gran visibilidad e importancia en la organización interna del cliente y necesitan estabilizar el nivel de versiones en su Infraestructura Global tan pronto como sea posible.

Debido a la limitación presupuestaría que tiene el cliente asignada para abordar este proyecto durante este año fiscal, era condición indispensable no exceder este presupuesto.

Teniendo en cuenta los requisitos que se debían cumplir, estudiamos las diferentes opciones que le podíamos ofrecer al cliente así como la opción más viable para este proyecto.

2.3.2. Estudio de Viabilidad

Tras tener en cuenta los requisitos propuestos por el cliente, sometimos a estudio las herramientas que Sun Microsystems disponía hasta la fecha para determinar cual de ellas podría adaptarse de mejor modo a las necesidades del cliente y que por los requisitos temporales expuestos por el mismo, cual de ellas requería un tiempo de implementación mínimo para adquirir los Objetivos en el mínimo tiempo posible.

De las principales herramientas para la ayuda a la gestión de Centros de Cómputo que dispone Sun Microsystems, se sometieron a estudio tres de las que podían cubrir las expectativas del cliente:

- Sun N1 *Service Provisioning System*
- *Sun Connection Satellite*
- *Sun xVM Ops Center*

De las principales funcionalidades que cada una aporta a la gestión de Sistemas nos debíamos centrar en lo fácil que cada una de ellas emula de forma automática los procesos normales de parcheo de un Sistema Open Unix. Para que se desarrolle de forma segura un proceso de parcheo, la herramienta nos debe permitir realizar Planes de Contingencia y de Marcha atrás de forma automática, permitiendo recuperar el sistema en caso de sufrir un cambio inesperado durante la actualización del propio Sistema Operativo.

A continuación se detallan las principales funcionalidades de cada una de las herramientas sometidas a estudio:

Sun N1 Service Provisioning System, ideal para negocios donde se necesita un despliegue rápido y efectivo de aplicaciones de forma masiva y en sistemas totalmente heterogéneos. Mediante Sun N1 *Service Provisioning System* el cliente tiene mayor control y consistencia del despliegue de sus aplicaciones.

Claves de *Sun N1 Service Provisioning System* :

- Facilita la rápida integración y reciclaje del servicio que desempeña cada Sistema mediante la automatización del despliegue a diferentes niveles de aplicación y en entornos totalmente heterogéneos (Windows, AIS, HPUX, Linux y Solaris).
- Incorpora un modo de simulación que facilita la detección de errores en el proceso de despliegue antes de su ejecución real.
- Guarda todas las acciones y cambios realizados mediante la herramienta, permitiendo el control de versiones y marcha atrás en caso de ser necesario.
- Incorpora todas las herramientas necesarias para el desarrollo de modelos específicos de despliegue.

Sun Connection Satellite, ideal para negocios que necesitan desplegar actualizaciones de aplicaciones y Sistema Operativos. Ayuda a la gestión del ciclo de vida de forma automática de los entornos Linux y Solaris.

Claves de *Sun Connection Satellite:*

- Gestiona desde una única consola, entornos Solaris, Red Hat y SuSE.
- Automatización del despliegue y gestión de las actualizaciones según una base de *Conocimiento Universal* donde reside cada una de las dependencias, acciones y reglas necesarias para instalar cada parche o paquete de los diferentes distribuidores antes mencionados.
- Mitiga de forma proactiva los riesgos y vulnerabilidades a las que puede estar sometido un Sistema gracias a un motor de dependencias de necesidades que analiza de forma independiente cada uno de los parches, paquetes, incidente y conflictos a los que se encuentra sometido un sistema.
- Posibilita la planificación de forma automática de la ejecución de un trabajo de actualización con todas las medidas necesarias de contingencia y marcha atrás para abordar de forma segura el cambio.
- Posibilita la marcha atrás mediante la generación de forma automática de fotos temporales del

inventario de cada sistema, antes, durante y después de un cambio.

- Control de Gestión de la configuración basado en metodología *ITIL*.

Sun xVM Ops Center es la siguiente versión de Sun Connection Satellite, que a parte de integrar todas las funcionalidades de este producto integra las funcionalidades de otras herramientas de gestión que Sun Microsystems dispone y cuyas funcionalidades principales que aportan al producto se pueden ver en la *figura 2*.

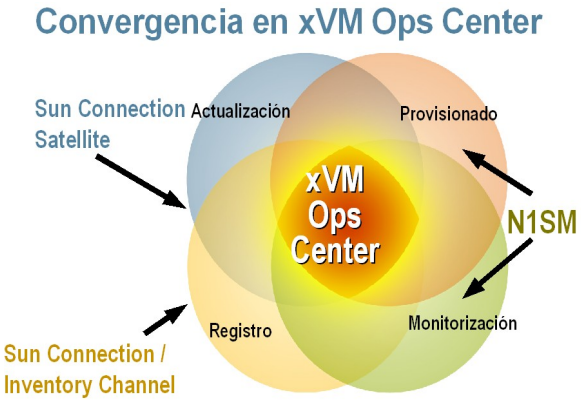

figura 2: Convergencia Sun xVM Ops Center

Claves de Sun xVM Ops Center:

- Todas las claves mencionadas para *Sun Connection Satelltie* más:
- Control de Gestión de la configuración e Inventario basado en metodología *ITIL*.
- Solución integral para la gestión – Descubrimiento *Bare Metal*, Chequeo y Despliegue de Sistemas Operativos.
- Monitorización Básica de Hardware Sun y Sistema Operativo.
- Gestión de granjas de servidores (Encender, Parar, Control y Gestión de Chasis de Blades, Control por Consola de los equipos).
- Multiplataforma Soporta Solaris 8,9 y 10 SPARC, Solaris 10 X64 y más de 40 distribuciones Linux.

Tras estudiar en detalle las tres herramientas y analizar las funcionalidades que nos aportaban cada una de ellas, así como las principales necesidades que requería el cliente y descritas en apartados anteriores, desestimamos como herramienta para abordar este proyecto **Sun N1** ***Service Provisioning System***, ya que esta herramienta es una herramienta orientada al despliegue de Sistemas Operativos y aplicaciones y para adaptarla a las funcionalidades requeridas de despliegue de actualizaciones, necesitaba de un desarrollo dedicado a la infraestructura del cliente mayor que cualquiera de las otras dos herramientas propuestas, ya que estas dos últimas disponen de la lógica necesaria y dedicada a tales fines. Consecuentemente el tiempo de puesta en marcha de la solución en el entorno global de cliente sería mucho más costosa en tiempo y presupuesto, ambos requisitos también expuestos como claves para la implantación del proyecto desde inicio por el cliente.

La implantación de ***Sun Connection Satellite*** o ***Sun xVM Ops Center*** en este proyecto, teniendo en cuenta los requisitos temporales, facilitarían tener el entorno activo en un tiempo mucho menor que si utilizásemos ***Sun N1 Service Provisioning System***, ya que como hemos podido observar ambas están orientadas a estos fines y el desarrollo necesario para actualizar sistemas Linux y Solaris es mínimo, lo que nos permite tener la solución funcionando un mes más tarde del inicio del proyecto, tal y como pedía el cliente.

Teniendo en cuenta las fechas en las que se planteaba el inicio de proyecto, Febrero del 2008, hizo que la elección de ***Sun Connection Satellite*** como herramienta óptima para este proyecto, frente a ***Sun xVM Ops Center,*** tomará más fuerza, ya que la versión GA 1.0 de ***Sun xVM Ops Center*** se retrasó hasta Marzo, y aunque hubiésemos podido planificar una arquitectura y esperar a la liberación del producto para su instalación definitiva, decidimos no optar por esta opción, ser más conservadores y trabajar con un producto estable y que estaba funcionando en muchos de nuestros cliente del resto del mundo; no obstante este proyecto era el primer proyecto de estas características para Sun Ibérica y era un referente a seguir con respecto al uso de una herramienta de automatización y despliegue de actualizaciones de paquetes y parches en un cliente con esta envergadura y con este tipo Sistemas, por lo que era necesario conseguir los Objetivos para precisamente ser un buen referente en Sun Ibérica y poder tomar buen ejemplo en muchos de nuestros clientes con tales dimensiones e importancia para la compañía.

Llegados a este punto en el plan de proyecto que se iba a proponer al cliente, debíamos defender la opción que Sun Microsystems le estaba ofreciendo para conseguir los Objetivos como la opción óptima frente a otros posibles Distribuidores y que el cliente escogiera solución. Para ello, teniendo en cuenta que el entorno Global del cliente sobre el que se centra el proyecto, es un entorno Open Unix, divididos entre Sistemas Red Hat Linux y Sistemas Sun Solaris, se sometieron a estudio dos herramientas de dos Distribuidores, a la vez que Competidores, diferentes, para analizar de esta forma los pros y contras de la solución propuesta, frente a las posibles soluciones que podían ofrecer estos para un entorno de estas características.

Estas herramientas fueron:

- Red Hat Network Satellite – de Red Hat
- Opsware System – de HP

A continuación se van a comparar estas dos herramientas con Sun Connection Satellite, para analizar los pros y los contras de la Solución propuesta respecto a cada una de ellas:

Comparación de Sun Connection 1.1.1 con Red Hat Network Satellite 5.0 y Opsware System 6 de HP

Los tres productos ofrecen una solución centralizada para la gestión y control de despliegue de aplicaciones y actualizaciones de Sistemas Operativos, no obstante, los diferentes productos tienen un conjunto de funcionalidades clave que diferencian un producto del otro, así como la Infraestructura Global a la que van destinados cada unos de ellos.

Sun Connection provee una serie de mejoras respecto a *Red Hat Network 5.0* y *Opsware System 6*.

- **Base de Datos de Conocimiento Universal (KB)** – *Sun Connection* provee una base de Datos de Conocimiento Universal en la que residen todas las acciones, dependencias y reglas necesarias para poder instalar un parche o paquete en los diferentes Sistemas Operativos que se Soportan (Red Hat, SuSE y Sun Solaris). Esta Base de Datos de Conocimiento está probada y certificada para los Distribuidores que soporta.
- **Amplia soportabilidad de Sistemas Operativos** – *Sun Connection* provee el completo soporte y certificación de todos los sistemas Sun Solaris 8, 9, 10 y Solaris 10 con zonas (ver Apéndice 5), a su vez soporta más de 40 distribuciones Red Hat Linux y SuSE a través de diferentes plataformas Hardware (x86, AMD64, Itanium, PPC y zLinux), mientras *Red Hat Network* sólo soporta las propias distribuciones Red Hat, así como un número limitado de distribuciones Solaris.
- **Repositorio de Software comprensivo y automatizado** – La Base de Datos de Conocimiento Universal provee acceso a todas las versiones de parches y paquetes de todo el ciclo de vida de los diferentes Sistemas Operativos, no única y exclusivamente la última versión, sin requerir un mantenimiento manual para la actualización de esta Base de Datos, ya que se actualiza de forma automática a través de *Internet*. De forma adicional Sun Connection automáticamente incluye en su Base de Datos de Conocimiento todos los paquetes *Sun Freeware*.
- Modo simulación – Permite a los administradores simular el despliegue de la actualización o parcheo en cientos o miles de Sistemas antes de ejecutarlo, de este modo se pueden prever los cambios que se van a producir en estos sistemas y estudiar su impacto, siguiendo la metodología *ITIL* implantada en el cliente y abrir la Orden de cambio pertinente, para su consecuente aprobación, en caso de ser aceptado.
- **Clonado y marcha atrás** – Gracias a un sofisticado sistema de control de inventario, permite a los administradores detectar las diferencias entre varios Sistemas, permitiendo clonarlos, mediante la automatización de las acciones necesarias para que estos tengan el mismo inventario. También permite detectar estos cambios en el inventario de un mismo sistema tras cada cambio sufrido por el mismo, permitiendo volver atrás el cambio sufrido por el sistema o parte del mismo. *Red Hat Network* no provee este nivel de granularidad, ya que el control de inventario que ofrece es temporal y no permite llegar al mismo nivel de detalle que Sun Connection Satellite.
- **EIS (Enterprise Installation Standards) Baseline** – Mensualmente Sun provee un nivel de actualizaciones de parches y paquetes para todos sus Sistemas Operativos llamado EIS Baseline, mediante Sun Connection podemos controlar si nuestros sistemas cumplen esta Baseline o que Baselines tenemos disponibles para aplicar a nuestros Sistemas, no requiere que el administrador genere un grupo de parches específico para sus sistemas, ya que mediante esta Baseline podrá desplegar los recomendados realizadas por Sun Microsystems, para Sistemas Operativos Solaris.
- **Resolución de dependencias** – Sun Connection asegura que todo el Software sea desplegado con todas las dependencias que este requiera para ser instalado, de forma automática, y que funcione correctamente, no es necesario que un administrador investigue todas las dependencias que requiere cada software ya que gracias a la KB Sun Connection las provee.
- **Soporte a scripts** – Sun Connection permite añadir, como tareas previas y/o posteriores al despliegue de una actualización, scripts que permitan una mayor automatización del proceso total de actualización de un Sistema, aportando medidas de seguridad a un parcheo, tales como parar un servicio o realizar una copia de seguridad de forma totalmente automática e implícita en el propio proceso de Actualización, emulando todos aquellos pasos de contingencia que realizaba de forma manual un Administrador.

Red Hat Nework aporta una serie de beneficios al cliente que Sun Connection Satellite, hasta la fecha, no puede ofrecer:

- **Monitorización de Sistemas** – *Red Hat Network* provee monitorización de Sistemas Operativos y aplicaciones, permite enviar notificaciones por correo cuando los márgenes de monitorización establecidos han sido superados, también permite generar gráficos para analizar la carga de cada uno de los Sistemas.
- **Notificación por correo de nuevas actualizaciones** – El cliente tiene la posibilidad de recibir una notificación por correo cuando Red Hat libera nuevas actualizaciones.

- **Despliegue de Sistemas Operativos Integrado** – *Red Hat Network* permite desplegar Sistemas Operativos Linux en los diferentes Sistemas de

la Infraestructura Global del cliente, permitiendo realizar instalaciones de diferentes equipos de forma centralizada y totalmente automatizada.

- **Modo Desconectado** – *Red Hat Network* a diferencia de *Sun Connection*, permite actualizar los sistemas de la red, sin necesidad de que el Servidor Satélite se encuentre conectado a Internet para tener acceso a nuevas actualizaciones. Esta funcionalidad se requiere para muchos clientes que desean tener totalmente actualizados sus equipos, pero ven un problema de Seguridad el tenerlos conectados a Internet a través de un servidor Satélite, aún teniendo en su infraestructura Sistemas Corta Fuegos par evitar cualquier problema en la seguridad. Esta funcionalidad se convierte en necesidad en muchos de nuestros clientes, tales como Bancos, Organismos Estatales, etc.
- **Criticidad de las actualizaciones** – *Red Hat Network* permite catalogar las actualizaciones liberadas por su criticidad (alta/media/baja), de este modo permite al administrador detectar las problemas de criticidad alta que afectan a su Infraestructura y desplegar las actualizaciones que los solventan.

Opsware de HP se presenta al cliente como la solución Íntegra para la gestión y control de los Centros de Cómputo, aportando una serie de beneficios al cliente que Sun Connection Satellite, hasta la fecha, no puede ofrecer:

- **Despliegue de Sistemas Operativos** – Una de las bazas más importantes de *Opsware System*es precisamente ésta; *Opsware System* permite desplegar de forma totalmente automática diferentes Sistemas Operativos (Windows, HP-UX, AIX, Solaris y Linux), permitiendo realizar instalaciones de diferentes equipos de forma centralizada.
- **Descubrimiento automatizado de Sistemas** – *Opsware System* permite descubrir los sistemas que existen en la Infraestructura Global del cliente, sin necesidad de que el cliente tenga que instalar uno por uno la aplicación necesaria para conectarse al Servidor Satélite. *Opsware* certifica su uso en Sistemas Windows, HP-UX, AIX, Solaris, Linux y dispositivos de red, tales como switches y routers, permitiendo configurarlos y gestionarlos de forma centralizada.
- **Descubrimiento automatizado de Aplicaciones** – *Opsware System* provee la posibilidad de descubrir de forma automática ciertas aplicaciones, tales como Oracle, BEA WebLogic, Apache, Microsoft SQL, etc, pudiendo gestionarlas, desplegarlas y monitorizarlas, sin necesidad de desarrollar complicados perfiles que permitan esta funcionalidad. De forma totalmente automática, permite descubrir las dependencias existentes entre estas aplicaciones y el entorno global del cliente (servidores, dispositivos de red), permitiendo generar gráficos para la mayor comprensión y entendimiento de la Infraestructura Global del cliente.
- **Integración con la Herramientas de control ITIL del cliente** – *Opsware System* permite la completa integración del producto con la Base de Datos de Configuración (CMDB) del cliente, permitiendo de esta forma tener un total control de la gestión del Inventario e integrar funcionalidades de Documentación más comprensibles y adaptables a las necesidades del cliente, teniendo en cuenta su Infraestructura ya existente.

Tras conocer en detalle las funcionalidades que aportan cada una de las herramientas, podríamos definir a que tipo de entornos y con que fines estarían altamente recomendadas cada una de ellas y posteriormente decidir si Sun Connection Satellite es mejor opción que el resto de competidores para este proyecto:

Red Hat Network 5.0 – Esta herramienta estaría destinada para entornos única y exclusivamente Red Hat Linux, aportando una solución global para el despliegue de Sistemas Operativos, Monitorización y Actualización de los mismos, ofreciendo en gran detalle y precisión las funcionalidades necesarias para cubrir la gestión integral de este tipo de entornos.

Opsware System 6 – Esta herramienta estaría destinada para entornos totalmente heterogéneos, pudiendo cubrir todas las arquitecturas Open (Windows, HP-UX, Solaris y Linux) y MainFrame (AIX) ofreciendo funcionalidades en gran detalle y precisión para el despliegue de Sistemas Operativos y aplicaciones de forma fácil y completamente automatizada. Esta herramienta ofrece a su vez un gran control del Inventario y se integra fácilmente en las herramientas de Inventario que el cliente ya disponga.

Sun Connection Satellite 1.1.1 – Esta herramienta estaría destinada para entornos Open Unix, con mayoría de Sistemas Solaris y Linux, ofreciendo funcionalidades, en gran detalle, para la actualización y despliegue de aplicaciones y obteniendo, de esta forma, gran control del cambio que cada actualización o despliegue supone en cada uno de los Sistemas.

Por lo tanto, teniendo en cuenta los Objetivos que se pretenden cubrir en este proyecto, así como los detalles del Entorno del cliente al que va dirigido, podemos decir, con toda seguridad, que la herramienta óptima para abordar este proyecto es ***Sun Connection Satellite*** .

Las funcionalidades que aporta cubren de forma sobrada los requisitos planteados por el cliente y el Entorno Certificado para el uso de *Sun Connection Satellite* se ajusta a la Infraestructura Global Unix del cliente, en la que centramos este proyecto.

Tras tener claro que la herramienta a implantar para la automatización del parcheado y actualización de paquetes iba a ser ***Sun Connection Satellite***, por los motivos antes mencionados, se nos planteaba otro reto, el de cubrir este proyecto con un límite presupuestario por parte del cliente para este año.

Teniendo en cuenta este límite presupuestario y que imposibilitaba poder abarcar la adopción de ***Sun Connection Satellite*** en la plataforma Global del cliente en una única fase o proyecto, propusimos dividir esta adopción en las dos fases siguientes:

- **Fase I**: Esta fase es la que se detalla en este plan de proyecto y consiste en escoger 25 de los 500 sistemas de forma que podamos implementar en esta primera fase los procesos necesarios para actualizar estos sistemas siguiendo la tipología de 3 grupos que hemos diferenciado dentro del Entorno del cliente. El hecho de escoger 25 sistemas, se debe al modo de licenciamiento de ***Sun Connection Satellite***, al adquirir la licencia Satélite incluye la instalación del Sistema Satélite y los 25 primeros agentes. Y el hecho de escoger 3 grupos de Sistemas dentro de la plataforma Global del cliente, nos permitía definir claramente 3 tipologías diferentes de parcheo independientemente del servicio o criticidad de los sistemas, reduciendo al máximo el desarrollo necesario y consecuentemente el coste necesario para la automatización de la actualización en la plataforma global del cliente. Por lo tanto en esta primera fase se da un peso más importante en la oferta económica a los servicios profesionales valorados para la implementación.

- **Fase II**: Esta fase no se contemplará en este plan de proyecto y consistirá en la adopción de Sun Connection Satellite en el resto de plataformas del cliente, reutilizando el desarrollo realizado durante la primera fase, de forma que al abaratar los costes en los servicios Profesionales valorados en esta segunda fase, el cliente puede hacer frente a la compra del resto de licencias.

Por lo tanto al dividir la adopción en dos fases no sólo hacemos frente a un requisito presupuestario, sino que aseguramos una mejor adopción de la herramienta en el entorno global del cliente.

3. Arquitectura propuesta para la Solución

Sun Connection Satellite es un software desarrollado y diseñado para la gestión del ciclo de vida de los sistemas Linux y Solaris. Con *Sun Connection Satellite* disminuiremos de forma significativa el tiempo y los conocimientos necesarios para crear y desplegar actualizaciones de forma fiable y segura, a su vez nos permitirá controlar el Inventario Hardware y Software de cada uno de los Sistemas y automatizar el proceso de control del cambio de estos Inventarios; a este proceso de control automatizado del Inventario Hardware y Software de cada uno de los sistemas lo conocemos como *Inventario Activo.*
Gracias al *Inventario Activo* podremos realizar funcionalidades de *Marcha atrás* y *Restauración* de Sistemas Operativos tras un cambio, al igual que el clonado de Sistemas, mediante la comparación de Inventarios en diferentes estados del tiempo de un mismo Sistema o de diferentes Sistemas, de este modo podremos tener pleno control de los diferentes grupos de criticidad, desplegando de forma ordenada conjuntos de cambios desde entornos menos críticos hasta entornos más críticos dependiendo de las políticas de cambio que el cliente haya planificado siguiendo una metodología *ITIL* de gestión y control del cambio.
Las funcionalidades principales que aporta *Sun Connection Satellite* son:

- Gestión inteligente de parches y paquetes.
- Gestión del cumplimiento de *Baselines* de Solaris.
- Agrupación de Sistemas
- Planificación de trabajos.
- Comparación de perfiles.
- Simulación y Marcha atrás.
- Cache local de parches.
- Soporte a componentes propios del cliente.
- Gestión de la configuración.

3.1. Arquitectura Sun Connection Satellite

Sun Connection Satellite consiste en una serie de componentes que deben ser instalados en diferentes sistemas de la infraestructura Global del cliente, parte de estos componentes forman la arquitectura local, un tercer componente, llamado *Servidor Universa*l, reside en Sun Microsystems y proveerá las actualizaciones al Servidor Satélite del cliente.

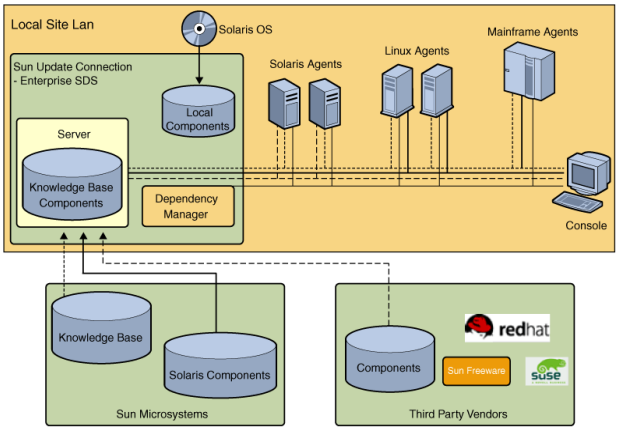

figura 3: Arquitectura Sun Connection Satellite

El ***Servidor Universal*** se mantiene en Sun. Éste es un servidor de aplicaciones Web que incluye la *Base de Conocimientos Universal* (*Universal Knowledge Base - KB*) y los componentes software de varias distribuciones. Este servidor mantiene también los ficheros de firmas y otras medidas de seguridad.

El ***System Dependency Server*** (SDS) reside en el Servidor Satélite del cliente, este servidor es el proxy al Servidor Universal. El servidor contiene los siguientes componentes:

- Un Servidor Apache embebido, que descarga las actualizaciones certificadas del Servidor Universal.
- La *Base de Datos de Conocimiento Local,* que es el repositorio de paquetes y sus reglas de despliegue.
- El *Dependency Manager* que gestiona los agentes, también llamado motor. El Dependency *Manager* también gestiona las comunicaciones entre los agentes, consolas, la interfaz de comandos y una API, que permite al cliente desarrollar programas en *perl* e integrar funciones básicas de *Sun Connnection Satellite*, para poder integrar de este modo funcionalidades del producto con otros programas del cliente e incluso desarrollar sus propias herramientas de despliegue.

Un ***Agente*** es un sistema gestionado. El agente ejecuta la aplicación de resolución de dependencias. Este componente busca la solución más efectiva para realizar cualquier trabajo. Cada Sistema gestionado busca la mejor solución para su propio software.

La ***Consola*** provee la interfaz gráfica de usuario desde la que se inician las tareas sobre los agentes instalados en los Sistemas gestionados. También se puede hacer uso de la interfaz de comandos (CLI) de forma opcional para iniciar estas tareas.

La ***API*** es una interfaz de desarrollo opcional que se le ofrece al cliente, para poder integrar las funcionalidades de *Sun Connection* en las propias aplicaciones del cliente.

Todos los requisitos y detalles técnicos de la Instalación de *Sun Connection Satellite* en el cliente se han detallado en el Apéndice 6.

3.2. Definición de políticas de parcheo

Como ya se ha mencionado anteriormente, se ha definido una tipología de parcheo para cada uno de los tres grupos de sistemas comprendidos dentro de los 25 sistemas, que junto al cliente se han seleccionado para implementar este proyecto; entendiendo como política todas aquellas tareas de parcheado que sólo se pueden aplicar a un grupo de sistemas en contraposición a otro grupo y que forman un proceso necesario para actualizar de forma adecuada y completamente segura cada grupo de sistemas, aportando todas las medidas de contingencia y marcha atrás necesarias.

Recomendado por Sun Microsystems y aprobado por el cliente se han definido los tres grupos de sistemas siguientes:

Grupo 1 : Este grupo de sistemas estaría formado por equipos cuyos Sistemas Operativos sean *Red Hat Enterprise Linux AS 3, 4 y 5*.

Grupo 2 : Este grupo de sistemas estaría formado por equipos cuyos Sistemas Operativos sean *Solaris 8,9 y 10*.

Grupo 3 : Este grupo de sistemas estaría formado por equipos cuyos Sistemas Operativos sean *Solaris 10 con zonas* (ver Apéndice 5)

Una vez acordado con el cliente la definición de estos tres grupos de parcheo, se detalla a continuación, los planes de acción a seguir en cada uno de los tres grupos.
Para cada grupo existe un paso previo a realizar antes de la oportuna *Petición de Cambio*, por parte del departamento de explotación del cliente y poder, de este modo, determinar los cambios que va a sufrir el Sistema :

- Simulación del despliegue de la *Baseline* seleccionada sobre el Sistema o grupo de Sistemas que van a ser actualizados.

3.2.1. Sistemas Linux (Red Hat, SuSE)

Para poder parchear los entornos Linux del cliente se propone el Plan de acción siguiente:

- **Realizar Copia de Seguridad del Sistema** – Esta operativa la realizará el cliente antes de la hora planificada del despliegue y servirá como una de las medidas de contingencia necesarias para restaurar el Sistema en caso de que fuese necesario. Si el Sistema sufriese alguna corrupción de datos o se detectara una inestabilidad tras la realización del despliegue, se intentaría restaurar a partir de esta copia.
- **Ejecutar *cfg2html*** en el *host* y enviar el fichero al repositorio local. Mediante esta acción podremos guardar los datos de la configuración local del Sistema, para comprobar si se ha realizado alguna modificación tras el despliegue.
- **Parada de los servicios.**
- **Rebote inicial del sistema** – Esta acción se realiza siempre antes de realizar cualquier cambio importante en el Sistema y que puede afectar al Servicio, de este modo nos aseguramos que el Sistema estaba correcto antes de la realización del cambio.
- **Actualización según Service Pack** .
- **Operación Finalizada** .

Para poder desarrollar este plan de acción en *Sun Connection Satellite*, se ha generado un *script* que ejecuta de forma automática *cfg2html*, una herramienta que se instala en los Sistemas Linux y que extrae toda la información de la configuración del mismo, este será ejecutado como tarea previa al rebote inicial del sistema y al parcheo final. Los pasos que se deberán seguir para realizar este plan de acción en Sun Connection Satellite serán:

- Realizar un informe del Cumplimiento de *Service Pack* de Red Hat, que junto a la Gestión de Cuenta asignada al cliente por parte de Sun, se considere el adecuado (Ver Apéndice 1). Una vez generado el Informe, ejecutaremos el trabajo de despliegue del cumplimiento del *Service Pack* en modo simulación, lo que nos permitirá conocer los cambios que se van a producir en el sistema y descargar en Cache del Satélite todos aquellos paquetes que van a ser instalados.

- Una vez haya finalizado el trabajo de simulación y si no se ha producido ningún fallo, podremos ejecutar de nuevo el trabajo en modo despliegue como tarea final de un trabajo que estará formado por las tres tareas siguientes :
- Ejecutar cfg2html
- Realizar reinicio del sistema
- Aplicar los cambios para cumplir el *Service Pack* escogido.

De este modo tras la finalización del trabajo, obtendremos el Sistema cumpliendo el *Service Pack* escogido y siguiendo el proceso descrito al inicio.

3.2.2. Sistemas Solaris (Solaris 8, Solaris 9, Solaris 10)

Para poder parchear los entornos Solaris del cliente se propone el Plan de acción siguiente:

- **Realizar Copia de Seguridad del Sistema** – Esta operativa se realizará tal y como se ha detallado para los Sistemas Linux.
- **Ejecutar explorer reducido** en el host y enviar el fichero al repositorio local. Mediante esta acción podremos guardar los datos de la configuración local del Sistema, para comprobar si se ha realizado alguna modificación tras el despliegue. *explorer* es una herramienta instalable que integran los sistemas Solaris, con la finalidad de poder extraer toda la información de configuración y logs del Sistema.
- **Parada de los servicios** .
- **Partir disco de *mirror*** (/; /usr; /var; /opt) – Esta es una operación habitual en los sistemas Solaris cuando se va a realizar alguna modificación importante en el Sistema Operativo. Realizando esta operación, conseguimos que una de las dos vías de mirror o Raid-1 del Sistema permanezcan tal y como estaban antes de la modificación. En caso de cualquier problema durante la actualización en la vía que está siendo actualizada podemos recuperar el sistema, iniciándolo de esta otra vía.
- **Rebote inicial del sistema** – Esta acción se realiza siempre antes de realizar cualquier cambio importante en el Sistema y que puede afectar al Servicio, de este modo nos aseguramos que el Sistema estaba correcto antes de la realización del cambio.

- **Parcheo según Baseline.**
- **Regenerar de nuevo el disco** de *mirror* o programar un trabajo para esta finalidad.
- **Operación Finalizada** .

Para poder desarrollar este plan de acción en *Sun Connection Satellite*, se han generado varios scripts para ejecutar el explorer y copiarlo a un repositorio local, así como para partir el disco de *mirror* y preparalo para el inicio de sistema sin gestor de volúmenes (*desencapsularlo*), en caso de que por motivos de tiempo tengamos que realizar una marcha atrás alternativa a la que se propone mediante Sun Connection Satellite (ver Apéndice 3); la ejecución de estos scripts se realizarán previos al parcheo final siguiendo los siguientes pasos:

- Seleccionar la *Baseline* que junto al equipo de *Gestión de Cuenta* asignado al cliente se ha considerado el adecuado y generar el perfil que servirá para desplegarlo en modo simulación, de este modo nos permitirá bajar todos los parches que se van a aplicar a los diferentes sistemas a la Cache del Sistema Satélite (ver Apéndice 2).

- Una vez haya finalizado el trabajo de simulación y si no se ha producido ningún fallo, podremos ejecutar de nuevo el trabajo en modo despliegue como tarea final de un trabajo que estará formado por cinco tareas :

- Ejecutar explorer
- Copiar explorer a repositorio local
- Partir y desencapsular disco de "espejo"
- Realizar reinicio del sistema
- Parchear según Baseline escogida.

De este modo tras la finalización del trabajo, obtendremos el sistema cumpliendo la *Baseline* escogida y siguiendo el proceso descrito al inicio.

3.2.3. Sistemas Solaris 10 con Zonas

Para poder parchear los entornos Solaris 10 con Zonas del cliente se propone el Plan de acción siguiente:

- **Realizar Copia de Seguridad del Sistema**
- **Ejecutar explorer reducido**
- **Parada de los servicios de las zonas locales.**
- **Partir disco de mirror** de la zona global (/; /usr; /var; /opt)**,** las zonas locales residen en Sistemas de ficheros que se encuentran en una *Storage Area Network* (SAN), a la que se accede mediante una red de fibra totalmente redundante e incorpora las medidas de seguridad necesarias como para evitar implícitamente la corrupción o pérdida de datos.
- Generar **Plan de marcha atrás rápido para las zonas locales** – Debido a la integración en la SAN de los Sistemas de Ficheros de las zonas locales, la única medida de contingencia que se ha determinado, ha sido realizar una copia de seguridad de las zonas locales.
- **Rebote inicial del sistema** – Esta acción se realiza siempre antes de realizar cualquier cambio importante en el Sistema y que puede afectar al Servicio, de este modo nos aseguramos que el Sistema estaba correcto antes de la realización del cambio.
- **Parcheo según *Baseline* de la zona global.**
- **Regenerar de nuevo el disco de *mirror*** o programar un trabajo para esta finalidad.
- **Operación Finalizada** .

Para poder desarrollar este plan de acción en Sun Connection Satellite, se han generado varios scripts para ejecutar el explorer y copiarlo a un repositorio local, así como para partir el disco de mirror y desencapsularlo, en caso de que por motivos de tiempo no se pudiese realizar una marcha atrás normal (ver Apéndice 3); la ejecución de estos scripts se realizarán previos al parcheo final siguiendo los siguientes pasos:

- Seleccionar la *Baseline* que junto al equipo de Gestión de Cuenta asignado al cliente se ha considerado el adecuado y generar el perfil que servirá para desplegarlo en modo simulación en la zona global, de este modo nos permitirá bajar todos los parches que se van a aplicar a los diferentes sistemas globales a la Cache del Sistema Satélite (ver Apéndice 2).

- Una vez haya finalizado el trabajo de simulación en la zona global y si no se ha producido ningún fallo, podremos ejecutar de nuevo el trabajo en modo despliegue como tarea final de un trabajo que estará formado por cinco tareas:

- Ejecutar explorer
- Copiar explorer a repositorio local
- Partir y desencapsular disco de mirror
- Realizar reinicio del sistema
- Parchear según Baseline escogida.

De este modo tras la finalización del trabajo, obtendremos el sistema cumpliendo la *Baseline* escogida y siguiendo el proceso descrito al inicio.

Para conseguir que cuando Sun Connection Satellite parchee la zona global todas las zonas locales se parcheen una tras otra, se ha modificado el comando *patchadd* de la zona global, de forma que cuando se hace una llamada a este comando con la opción -G, este se obvia y se aplica el parche a todas las zonas, consiguiendo el comportamiento acordado junto al cliente.

Como medida de contingencia previa al parcheo de las zonas locales, se ha generado un script (ver Apéndice 4) que realiza una copia de las zonas a un repositorio local, tras las últimas pruebas realizadas, se detectó que la copia de cada zona duraba alrededor de 60 minutos, haciendo inviable lanzar este script como Pre-acción al parcheo, por lo que se recomienda que se lance con un día de antelación o bien el propio cliente realice una Copia de Seguridad de cada una de las zonas locales.

4. Planificación de tiempo y recursos del Proyecto

4.1. Recursos

Los recursos definidos para la ejecución del trabajo son los siguientes:

Empresa	Función
Cliente	Responsable del Proyecto
Cliente	Especialista de la plataforma
Sun Microsystems	Project Manager
Sun Microsystems	Arquitecto y Resp. técnico
Sun Microsystems	Ingenieros de sistemas

Tabla 3 – Recursos

Si bien podrán participar otras personas que se necesitarán de forma esporádica, las personas que se han nombrado son los principales miembros del equipo.

Dentro de los recursos que se requieren para desarrollar este proyecto, el desarrollado por mi ha sido el de Arquitecto y Responsable técnico para la implementación de la solución propuesta.

4.2. Fases de proyecto

Para implementar el proyecto en la Infraestructura Global del cliente, se ha considerado dividirlo en dos fases, bien diferenciadas teniendo en cuenta los objetivos que quieren cubrirse en cada una de ellas:

- **Fase I:** Implantación del producto
- **Fase II:** Parametrización de Sun Connection

4.1.1.1. Fase I: Implantación del producto

Definición de Arquitectura de despliegue.

Por parte de Sun se dará al cliente una relación lo mas completa posible de los pre-requisitos Hardware y Software de los Sistemas sobre los que se centra el proyecto (ver Apéndice 6).

Por parte del cliente se pasará al Ingeniero de Sun una relación completa de sistemas que participarán en el proyecto indicando toda la información relevante para el diseño de la solución:

> Sistema Operativo y Arquitectura (Solaris, Linux, sparc, x86, zLinux etc), ubicaciones, estructura de red (Corta fuegos, proxy, WAN, LAN etc) y capacidad, problemas de seguridad, análisis de las aplicaciones que están funcionando en estos sistemas, procedimientos de mantenimiento, como se está realizando el parcheado actualmente, tareas de administración, usuarios de sistema, repositorios de software, procedimientos de despliegue de nuevos sistemas y nuevas aplicaciones, relación entre los sistemas , dependencias con otros sistemas y cualquier información que se considere relevante para el diseño de la Arquitectura global de la solución.

El objetivo de este punto es obtener una visión lo más completa posible del entorno donde se implantará *Sun Connection Satellite*, así como una visión general de la arquitectura del cliente. Estos sistemas han de cumplir los requisitos necesarios dependiendo su función tal y como se detallaba en el apartado de Arquitectura.

Instalación *Sun Connection Satellite*

Por parte del cliente se realizará la entrega de las licencias al Ingeniero de Sun. Estas licencias tienen una validez de 365 dias y se envían desde Sun Microsystems al Responsable de proyecto o persona de contacto del cliente.

Sun realizará un Chequeo de la instalación del producto en los sistemas implicados. Si durante el chequeo se encontrase algún punto que no permite la instalación se procederá a informar al cliente de la situación. Se espera por parte del cliente respuesta y solución de la situación así como fecha máxima de resolución o bien un sistema alternativo para la realización del proyecto con características similares a las definidas en el apartado de Arquitectura.

Instalación del Servidor Satélite de *Sun Connection Satellite*.

Instalación de Agentes. (25 sistemas/Zonas) en los sistemas del documento de Arquitectura.

Chequeo funcional de comunicación entre los agentes y el Satélite, así como entre éste y la Base de datos de Conocimiento Universal y a los diferentes distribuidores (Sun, Red Hat, Novell y SunFreeware).

4.1.1.2. Fase II: Parametrización de Sun Connection Satellite

Definición detallada de requerimientos para tres grupos de sistemas

Se ha definido por parte del cliente y Sun una tipología de parcheado aplicable como máximo a tres grupos de sistemas. Se entiende por tipología todas aquellas tareas de parcheado que solo se pueden aplicar a un conjunto de sistemas en contraposición a otro conjunto.

Recomendamos por parte de Sun la elección diferencial de Sistemas Solaris con Zonas, Sistemas Solaris y Sistemas Linux. En cualquier caso el tratamiento será solo de tres grupos al máximo nivel de detalle. Permitiendo estos tres grupos el parcheado completo de los 25 sistemas escogidos para este proyecto, siempre que estos reúnan las características de estos tres grupos o las definidas por el cliente en el momento de la implantación.

Durante la definición de estas tres estrategias será necesaria la participación activa por parte del cliente en la validación de las mismas y coherencia con el entorno real de explotación.

Implantación de las tres políticas

Con el documento de descripción de las políticas de parcheado se procederá al despliegue completo de estas mediante la parametrización de Sun Connection Satellite. Para la realización de esta tarea será necesario disponer de como mínimo un sistema de cada uno de los grupos definidos en el apartado 3.2 y poder realizar las pruebas adecuadas.

Estos sistemas tendrán que estar a disposición de los Ingenieros de Sun en el proyecto en horario laboral para la realización de las pruebas necesarias.

Una vez refinado e implantado el proceso de las tres tipologías se realizará una validación por parte de los responsables de proyecto de la solución técnica implantada en relación a los requisitos pactados en los documentos anteriores.

Jornadas para parcheado

Una vez validado el procedimiento de parcheado se procederá a la realización de unas jornadas de parcheado cuyo objetivo es la transferencia del entorno a los responsables de su gestión y la extracción de conclusiones para la evolución del proyecto.

Será necesaria como mínimo una persona del cliente para la realización de estas jornadas de parcheado, para que valide la implantación detallada de requerimientos ya definida, así como el éxito o fracaso de las pruebas realizadas.
El objetivo de estas jornadas no es el parcheado de los sistemas sino la validación de la operatividad del producto en el entorno del cliente.

Este trabajo se puede realizar fuera del horario laboral con un máximo de 5 jornadas de dedicación por parte de un Ingeniero de Sun Microsystems.

Durante estas jornadas se realizara:

1. Explicación general del producto y transferencia de información sobre la instalación.
2. Demostración práctica de las funcionalidades relacionadas a la consola gráfica y el entorno en general.
3. Parcheado de los tres casos de grupos:

Revisión por parte del cliente del entorno y momento del parcheo, Copia de seguridad, control de cambios y procesos implicados en el mismo.

Ejecución por parte de *Sun Connection Satellite* de las políticas definidas, primero en modo simulación y entregado un reporte del estado y resultado de la simulación.

Una vez validado por parte del cliente se procederá a la ejecución en modo real.

Criterios de éxito:

Parcheado en base a una línea Base:
Ejemplo detectar problemas de seguridad en los sistemas y desplegar los parches que los solucionan, chequeando todas las dependencias relacionadas.

Marcha atrás y recuperación:
Realizar la marcha atrás, mediante *Sun Connection Satellite*, de algunos cambios y demostrar que el sistema es capaz de deshacer, tras la ejecución de un parcheo, este parcheo de forma íntegra o única y exclusivamente un parche en concreto. Esta marcha atrás la realizará el sistema mediante la desinstalación de los parches o paquetes que se hayan instalado durante la actualización, por lo tanto se deberá tener en cuenta que es una marcha atrás lenta para muchos de los caso.

Uso de pre y Post Scripts:
Utilización de scripts pre-parcheado y post-parcheado durante la ejecución de las políticas demostrando que diferentes tareas se pueden realizar en función del éxito de los pre-scritps (si no es posible realizar una pre tarea se para el parcheado)

4.3. Planificación temporal

El siguiente Planificación temporal es estimado, teniendo en cuenta las diferentes fases técnicas planteadas y que se estimará el comienzo de las mismas a partir del inicio de este proyecto. Las jornadas estimadas son aproximadas y no garantizan que el tiempo dedicado a cada unas de las fases sea el propuesto:

Fase	Jornadas propuestas	Fecha de Inicio Planificada	Fecha Fin Planificada	Fecha Fin Actualizada
Fase I: Definición de Arquitectura de despliegue e Instalación	5-7	1 de Abril	4 de Abril	11 de Abril
Fase II: Parametrización de Sun Connection	12-15	14 de Abril	5 de Mayo	23 de Mayo
Jornadas de parcheado	5	30 de Mayo	1 de Junio	29 de Junio

Tabla 4 – Planificación temporal

5. Referencias

Ref	Referencia
1	Portal de documentación de todos los productos de Sun Microsystems. [web: http://docs.sun.com]
2	Portal en el que se detalla todas las Herramientas, que Sun Microsystems dispone ,para la Gestión Integral de Centros de Cómputo. [web: http://www.sun.com/software/systemsmanagement/index.jsp]
3	Portal de *Red Hat Networks*. [web: https://www.redhat.com/rhn/]
4	Wikipedia generada para la documentacion en información de *Opsware*. [web: http://en.wikipedia.org/wiki/Opsware]
5	Wikipedia *Infomation Technology Infrastructure Library* (ITIL). [web: http://en.wikipedia.org/wiki/ITIL]

6. Apéndice 1 – Realización del Informe de cumplimiento de Service Pack

Para poder realizar el Informe de cumplimiento de *Service Pack* en los Sistemas Linux que forman el **grupo 1**, se deberán seguir una serie de pasos que se detallan a continuación. Teniendo en cuenta que la mayoría de estos pasos se realizarán mediante la consola gráfica de Sun Connection Satellite, se ha considerado oportuno incluir los pantallazos de cada una de las acciones a realizar:

- Para realizar el informe de cumplimiento de Service Pack, sobre cualquiera de los Sistemas que formen el **grupo 1** , se deberá acceder al Menú de Herramientas y Seleccionar Informes.
- Dentro del submenú de Informes se deberá seleccionar el Informe *Service Pack Compliance,* tal y como se puede observar en la figura 4.

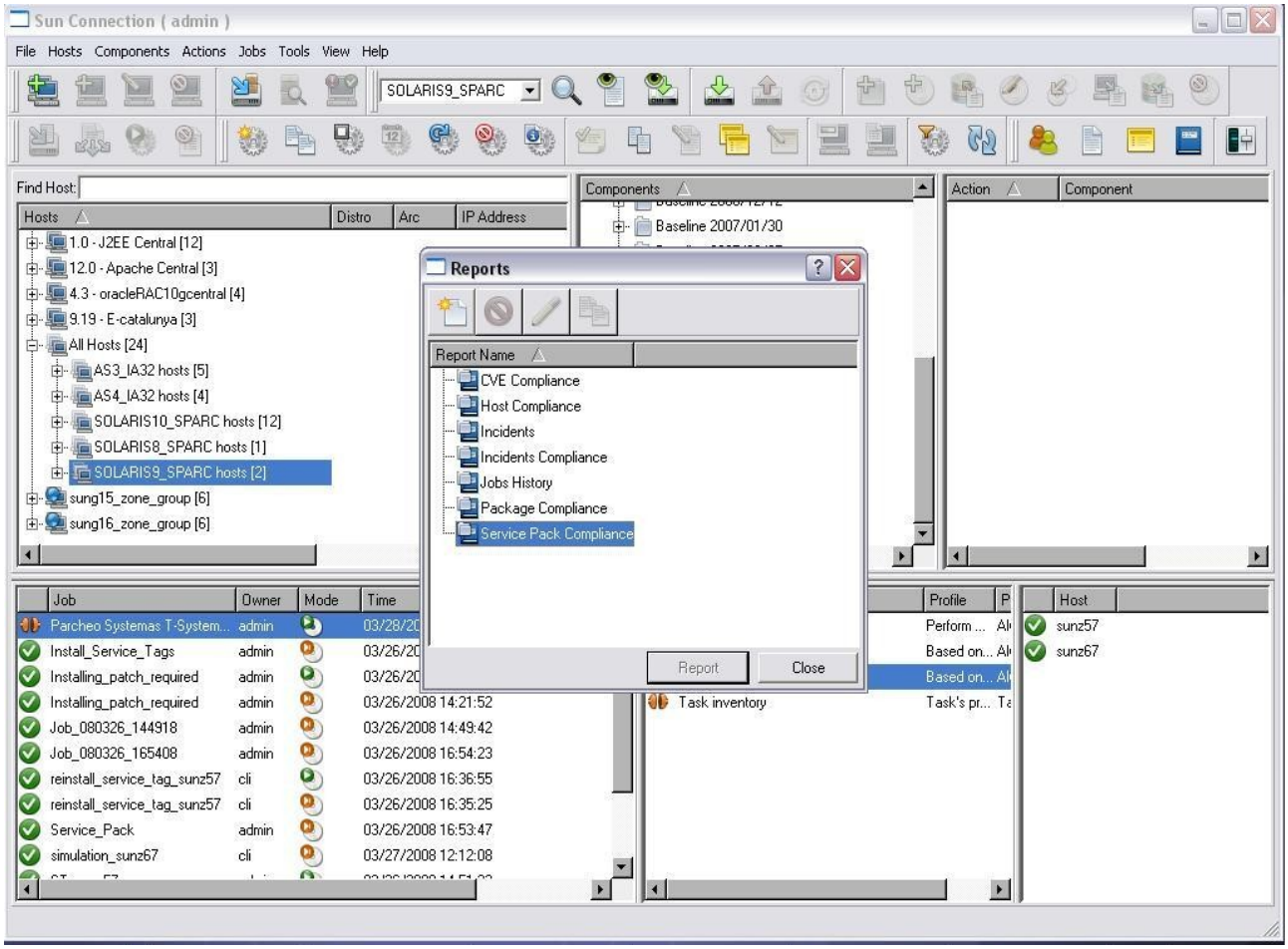

figura 4 – Selección Service Pack Compliance

- Una vez hayamos seleccionado el tipo de informe a realizar, deberemos seleccionar sobre que Sistema, o grupo de Sistemas, queremos lanzar la consulta de Cumplimiento del Service Pack de Red Hat., tal y como podemos observar en la figura 5.

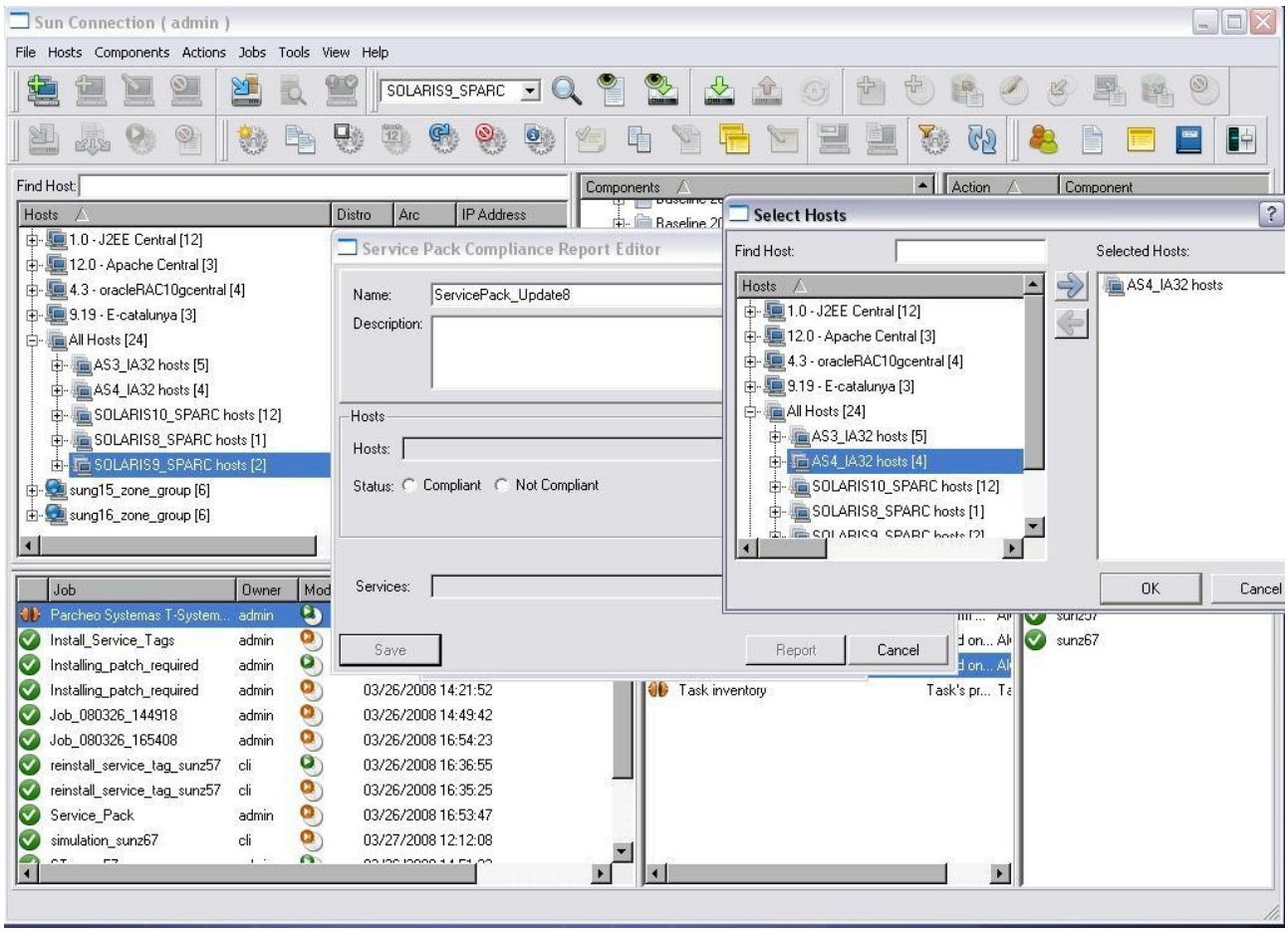

figura 5 – Selección de Sistema o grupo de Sistemas

- Tras haber seleccionado los Sistemas sobre los que queremos realizar el Informe, debemos seleccionar que Service Pack o actualización de Red Hat queremos estudiar el Cumplimiento, tal y como podemos observar en la figura 6. De este modo, Sun Connection Satellite, realizará un informe con todos los cambios necesarios para que los Sistemas se encuentren al nivel del Service Pack seleccionado, tal y como podemos observar en la figura 7.
- Una vez estudiemos en detalle los cambios que se van a realizar en cada uno de los Sistemas, podremos lanzar un trabajo con los cambios que queremos que se realicen, tal y como se observa en la figura 8. Este trabajo podremos lanzarlo en modo simulación, para descargar los paquetes de la web de Red Hat a la *Cache* del Servidor Satélite de Sun Connection y comprobar el proceso de instalación antes de su instalación definitiva, o bien, podremos ejecutarlo en modo despliegue (ver figura 9), de modo que el trabajo instalaría directamente los paquetes de Red Hat en los equipos justo después de descargarlos de la Web, y en caso de que fuese necesario, recompilaría el *kernel.*

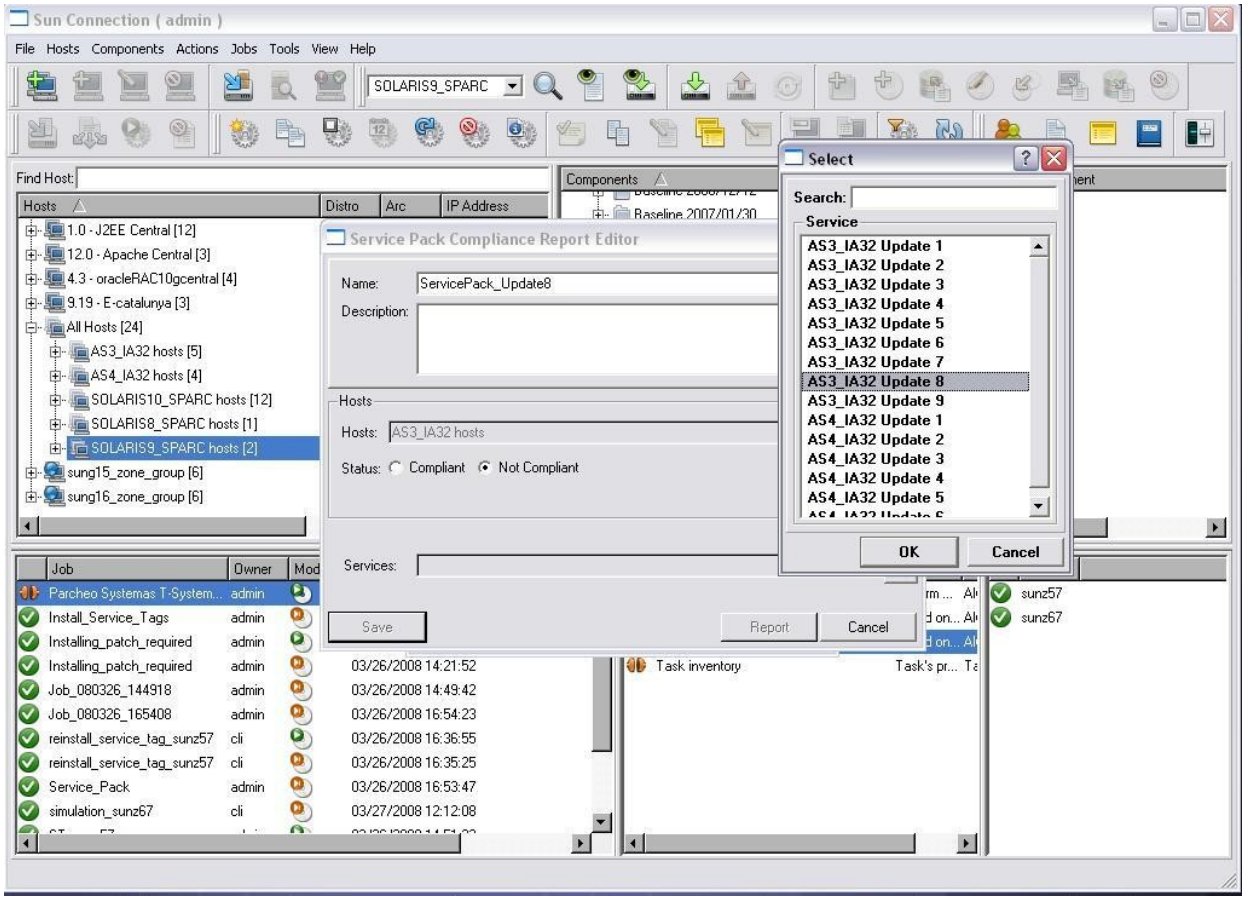

figura 6 – Selección del Service Pack not Compliance

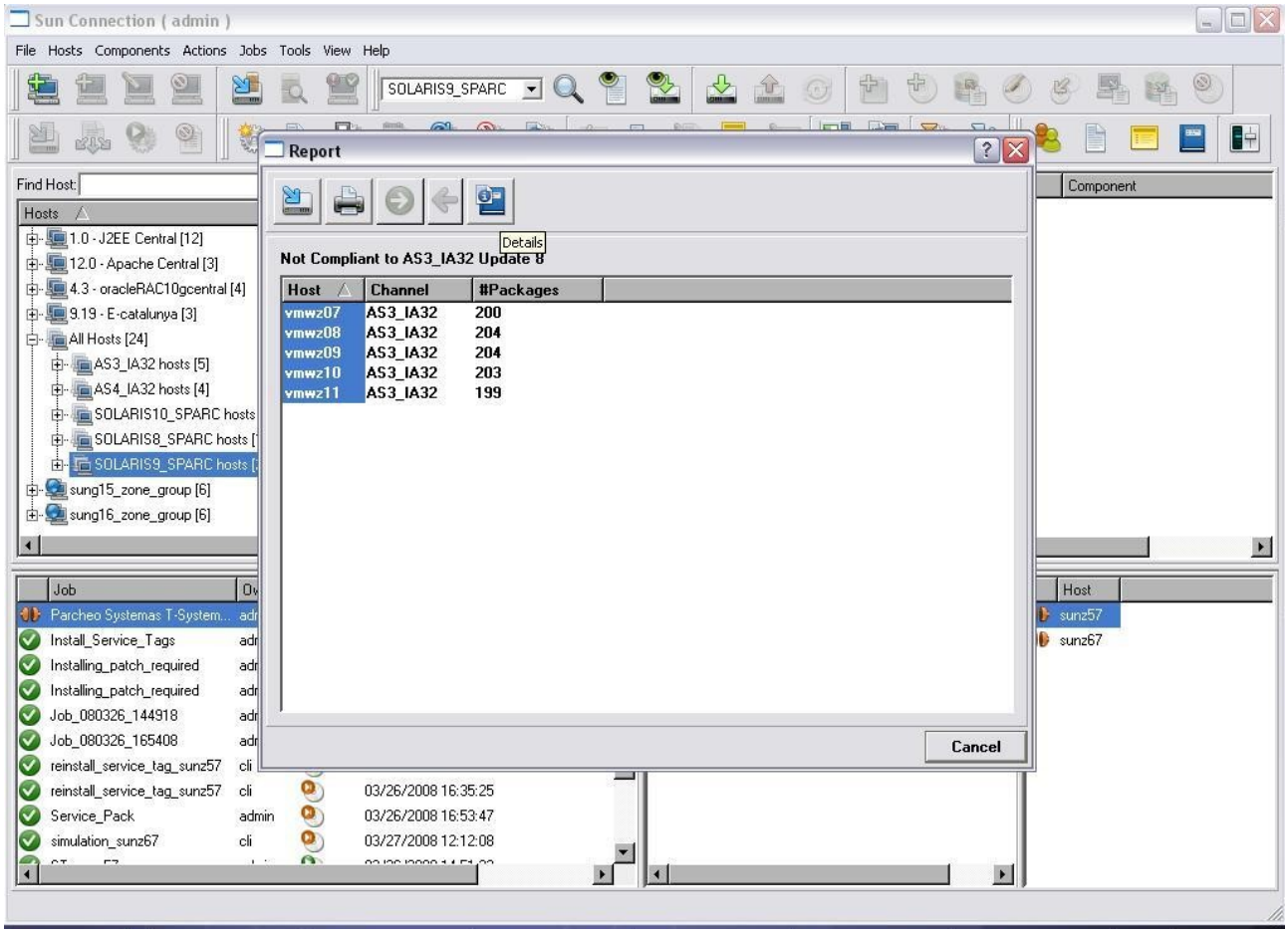

figura 7 – Selección del detalle de todos los Sistemas implicados

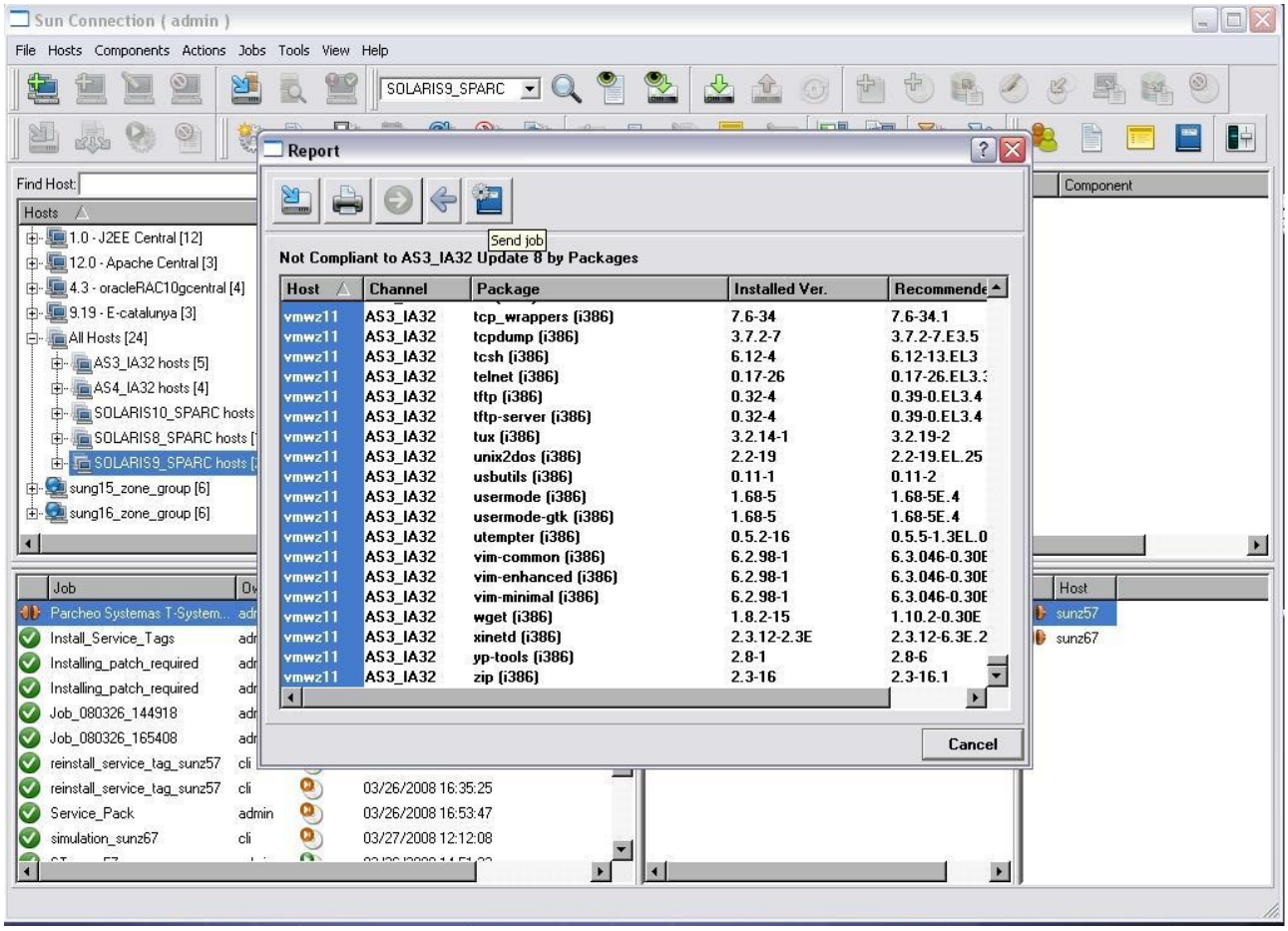

figura 8 – Selección de todos los cambios que se van a ejecutar en el trabajo de simulación

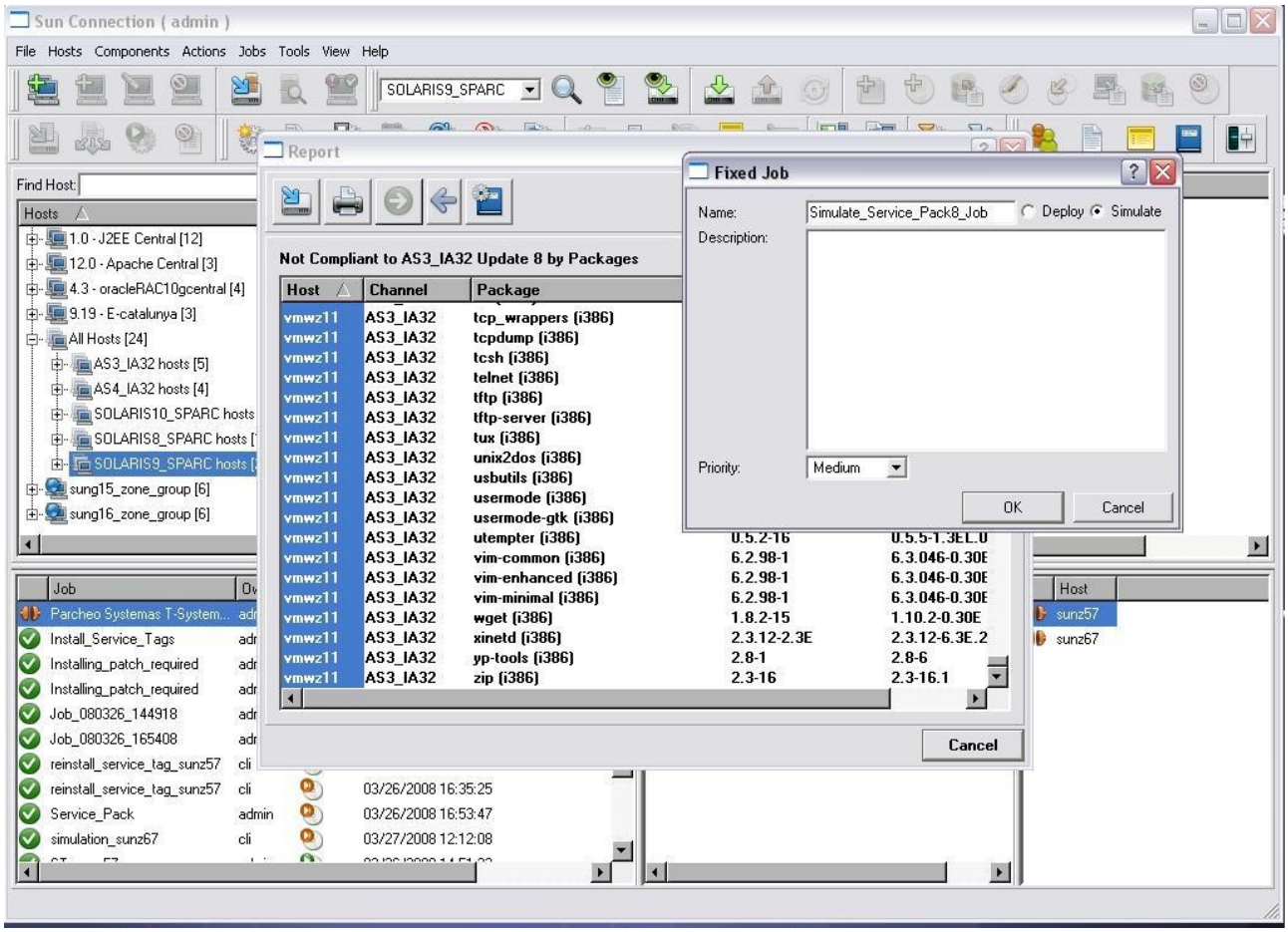

figura 9 – Ejecución en modo simulación del trabajo que va a instalar los paquetes necesarios par cumplir el nivel de Service Pack Selecciondo.

7. Apéndice 2 – Generación de un Perfil para el despliegue de una *Baseline* en Sistemas Solaris

Para poder realizar las tareas necesarias para que cualquiera de los Sistemas Solaris, que forman el grupo 2 y 3, cumplan con una *EIS Baseline*, se deberán seguir una serie de pasos que se detallan a continuación. Teniendo en cuenta que la mayoría de estos pasos se realizarán mediante la consola gráfica de Sun Connection Satellite, se ha considerado oportuno incluir los pantallazos de cada una de las acciones a realizar.
La *EIS Baseline*, como ya hemos detallado en apartados anteriores, es la Linea base que Sun libera mensualmente, con respecto al Software de parches y paquetes que solventan problemas conocidos en los Sistemas o bien aportan nuevas funcionalidades.

- Primero deberemos escoger, en la ventana de Sistemas, el Sistema o grupo de Sistemas de igual versión de Sistema Operativo Solaris sobre los que queremos desplegar la *Baseline.* Una vez hayamos escogido los Sistemas, en la ventana de componentes dentro del grupo de *Baselines,* deberemos escoger que *Baseline* queremos desplegar. Apretando al botón derecho del ratón la seleccionaremos como Requerida, tal y como se observa en la figura 20.

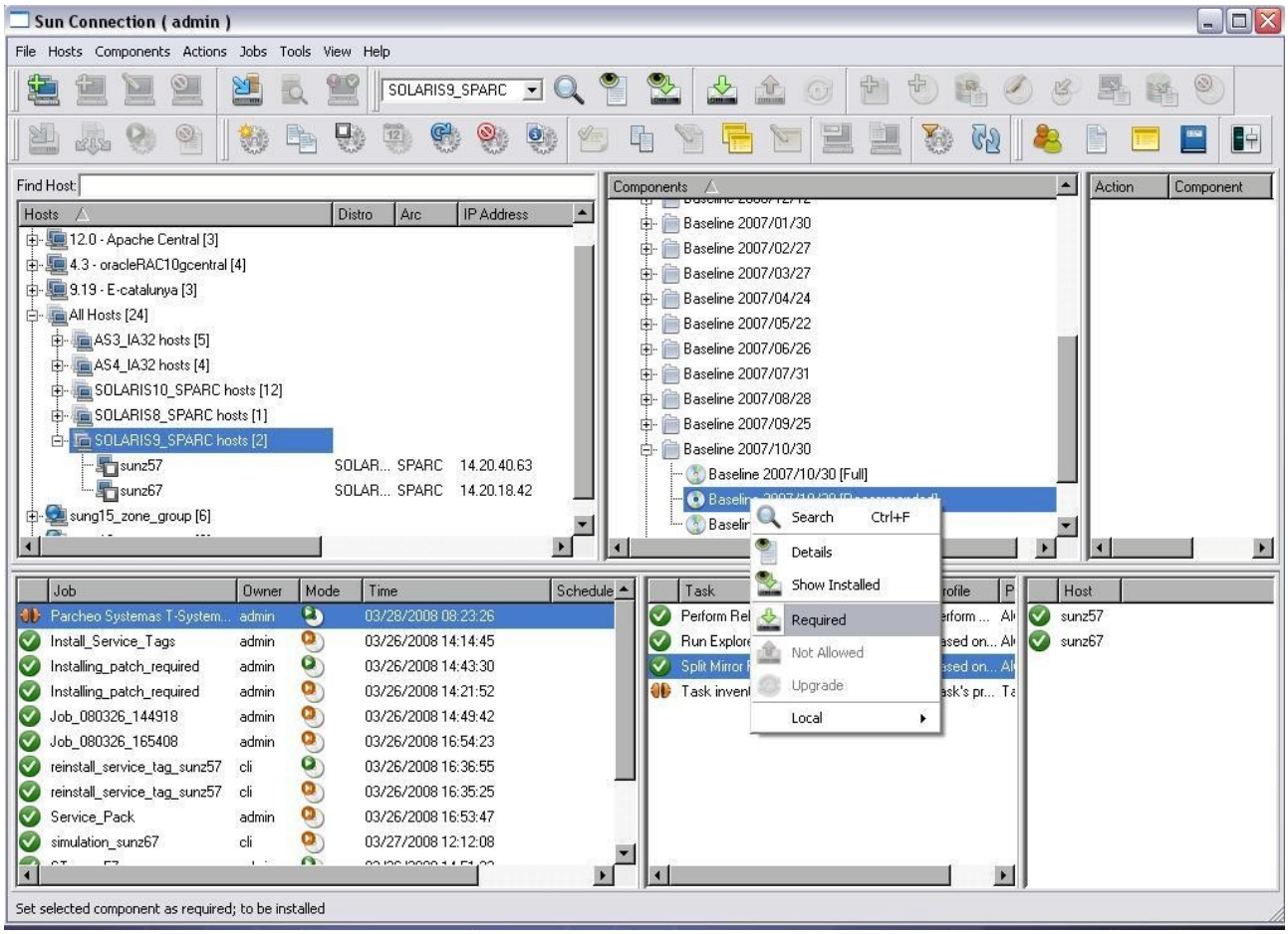

figura 20 – Selección de *Baseline* a aplicar como elemento requerido

- Cuando hayamos marcado la *Baseline* como Requerida, aparecerá en la ventana de la derecha, esta ventana se utiliza para lanzar trabajos denominados rápidos o de asignación directa, sin necesidad de generar perfiles complicados para la instalación de un componentes. Como vemos en la figura 21, seleccionando sobre el componente requerido, en esta ventana de asignación directa, podremos crear un perfil, para después desplegarlo y reutilizarlo en todos los Sistemas que queramos, o bien podremos lanzar directamente el trabajo para instalar la *Baseline* en los Sistemas deseados.

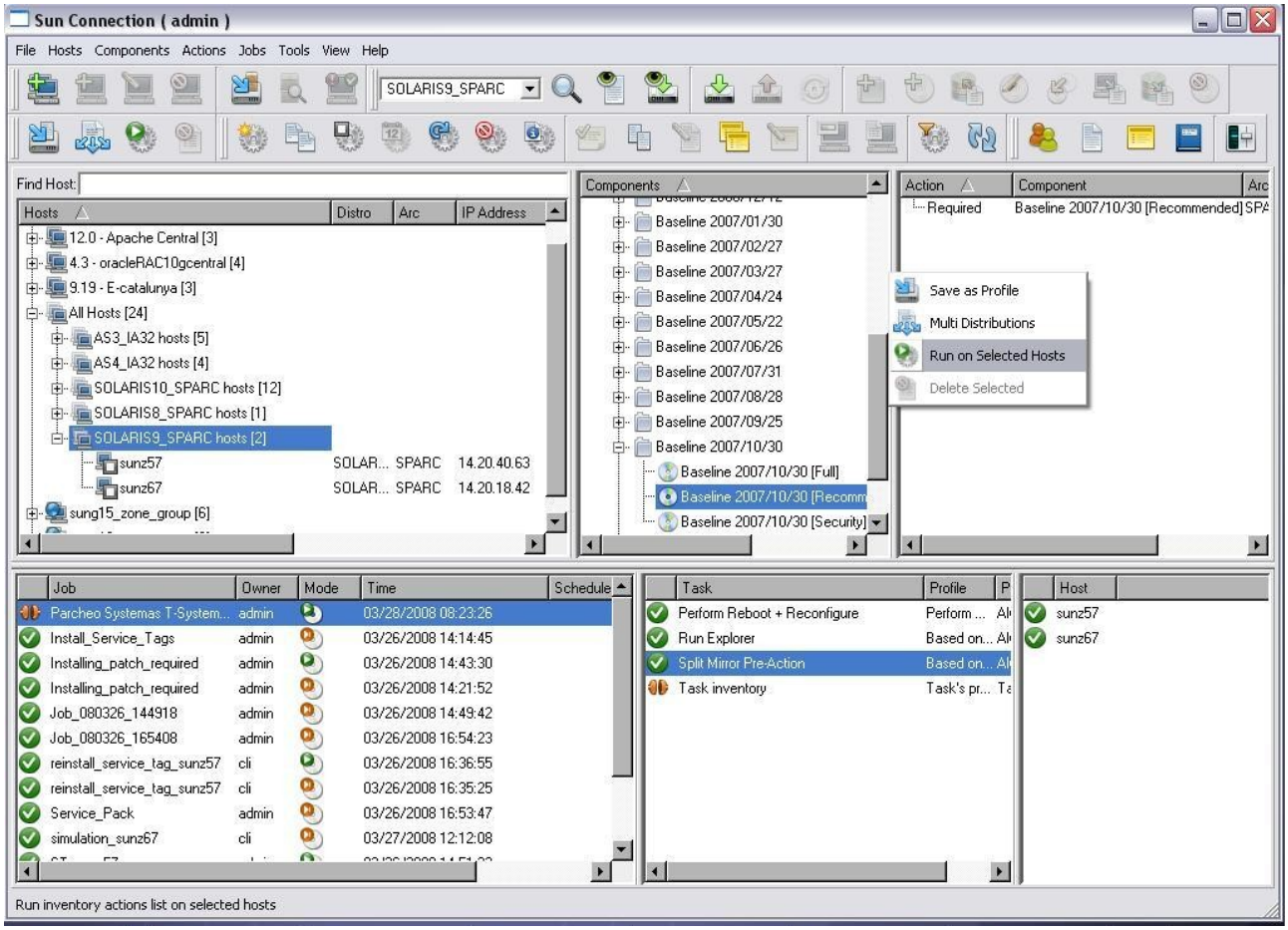

figura 21 – Selección de *Baseline* a aplicar como *Multi Distribución,* par aplicar el mismo nivel de parcheado a todas las distribuciones Solaris

- En nuestro caso seleccionaremos guardar como Perfil y en la Ventana de Perfil, le daremos el nombre deseado y apretaremos sobre el botón de mutlidistribución, para poder desplegar mediante el mismo Perfil la misma Linea Base de producto sobre cada una de las diferentes distribuciones Solaris que forman los grupos 2 y 3 (figura 22).

- Para después desplegar la *Baseline* seleccionada, generaremos un trabajo, seleccionando el perfil generado para este fin (figura 23), sobre los Sistemas que queramos mantener al mismo nivel de actualización (figura 24).

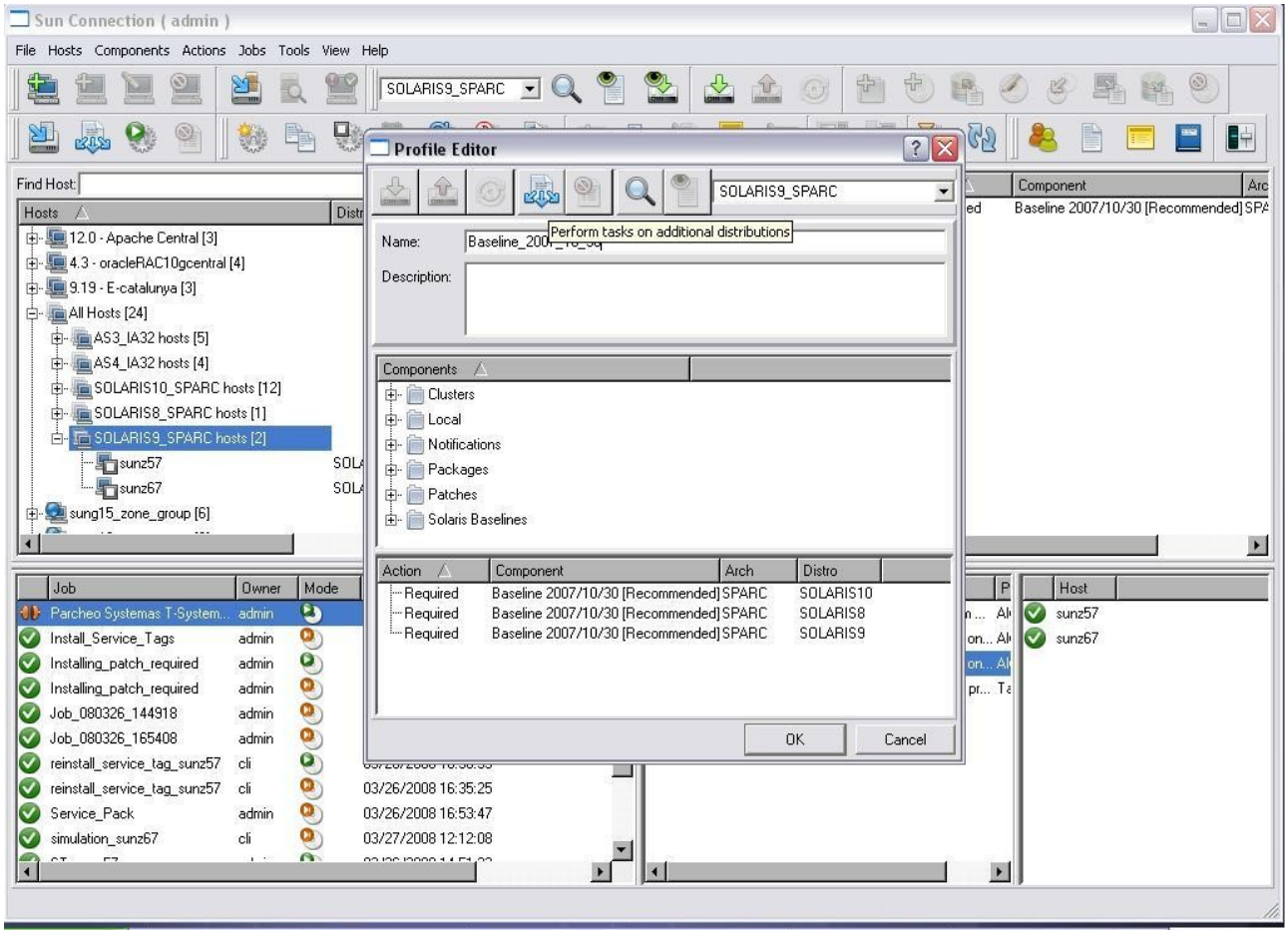

figura 22 – Guardar *Baseline* como Perfil

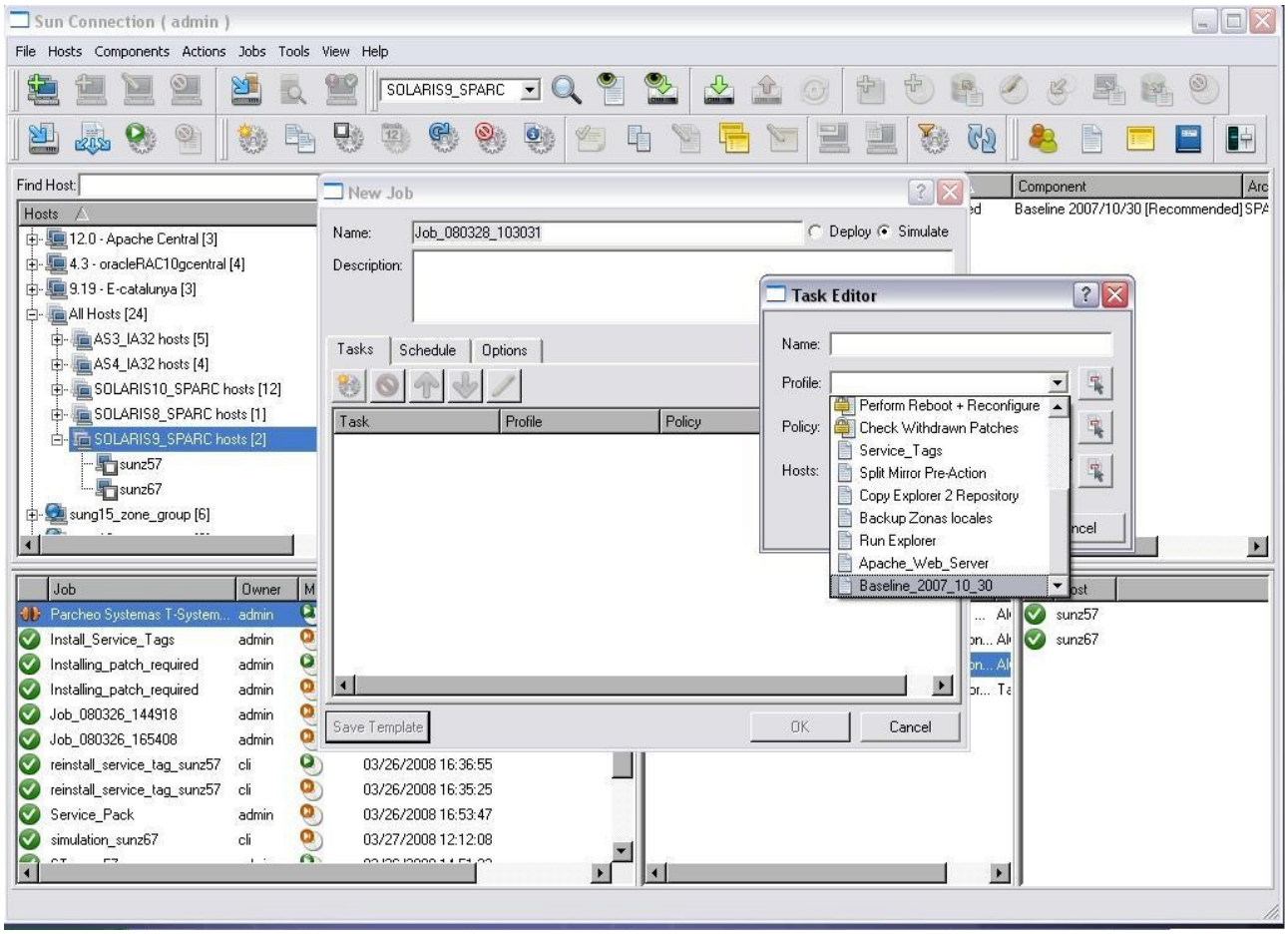

figura 23 – Realizar nuevo trabajo con Perfil de despliegue de *Baseline* como tarea única

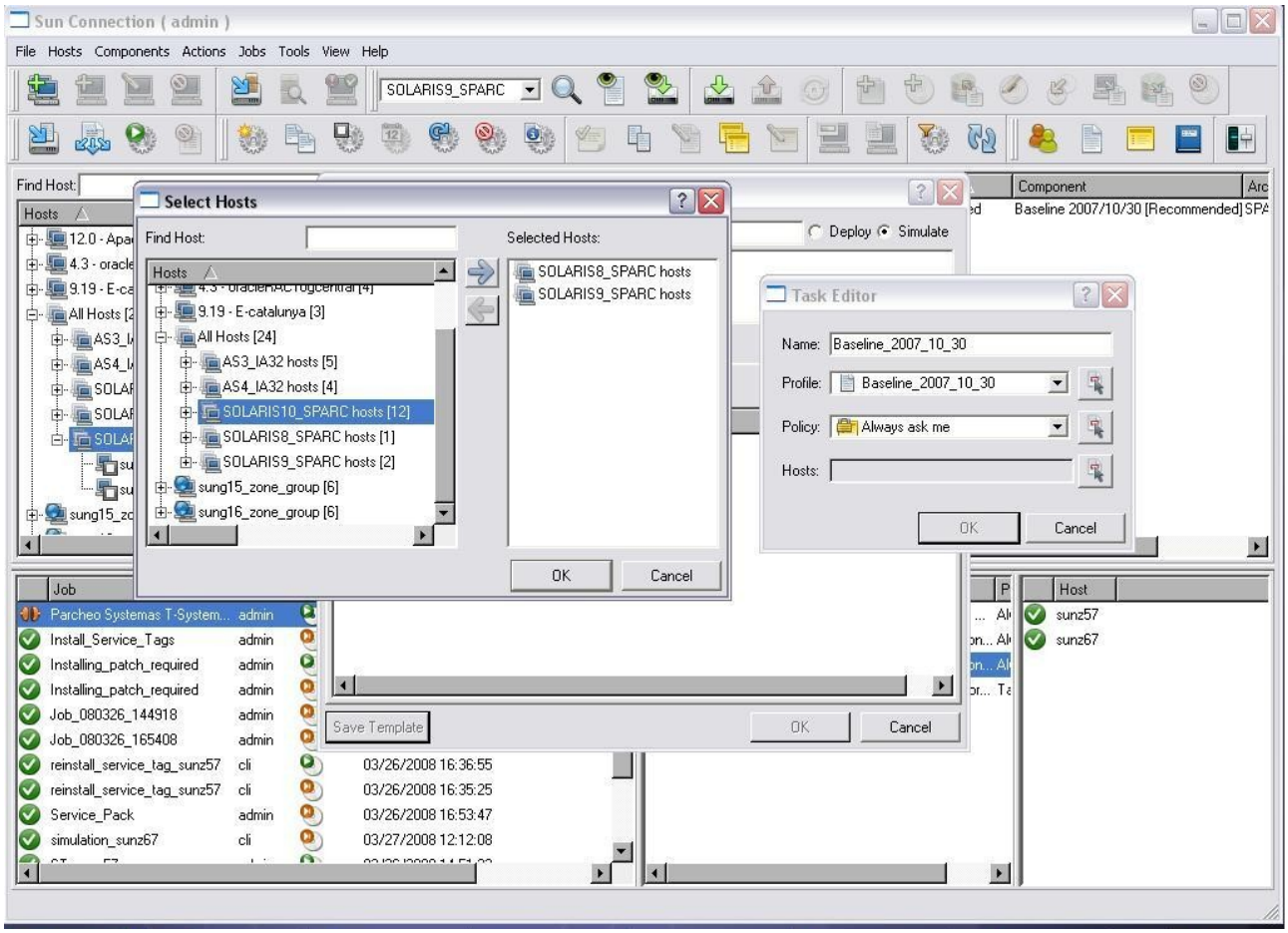

figura 24 – Selección del sistema o grupo de sistemas donde se va a aplicar el perfil en modo simulación.

8. Apéndice 3 – Marcha atrás (*Rollback*)

La marcha atrás se realizará en caso de que el cliente haya detectado algún problema en sus aplicaciones después de aplicar algún parche, paquete o cambio en cualquiera de los ficheros de configuración controlados por Sun Connection Satellite. Esta marcha atrás se podrá realizar a través de los medios que Sun Connection Satellite nos proporciona o bien a través de los medios habituales de iniciar el sistema a partir del disco de *mirror desencapsulado*, que previamente se ha preparado gracias al script generado para este fin.

En caso de que queramos realizar la marcha atrás mediante Sun Connection Satellite lo haremos siguiendo los pasos siguientes:

- Seleccionar el sistema sobre el que queremos realizar la marcha atrás y seleccionar la opción de comparar inventarios:

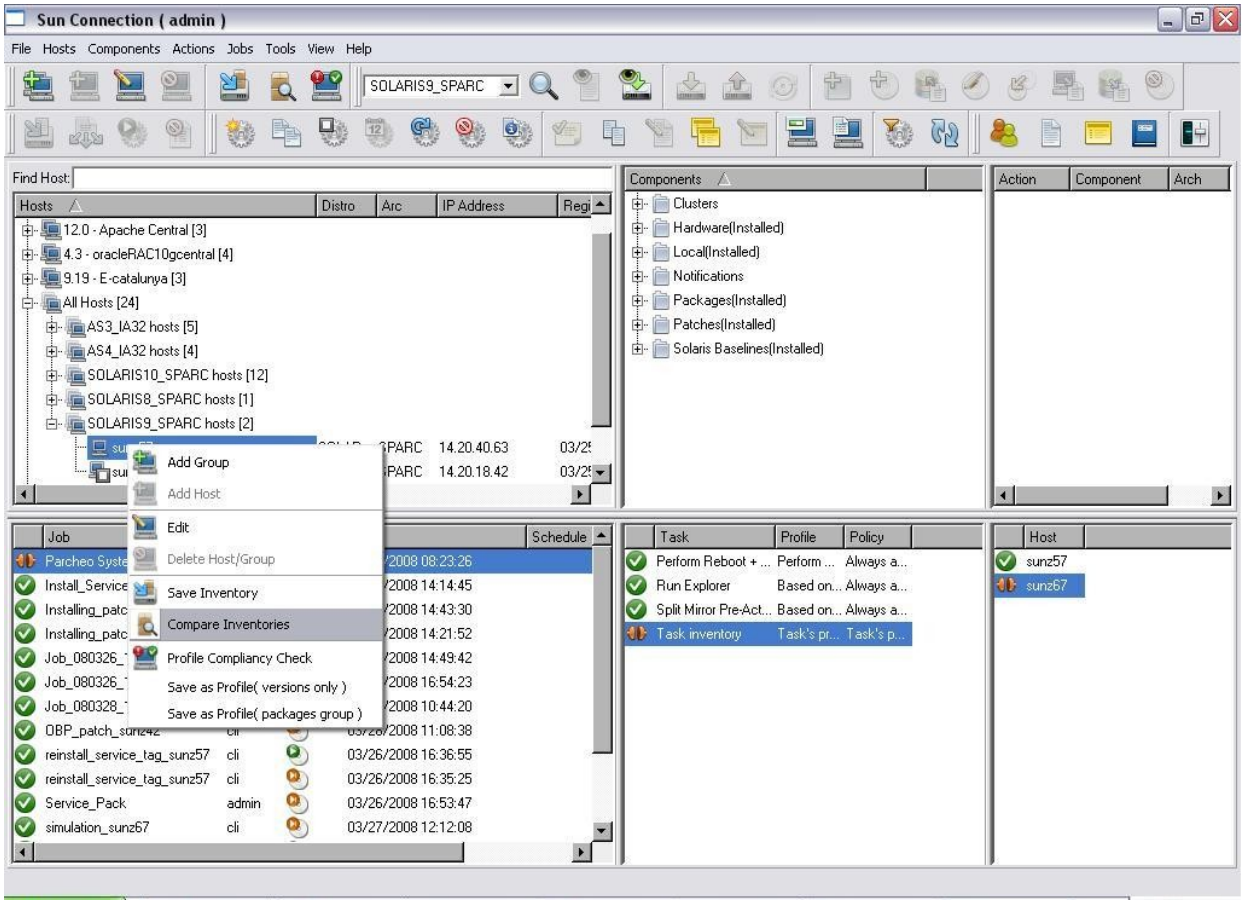

figura 25 – Comparar inventarios del sistema

- Compararemos el inventario actual (*target*) con el inventario que había antes de generar el cambio (*source*) y que se genera de forma automática antes de cada cambio. Como se puede comprobar en la figura 24, podremos comparar no sólo inventarios en diferentes estados del tiempo del propio sistema, sino también inventarios de diferentes sistemas, utilidad que nos dará la propiedad de desplegar o clonar sistemas fuente sobre cualquier otro sistema o grupo de sistemas destino.

- Una vez seleccionados los diferentes inventarios deberemos seleccionar que queremos comparar: *Software*, *Hardware*, Ficheros de Configuración... (si el cambio que se ha producido ha sido tras un parcheo, seleccionaremos *Software).* Tras la comparación obtendremos el resultado siguiente:

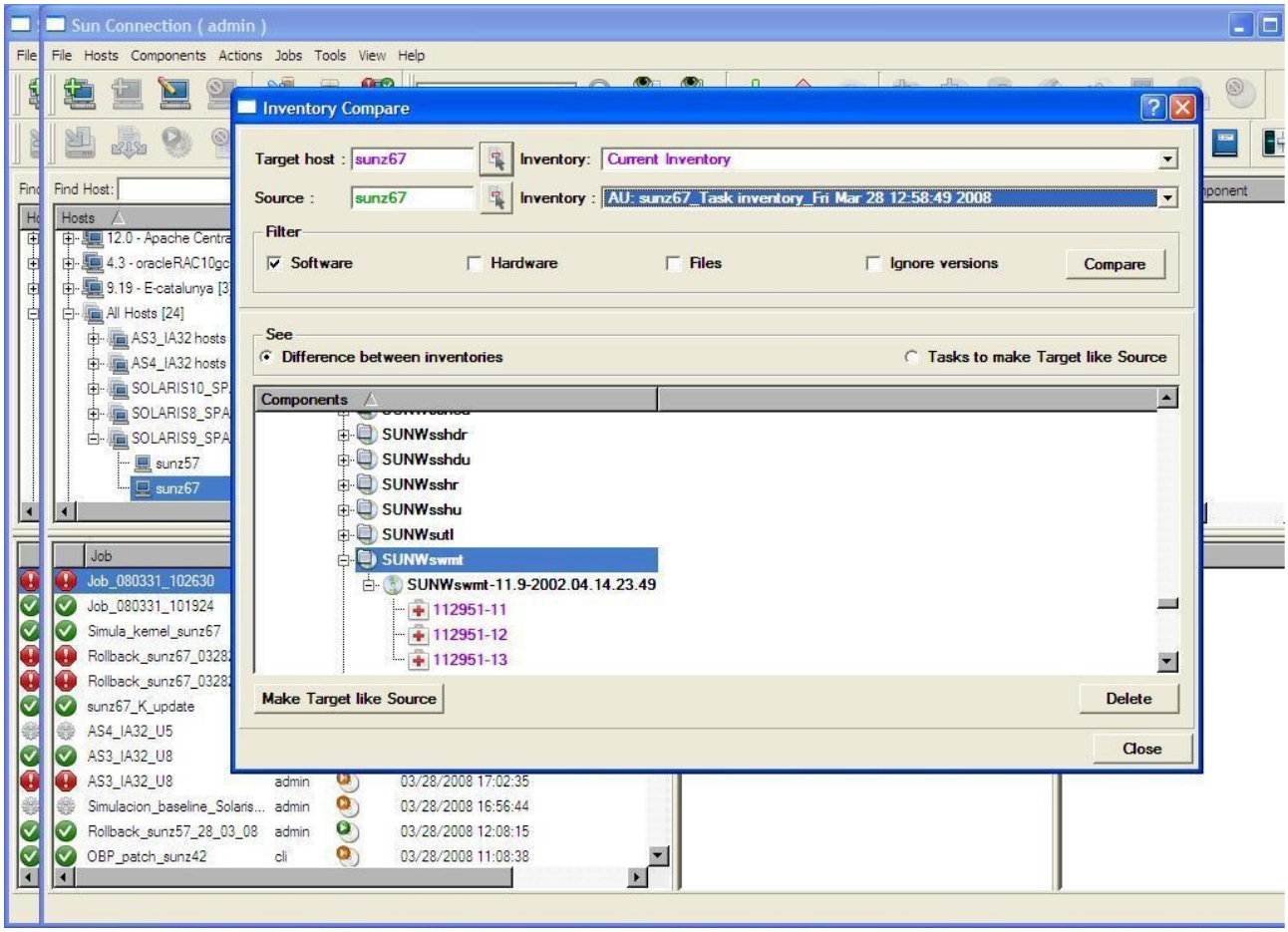

figura 26 – Resultado de comparar Inventarios

- Tras obtener el resultado, podremos lanzar un trabajo en modo despliegue o simulación mediante el botón *"Make target like Source"*, este trabajo realizará todas las tareas de instalación y desinstalación necesarias para llevar el inventario actual al inventario origen.

  Como nota importante a tener en cuenta, podemos destacar que en algunos casos nos podemos encontrar que un parche o paquete que se ha instalado durante el cambio, no puede ser desinstalado debido a la propia naturaleza del parche o paquete. En estos casos podremos seleccionar el parche y borrarlo mediante el botón *"Delete"* para eliminarlo de las tareas que se van a realizar durante la marcha atrás.

9. Apéndice 4 – Scripts generados para realizar las tareas Previas a un cambio

9.1. Partir y desencapsular Disco de Mirror (split_allges_mirror.ksh)

Este script es uno de los principales y más importantes implementados para este proyecto. La finalidad de este script es automatizar la medida de contingencia que normalmente un Ingeniero Sun, cuando parchea un Sistema Operativo, realiza como salvaguarda ante un Desastre o corrupción de datos durante el proceso normal de parcheo.
De este modo el script detecta automáticamente el disco de botado del Sistema, detecta automáticamente el gestor de Volúmenes que se está usando, Solaris Volume Manager (SVM) o Veritas Volume Manager (VxVM) y realiza las acciones necesarias para partir el disco de *mirror* y lo prepara para que pueda iniciar el sistema tal y como estaba antes del parcheo, deshabilitando el Gestor de Volúmenes que esté actuando, o como los Ingenieros Sun catalogamos como desencapsulado de disco. El script si acaba en fallo deberá retornar como valor de salida un 1 y sino un 0, ya que es así como lo entenderá Sun Connection Satellite.

1. Detectar Disco de boot
2. Detectar Gestor de Volúmenes
   - 2.1 Si no tiene gestor de volúmenes
     - 2.1.1. Sale con error
   - 2.2 Si el gestor de Volúmenes es SVM
     - 2.2.1. Si no tiene *mirror*
       - 2.2.1.1. Sale con error
     - 2.2.2 Si tiene *mirror*
       - 2.2.2.1. Parte el disco de *mirror* y lo deshabilita
       - 2.2.2.2. Chequea las particiones del disco de *mirror*
       - 2.2.2.3. Desencapsula el disco de *mirror*
   - 2.3. Si el gestor de Volúmenes es VxVM
     - 2.3.2 Si no tiene *mirror*
       - 2.3.2.1. Sale con error
     - 2.2.2 Si tiene *mirror*
       - 2.2.2.1. Parte el disco de *mirror* y lo deshabilita
       - 2.2.2.2. Chequea las particiones del disco de *mirror*
       - 2.2.2.3. Desencapsula el disco de *mirror*
   - 2.4 En cualquier otro caso
     - 2.4.1. Sale con error
3. Completa el fichero de logs y finaliza

```
#!/bin/ksh
#
# volume_split_mirrror  - provides an automatic way to split mirrors of a system, detecting
#                      the Volume Management in charge of this mirror and modifying
#                      files involved on the mirrored File System.
#                      volume_split_miror was created to help Sun Connection Satellite
#                      Agent to patch Solaris Systems.
#
#
# Aug 2007   - Felipe Herrera felipe.herrera@sun.com
#
#

#Variables
SBIN=/usr/sbin
BIN=/usr/bin
DATE=`$BIN/date '+%m%d%y'`
TIME=`$BIN/date '+%m%d%y_%H%M'`
RUNLEVEL=`/usr/bin/who -r | awk '{print $3}'`
OUTDIRBASE=/var/tmp/UCE
OUTDIR=/var/tmp/UCE/Split_$DATE
OUTFILE=$OUTDIR/split_mirror.log
OS=`$BIN/uname -r`
HOST=`$BIN/hostname`



install_bootblk ()
#
# Install boot block on mirror disk
#
#
```

```
{
  ROOTSLICE=$1
    echo " Installing boot block in /dev/rdsk/$ROOTSLICE"  >> $OUTFILE
    #$SBIN/installboot /usr/platform/`uname -i`/lib/fs/ufs/bootblk /dev/rdsk/$ROOTSLICE
    echo " Done " >> $OUTFILE

}



mount_etc_disk ()
#
# Mount Root File System of Mirrodisk in /UCE
#
#
{
 ROOTSLICE=$1
 if [ ! -d /UCE ]; then
        echo " Creating /UCE Directory"  >> $OUTFILE
        mkdir -p /UCE
        echo " Done " >> $OUTFILE
 fi

 echo " " >> $OUTFILE
 echo " Mounting Mirror root-Slice: /dev/dsk/$ROOTSLICE on /UCE"  >> $OUTFILE
 #$SBIN/mount /dev/dsk/$ROOTSLICE /UCE
 echo " Done " >> $OUTFILE

}


umount_etc_disk ()
#
# Umount Root File System of Mirrodisk in /UCE
#
#
{
 ROOTSLICE=$1

 echo " " >> $OUTFILE
 echo " Umounting Mirror root-Slice: /dev/dsk/$ROOTSLICE from /UCE"  >> $OUTFILE
 #$SBIN/umount -f  /UCE
 echo " Done " >> $OUTFILE

}


fs_get ()
#
# Returns the File System Mount Point and type associated with a device path
#
# $1: Device path
#
{
 sysdev=$1
  if [ "`$BIN/grep ${sysdev} /etc/vfstab`" != "" ];then
        dev=`$BIN/grep ${sysdev} /etc/vfstab| awk '{print $3" "$4}'`
        echo $dev
  else
        echo " " >> $OUTFILE
        echo " Unable to determine File System Mount Point of ${sysdev}"  >> $OUTFILE
        echo "0 0"
  fi
}




volmgt ()
#
# Check if filesystem is VxVM or SDS/SVM,
# returning R0 if is not a mirrordisk, M1 if is a SDS/SVM Mirror or M0 if not
# returning V1 if is a VxVM Mirror or V0 if not
#
#    $1: Volume path
#
{
    device=$1
    sysdev=`echo $device | $BIN/grep "/dev/dsk/" | $BIN/sed -e "s/\/dev\/dsk\///g"`
    if [ "${sysdev}" != "" ]; then
        echo R0
    else
        metadev=`echo $device | $BIN/grep "/dev/md/dsk/" | $BIN/sed -e "s/\/dev\/md\/dsk\///g"`
```

```
        if [ "${metadev}" != "" ]; then
            metastat_line=`$SBIN/metastat -p | $BIN/grep "^${metadev} "`
            mirror_dev=`echo ${metastat_line} | $BIN/grep "\-m"`
            if [ `echo ${mirror_dev} | $BIN/awk '{ print NF }'` -gt 4 ]; then
                echo M1
            else
                echo M0
            fi
        else
            vxvmdev=`echo $device | $BIN/grep "/dev/vx/dsk/" | $BIN/sed -e "s/\/dev\/vx\/dsk\///g"`
            if [ "${vxvmdev}" = "" ]; then
                echo RR
            fi
            vxvm_vol=`echo ${vxvmdev} | $BIN/awk -F\/ '{ print $2 }'`
            if [ "${vxvm_vol}" = "" ]; then
                vxvm_vol=${vxvmdev}
            fi
            if [ `$SBIN/vxprint | $BIN/grep "^pl" | $BIN/awk '{ print $3 }' | $BIN/grep -w
${vxvm_vol} | $BIN/awk 'END { print NR }'` -gt 1 ]; then
                echo V1
            else
                echo V0
            fi
        fi
    fi
    #return 0
}

################################################################################
#######################      SDS/SVM FUNCTIONS            ######################
################################################################################

modify_sds_files ()
#
# Modify /etc/system, /etc/vfstab and /etc/lvm/mddb.cf in Mirrordisk to boot without SDS/SVM
#
#  $1= Metadevices with it's respective device
#
{
        MD=$METADEV

  # Modify /etc/system
  ##############################################################
        echo " "
        echo " "
        echo " " >> $OUTFILE
        echo " Modifying /UCE/etc/system on mirrored disk"  >> $OUTFILE
        echo " Moving /UCE/etc/system on /UCE/etc/system.old ..."  >> $OUTFILE
        $BIN/cp /UCE/etc/system /UCE/etc/system.old

        echo " Erasing SDS Entries from /UCE/etc/system on mirrored disk..."  >> $OUTFILE
        perl -i -pe "s/^forceload: misc\/md.*//g" /UCE/etc/system
              perl -i -pe "s/^rootdev:.*//g" /UCE/etc/system
              perl -i -pe "s/^set md:mddb.*//g" /UCE/etc/system
              perl -i -pe "s/^set md:mirrored.*//g" /UCE/etc/system
        echo " Done." >> $OUTFILE


  # Modify /etc/vfstab
  ##############################################################
        echo " "
        echo " " >> $OUTFILE
        echo " Modifying /UCE/etc/vfstab on mirrored disk"  >> $OUTFILE
        echo " Moving /UCE/etc/vfstab to /UCE/etc/vfstab.old "  >> $OUTFILE

  #Modifying local devices
        cp /UCE/etc/vfstab /UCE/etc/vfstab.old


        echo ${MD}|while read v_vol v_disk
        do
        if [ "${v_vol}"  != "" ]; then
         if [ "`$SBIN/swap -l| $BIN/grep ${v_vol}`" != "" ];then
                perl -i -pe "s/\/dev\/md\/dsk\/${v_vol}/\/dev\/dsk\/${v_disk}/g" /UCE/etc/vfstab
         else
                perl -i -pe "s/\/dev\/md\/dsk\/${v_vol}/\/dev\/dsk\/${v_disk}/g" /UCE/etc/vfstab
                perl -i -pe "s/\/dev\/md\/rdsk\/${v_vol}/\/dev\/rdsk\/${v_disk}/g" /UCE/etc/vfstab
         fi
               fi
        done

  #Erasing the rest of SDS/SVM lines in vfstab

        perl -i -pe "s/^\/dev\/md\/dsk/#\/dev\/md\/dsk/g" /UCE/etc/vfstab
```

```
        echo " Done."  >> $OUTFILE


  # Modify /etc/lvm/mddb.cf
  ############################################################
        echo " " >> $OUTFILE
        echo " Modifying /UCE/etc/lvm/mddb.cf on mirrored disk"  >> $OUTFILE
        echo " Moving /UCE/etc/lvm/mddb.cf to /UCE/etc/lvm/mddb.cf.old on mirrored disk"  >> $OUTFILE
        mv /UCE/etc/lvm/mddb.cf /UCE/etc/lvm/mddb.cf.old

        $BIN/cat /UCE/etc/lvm/mddb.cf.old | \
        $BIN/grep "#" > /UCE/etc/lvm/mddb.cf

        echo " Done."  >> $OUTFILE

}

split_sds_mirror ()
#
# Dettach and erase mirrordisk
#
#  $1=Mirror metadevices that has to be dettached
#
#
{
  mir1=$SUBMIRROR1
#
# Constructing the script that will mirror the disk after patching
#

if [ ! -f $OUTDIRBASE/mirror_disk ]; then
        echo " Providing Mirroring Script FILE:      $OUTDIRBASE/mirror_disk"  >> $OUTFILE
        touch $OUTDIRBASE/mirror_disk
else
        echo " Providing New Script FILE:      $OUTDIRBASE/mirror_disk"  >> $OUTFILE
        rm $OUTDIRBASE/mirror_disk
        touch $OUTDIRBASE/mirror_disk
fi

 echo "#!/bin/ksh" >> $OUTDIRBASE/mirror_disk
 echo "#This Script generate the mirror as well as done before patching..." >>
$OUTDIRBASE/mirror_disk
 echo " " >> $OUTDIRBASE/mirror_disk

 echo " $TIME: Dettaching mirrors ..."
>> $OUTFILE

 for  md_mir1 in $mir1
 do
       md_mir=`$SBIN/metastat -p| $BIN/grep $md_mir1| $BIN/grep m| awk '{print $1}'`
       dev_slc_no=`$SBIN/metastat -p| $BIN/grep $md_mir1| $BIN/grep -v m| awk '{print $4}' | head
-1`
       dev_mir="/dev/md/dsk/$md_mir"
              sys_fs=`fs_get ${dev_mir}`
       meta_fs=`echo $sys_fs| $BIN/awk '{print $1}'`
       fs_type=`echo $sys_fs| $BIN/awk '{print $2}'`
       if [ "$meta_fs" != "0" ];then
              if [ "$fs_type" != "swap" ];then
                     echo " Detaching $meta_fs: metadetach $md_mir $md_mir1" >> $OUTFILE
                     #$SBIN/metadetach $md_mir $md_mir1
                     echo " Clearing $md_mir1: metaclear -f $md_mir1" >> $OUTFILE
                     #$SBIN/metaclear -f  $md_mir1

                      echo "# $meta_fs device " >> $OUTDIRBASE/mirror_disk
              else
                     echo " Detaching $fs_type: metadetach $md_mir $md_mir1" >> $OUTFILE
                     #$SBIN/metadetach $md_mir $md_mir1
                     echo " Clearing $md_mir1: metaclear -f $md_mir1" >> $OUTFILE
                     #$SBIN/metaclear -f  $md_mir1

                      echo "# $fs_type device " >> $OUTDIRBASE/mirror_disk

              fi
              echo " $SBIN/metainit $md_mir1 1 1 $dev_slc_no" >> $OUTDIRBASE/mirror_disk
              echo " $SBIN/metattach $md_mir $md_mir1" >> $OUTDIRBASE/mirror_disk

       else
              #echo " Exiting..." >> $OUTFILE
              echo " MEtadevice has not been detached" >> $OUTFILE
              #exit 1
       fi
 done
 echo " $TIME: Dettaching mirrors ... Done"
```

```
>> $OUTFILE
}

################################################################################
#######################      VxVM FUNCTIONS        ###################
################################################################################

build_slice ()
#
# Return VxVM vol with pre-encpasulated slice
# Pre-encapsulated slice information are stored in vfstab
# with a "#NOTE:" prefix.
#
#     $1 : disk
#
#
#
{
  disk=`echo $1| $BIN/cut -ds -f1`
  (cat /etc/vfstab \
      | egrep "^#NOTE: volume" | egrep "encapsulated partition") \
      | while read line
        do
       vol=`echo $line| awk '{print $3}'`
       slice_no=`echo $line| awk '{ print substr($NF, length($NF)-1, 2)}'`
              echo $vol" "$disk$slice_no
       done

# If that vfstab entry is missing it will helps on next version
#  slices="0 1 5 6 7"
 # for i in $slices
 # do
#     disk_tag=`prtvtoc -h /dev/rdsk/$disk | grep "^       $i" | $BIN/awk '{print $2}'`
#     if [ "$disk_tag" != "" ]; then
#           case $disk_tag in
#           '0')
#             echo $disk$disk_tag" mount point unassigned"
#             ;;
#           '2')
#             echo $disk$disk_tag" mount point /"
#             ;;
#           '3')
#             echo $disk$disk_tag" swap"
#             ;;
#           '4')
#             echo $disk$disk_tag" mount point /usr"
#             ;;
#           '7')
#             echo $disk$disk_tag" mount point /var"
#             ;;
#           '8')
#             echo $disk$disk_tag" mount point home"
#             ;;
#            *)
#             echo $disk$disk_tag" No mount point assigned to this slice"
#             ;;
#           esac
#   fi
#  done
}


split_vxvm_mirror ()
#
# Dettach and erase mirrordisk
#
#  $1=Mirrored root slice
#
{
 MIRROR=`echo $1| $BIN/cut -ds -f1`

 echo " $TIME: Dettaching mirrors ..."
>> $OUTFILE

plex=""
sd=""
(/usr/sbin/vxprint -nvg rootdg) | while read vol
do
#       dg_vol_array=`$SBIN/vxprint -thfg rootdg -v $vol 2> /dev/null`;
       plex_mirror=`$SBIN/vxprint -lvg rootdg -v $vol | $BIN/grep plex|awk -F, '{print $2}'`;
       sd_mirror_disk=`$SBIN/vxprint -thfg rootdg -v $vol| $BIN/grep $plex_mirror| $BIN/grep "^sd"|
awk '{print $8}'`
             if [ "$sd_mirror_disk" != "$MIRROR" ]; then
              plex_mirror=`$SBIN/vxprint -lvg rootdg -v $vol | $BIN/grep plex|awk -F, '{print $1}'|
```

```
awk -F= '{print $2}'`;
                sd_mirror_disk=`$SBIN/vxprint -thfg rootdg -v $vol| $BIN/grep $plex_mirror| $BIN/grep
"^sd"|awk '{print $8}'`
                if [ "$sd_mirror_disk" != "$MIRROR" ]; then
                        echo " Unable to determine mirrored disk... Exiting"  >> $OUTFILE
                        exit 1
                fi
               fi
        condition="false"
        ($SBIN/vxprint -thfg rootdg -v $vol) | while read line
        do
                if [ "`echo $line |awk '{print $2}`" != "$plex_mirror" ]; then
                        if [ "$condition" = "false" ]; then
                                condition="false"   #Still not arrive to the sd line
                        else
                                condition="true"   #Have arrive to the sd line
                                if [ "`echo $line|awk '{print $1}'`" = "sd" ]; then
                                        sd_mirror=`echo $line|awk '{print $2}'`
                                        mirror_disk=`echo $line|awk '{print $4}'`
                                        mirror_device=`echo $line|awk '{print $8}'`
                                        plex="$plex $plex_mirror"
                                        sd="$sd $sd_mirror"

                                fi

                        fi
                else
                        if [ "$condition" = "false" ]; then
                                condition="true"   #Still not arrive to the sd line
                        else
                                condition="true"   #Have arrive to the sd line
                        fi
                fi
        done
done #  @line_vols

echo " /usr/sbin/vxplex -g rootdg dis $plex"  >> $OUTFILE
$SBIN/vxplex -g rootdg dis $plex
echo " /usr/sbin/vxsd -g rootdg dis $sd"  >> $OUTFILE
$SBIN/vxsd -g rootdg dis $sd
echo " /usr/sbin/vxedit rm $plex"  >> $OUTFILE
$SBIN/vxedit rm $plex
echo " /usr/sbin/vxedit rm $sd"  >> $OUTFILE
$SBIN/vxedit rm $sd
echo " /usr/sbin/vxdg -g rootdg rmdisk $sd"  >> $OUTFILE
$SBIN/vxdg -g rootdg rmdisk $mirror_disk
#echo " /usr/sbin/vxdiskunsetup -C $mirror_disk"  >> $OUTFILE

echo " $TIME: Dettaching mirrors ... Done"
>> $OUTFILE
}

modify_vxvm_files ()
#
# Modify /etc/system, /etc/vfstab and /etc/lvm/mddb.cf in Mirrordisk to boot without VxVM
#
#  $1: Disk with Slice 0
#
{
  disk=$1

  # Modify /etc/system
  ################################################################
        echo " "
        echo " "
        echo " " >> $OUTFILE
        echo " Modifying /UCE/etc/system on mirrored disk "  >> $OUTFILE
        echo " Moving /UCE/etc/system on /UCE/etc/system.old ..."  >> $OUTFILE
        $BIN/cp /UCE/etc/system /UCE/etc/system.old

        echo " Erasing VxVM Entries from /UCE/etc/system ..."  >> $OUTFILE
        perl -i -pe "s/^forceload: drv\/vx.*//g" /UCE/etc/system
        perl -i -pe "s/^forceload: drv\/sbus.*//g" /UCE/etc/system
        perl -i -pe "s/^forceload: drv\/ssd.*//g" /UCE/etc/system
               perl -i -pe "s/^rootdev:.*//g" /UCE/etc/system
               perl -i -pe "s/^set vxio.*//g" /UCE/etc/system
        echo " Done." >> $OUTFILE


  # Modify /etc/vfstab
  ################################################################
        echo " "
        echo " " >> $OUTFILE
        echo " Modifying /UCE/etc/vfstab on mirrored disk "  >> $OUTFILE
```

```
       echo " Moving /UCE/etc/vfstab to /UCE/etc/vfstab.old "  >> $OUTFILE



  #Modifying local devices
       cp /UCE/etc/vfstab /UCE/etc/vfstab.old

       echo " Modifying /UCE/etc/vfstab entries to pre-encapuslate slices "  >> $OUTFILE
       (build_slice $disk)|while read line
       do
              v_vol=`echo $line| awk '{print $1}'`
              v_disk=`echo $line| awk '{print $2}'`
              if [ "${v_vol}"  != "" ]; then
               if [ "`$SBIN/swap -l| $BIN/grep ${v_vol}`" != "" ];then
                      perl -i -pe "s/\/dev\/vx\/dsk\/*${v_vol}/\/dev\/dsk\/${v_disk}/g"
/UCE/etc/vfstab
               else
                      perl -i -pe "s/\/dev\/vx\/dsk\/*${v_vol}/\/dev\/dsk\/${v_disk}/g"
/UCE/etc/vfstab
                      perl -i -pe "s/\/dev\/vx\/rdsk\/*${v_vol}/\/dev\/rdsk\/${v_disk}/g"
/UCE/etc/vfstab
               fi
                     fi
       done

  #Erasing the rest of VxVM lines in vfstab

       perl -i -pe "s/^\/dev\/vx\/dsk/#\/dev\/vx\/dsk/g" /UCE/etc/vfstab

       echo " Done."  >> $OUTFILE


  # Modify startup of VxVM
  ###############################################################
       echo " " >> $OUTFILE
       echo " Modifying Startup of VxVM on mirrored disk "  >> $OUTFILE
       echo " Making sure Volume Manager does not start on the next boot... "  >> $OUTFILE
       touch /UCE/etc/vx/reconfig.d/state.d/install-db
       echo " Removing the flag that tells Volume Manager that the root disk is encapsulated"  >>
$OUTFILE
       rm /UCE/etc/vx/reconfig.d/state.d/root-done

       echo " Done."  >> $OUTFILE

}

fsck_vxvm ()
{
#
# Check every Slice on mirror disk
#
#  $1: Disk with Slice 0
#
              disk=$1
       echo " Checking mirror disk..."     >> $OUTFILE
       echo ""     >> $OUTFILE

       (build_slice $disk)|while read line
       do
              v_vol=`echo $line| awk '{print $1}'`
              v_disk=`echo $line| awk '{print $2}'`
              echo " $TIME: Runing fsck on $v_vol : /dev/rdsk/$v_disk"     >> $OUTFILE
              $SBIN/fsck -y -F ufs /dev/rdsk/$dsk_part 2 > /dev/null
               echo " Done " >> $OUTFILE

       done

}

################################################################################
#
#                                                  main program
#
################################################################################


case "$1" in

help)
       echo " "
       echo " volume_split_mirror"
       echo " \tVersion 1.0 "
       echo " "
       echo " Please give feedback of this script to felipe.herrera@sun.com"
```

```
        echo " "
        exit 0
        ;;
esac




# Verify runlevel
#############################################################

if [ "${RUNLEVEL}" != "1" ]
then
        clear
        echo " "
        echo " "
        echo " Running not in Single User Mode!!! "
        echo " "
        echo " "
        echo " Please guarantee that all Volumes are correctly mirrored before to proceed"
        echo " "
        echo " "
        echo " Please consider also the settings of boot-devices in OpenBootPrompt."
        echo " "
        echo " "
fi


echo "  "
echo " Determining logs files and directories...."
echo "  "


if [ ! -d $OUTDIRBASE ]; then
        echo "Providing $OUTDIRBASE"
        mkdir -p $OUTDIRBASE
else
        echo "$OUTDIRBASE already exists."
fi
if [ ! -d $OUTDIR ]; then
        echo " "
        echo " Providing Output-Directory: $OUTDIR"
        mkdir -p $OUTDIR
else
        echo " "
        echo " Output-Directory:          $OUTDIR already exists."
fi
if [ ! -f $OUTFILE ]; then
        echo " Providing Output-FILE:      $OUTFILE"
        touch $OUTFILE
fi

echo " " >> $OUTFILE
echo " " >> $OUTFILE
echo " " >> $OUTFILE
echo " " >> $OUTFILE
echo "#############################################################" >> $OUTFILE
echo " $TIME: Starting Volum Management detection prior spliting mirror..."           >> $OUTFILE



# Determining use of VxVM or SDS/SVM
#############################################################

BOOTPATH=`$SBIN/prtconf -pvv | $BIN/grep bootpath| $BIN/awk -F: '{print $2}'| $BIN/awk -F\' '{print
$2}'`
#
# In some environments bootpath in OBP differs of bootpath recognized by OS, please check this issue
# on EIS checklist

#
if [ "`echo ${BOOTPATH} | grep -w disk`" != "" ];then
        echo " WARNING:  bootpath in OBP differs of bootpath recognized by OS, check EIS checklist
to solve that."  >> $OUTFILE
        BOOTPATH=`echo $BOOTPATH| $BIN/sed s,disk,sd,g`
fi
BOOTDISK=`$BIN/ls -l /dev/rdsk | $BIN/grep ${BOOTPATH} | $BIN/awk '{print $9}' | $BIN/cut -ds -f1|
$BIN/head -1`
if [ "$BOOTDISK" = "" ]; then
    BOOTPATH=`echo $BOOTPATH| $BIN/sed s,sd,dad,g`
    BOOTDISK=`$BIN/ls -l /dev/rdsk | $BIN/grep ${BOOTPATH} | $BIN/awk '{print $9}' | $BIN/cut -ds
-f1| $BIN/head -1`
fi
```

```
if [ "$BOOTDISK" = "" ]; then
    BOOTPATH=`echo $BOOTPATH| $BIN/sed s,dad,ssd,g`
    BOOTDISK=`$BIN/ls -l /dev/rdsk | $BIN/grep ${BOOTPATH} | $BIN/awk '{print $9}' | $BIN/cut -ds
-f1| $BIN/head -1`
fi
if [ "$BOOTDISK" = "" ]; then
    #Unresolved bootpath
    exit 1
fi

ROOTPATH=`$BIN/df /|$BIN/cut -d'(' -f2- |$BIN/cut -d')' -f1`

VolMGT=`volmgt ${ROOTPATH}`

case $VolMGT in
'R0')
       echo "#############################################################" >> $OUTFILE
       echo " $TIME: WARNING:File System not mirrored!!!"
>> $OUTFILE
       exit 1
       ;;
'M0')
       echo "#############################################################" >> $OUTFILE
       echo " $TIME: WARNING:Volume Management SDS/SVM but not mirrored!!!"        >> $OUTFILE
       exit 1
       ;;
'M1')
       echo "#############################################################" >> $OUTFILE
       echo " $TIME: Volume Management SDS/SVM"
>> $OUTFILE


       SUBMIRROR0=""
              SUBMIRROR1=""
              MIRROR=""
              MIRRORDISK=""
       METADEV=""

             SUBMIRROR=`$SBIN/metastat -p | $BIN/grep ${BOOTDISK}| $BIN/awk '{print $1}'`
             for i in ${SUBMIRROR}
             do
                SUBMIRROR0="${SUBMIRROR0} $i"
                j=`$SBIN/metastat -p | $BIN/grep $i| $BIN/grep m | $BIN/awk '{print $4}'`
                k=`$SBIN/metastat -p | $BIN/grep $i| $BIN/grep m | $BIN/awk '{print $1}'`
                MIRROR="${MIRROR} $k"
                SUBMIRROR1="${SUBMIRROR1} $j"
                MIRRORDISK="${MIRRORDISK} `$SBIN/metastat -p | $BIN/grep $j | $BIN/grep -v m|
$BIN/awk '{print $4}'| head -1`"
                echo "Mirrordisk: " ${MIRRORDISK}
          METADEV="$k `$SBIN/metastat -p | $BIN/grep $j| $BIN/grep -v m| $BIN/awk '{print $4}'| head
-1`\n$METADEV"
            done

       echo " $TIME: Acknowledging SDS/SVM Boot environment Configuration ..."
>> $OUTFILE

       split_sds_mirror $SUBMIRROR1

       #Mirror partition Check

       echo " Checking mirror disk..."     >> $OUTFILE
       echo ""     >> $OUTFILE

       for dsk_part in $MIRRORDISK
       do
              echo " $TIME: Runing fsck on /dev/rdsk/$dsk_part"     >> $OUTFILE
              #$SBIN/fsck -y -F ufs /dev/rdsk/$dsk_part
               echo " Done " >> $OUTFILE
              rootslice=`echo $dsk_part| $BIN/cut -ds -f1`s0
       done
       install_bootblk  $rootslice
       mount_etc_disk $rootslice
       #modify_sds_files $METADEV
       umount_etc_disk $rootslice

       exit 0
       ;;
'V0')
       echo "#############################################################" >> $OUTFILE
       echo " $TIME: WARNING:Volume Management VxVM but not mirrored!!!"     >> $OUTFILE
       exit 1
       ;;
'V1')
       echo "#############################################################" >> $OUTFILE
```

```
        echo " $TIME: Volume Management VxVM"                                   >> $OUTFILE
        MIRRORDISK=`($SBIN/vxprint -tdg rootdg | grep "^dm") | $BIN/grep -i -v ${BOOTDISK}|
$BIN/head -1| $BIN/awk '{print $3}'`
        rootslice=`echo $MIRRORDISK| $BIN/cut -ds -f1`s0

        split_vxvm_mirror $rootslice
        fsck_vxvm $rootslice
        install_bootblk  $rootslice
        mount_etc_disk $rootslice
        modify_vxvm_files $rootslice
        umount_etc_disk $rootslice

        #
        # Constructing the script that will mirror the disk after patching
        #

        if [ ! -f $OUTDIRBASE/mirror_disk ]; then
                        echo " Providing Mirroring Script FILE:      $OUTDIRBASE/mirror_disk"  >>
$OUTFILE
                        touch $OUTDIRBASE/mirror_disk
        else
                        echo " Providing New Script FILE:      $OUTDIRBASE/mirror_disk"  >> $OUTFILE
                        rm $OUTDIRBASE/mirror_disk
                        touch $OUTDIRBASE/mirror_disk
        fi

        echo "#!/bin/ksh" >> $OUTDIRBASE/mirror_disk
        echo "#This Script generate the mirror as well as done before patching..." >>
$OUTDIRBASE/mirror_disk
        echo " " >> $OUTDIRBASE/mirror_disk
               destination_disk=`echo $rootslice | $BIN/cut -ds -f1`s2
        rootdisk=`($SBIN/vxprint -tdg rootdg | grep "^dm") | $BIN/grep -i  ${BOOTDISK}| $BIN/head
-1| $BIN/awk '{print $2}'`
        echo " /usr/sbin/vxdg -g rootdg adddisk mirrordisk=$destination_disk" >>
$OUTDIRBASE/mirror_disk
        echo " /etc/vx/bin/vxmirror $rootdisk mirrordisk" >> $OUTDIRBASE/mirror_disk
        #
        # Done
        #

        exit 0
        ;;
*)
        echo "############################################################" >> $OUTFILE
        echo " $TIME: Unable to determine Volume Management!!!"              >> $OUTFILE
        exit 1
        ;;
esac

TIME=`$BIN/date '+%m%d%y_%H%M'`
echo " " >> $OUTFILE
echo " " >> $OUTFILE
echo "\tFinished "  >> $OUTFILE
echo "  " >> $OUTFILE
echo " ------------------------------------------------------ " >> $OUTFILE
echo " $TIME: Deencapsulation has finshed correcly. " >> $OUTFILE
echo " ------------------------------------------------------ " >> $OUTFILE
echo " " >> $OUTFILE

exit 0
```

9.2. Generar disco de *mirror* después de un parcheo (attach_mirror.ksh)

Una vez se ha finalizado correctamente la operativa de parcheo y ha pasado un tiempo prudencial, podemos ejecutar como una acción programada el siguiente script:

```
#!/usr/bin/ksh -hp
#
# Attach mirror after Patch a system with /var/tmp/UCE/mirror_disk
#
# Author: Felipe.Herrera@sun.com
#
# v1.0: 26-Mar-2008
#

#
# This is an easy example using a shared NFS-directory
#

MIRROR_DISK=/var/tmp/UCE/mirror_disk

chmod +x $MIRROR_DISK
ksh $MIRROR_DISK
```

9.3. Generar Explorer (run_explorer.ksh)

Como medida de contingencia, antes de cualquier parcheo, un Ingeniero Sun extrae toda la información de configuración de un Sistema Solaris mediante una herramienta ya predefinida del Sistema, que recopila toda aquella información de la configuración necesaria para restaurar el Sistema en caso de Desastre o necesidad de reinstalar el equipo. Esta herramienta viene integrada como paquete de los Sistemas Operativos Sun Solaris y podemos extraer por defecto toda la configuración o decidir que configuración extraer. En este caso se ha decidido no extraer información no crítica para la configuración, para de este modo minimizar el tiempo de ejecución del script.

```
#!/usr/bin/ksh -hp
#
# run_explorer
#
#
# Author: Juergen.Fleischer@sun.com
# Modified by: Felipe.Herrera@sun.com
#
# v2.0: 17-Mar-2008
#


LC_ALL=C
LANG=C
export LC_ALL LANG

EXPLORER=/opt/SUNWexplo/bin/explorer
DEFAULTS=!messages,!ndd,!nbu,!nbu_extended

cd /tmp
${EXPLORER} -w ${DEFAULTS}

exit 0
```

9.4. Copiar Explorer a un repositorio local (copy_explorer2repository.ksh)

Como medida de seguridad añadida, después de ejecutar el explorer ,se lanza otro script que copiará el fichero comprimido en un repositorio de datos, que se encuentre en la red del cliente.

```
#!/usr/bin/ksh -hp
#
# copy_explorer2repository
#
# Author: Juergen.Fleischer@sun.com
#
# v1.0: 31-Oct-2006
#

#
# This is an easy example using a shared NFS-directory
#

REPOSITORY=/net/dusserv1/export/explorer

EXPLORER=$(ls /opt/SUNWexplo/output/explorer.*.gz 2>/dev/null|tail -1)

if [ -f ${EXPLORER} ]
then
  cp $EXPLORER $REPOSITORY
  echo "explorer output transferred to $REPOSITORY ..."
  echo
  exit 0
else
  echo No explorer output found ...
  echo
  exit 1
fi
```

9.5. Realizar una copia de las zonas locales a un repositorio (tar_zones.ksh)

Debido a la naturaleza de la zonas (ver Apéndice 5), al parchear la zona global según la metodología detallada anteriormente, es necesario realizar una copia de seguridad de cada una de las zonas locales como medida de contingencia, para en caso de ser necesario tener una marcha atrás más rápida que la implícita de Sun Conneciton Satellite.

```
#!/usr/bin/ksh -hp
#
# Create tar of the Local zones
#
# Author: Felipe.Herrera@sun.com
#
# v1.0: 26-Mar-2008
#

#
# This is an easy example using a shared NFS-directory
#

REPOSITORY=/net/sung01/soft/images/sunx01/zones

zones=`zoneadm list| grep -v global`

for i in $zones
 do
    path=`zoneadm -z $i list -p|awk -F: '{print $4}'`
    echo "***"
    echo "Creating tar file of $i in $REPOSITORY"
    echo "***"
    tar cf - ${path}| gzip -c > $REPOSITORY/$i.tar.gz

done
```

10. Apéndice 5 – Definición de Zonas en Solaris 10

La función Zonas de Solaris™ del sistema operativo Solaris proporciona un entorno aislado en el que ejecutar aplicaciones en el sistema. Zonas de Solaris es un componente del entorno de contenedores de Solaris.

10.1. Descripción general de la zonas

La tecnología de partición de zonas de Solaris se utiliza para virtualizar servicios del sistema operativo y proporcionar un entorno aislado y seguro para ejecutar aplicaciones. Una **zona** es un entorno de sistema operativo virtualizado creado en una única instancia del sistema operativo Solaris. Cuando se crea una zona, se genera un entorno de ejecución de aplicaciones en el que los procesos están aislados del resto del sistema. Este aislamiento evita que los procesos que se están ejecutando en una zona sean controlados o se vean afectados por los procesos que se están ejecutando en otras zonas. Incluso un proceso que se está ejecutando con credenciales de superusuario no puede ver ni afectar a la actividad que se esté realizando en otras zonas.

Una zona también proporciona un nivel abstracto que separa las aplicaciones de los atributos físicos del equipo en el que se han implementado. Entre los ejemplos de este tipo de atributos, se incluyen las rutas de dispositivos físicos.

Las zonas pueden utilizarse en cualquier equipo en el que se ejecute Solaris 10. El máximo de zonas que puede haber en un sistema es de 8192. El número de zonas que puede admitir de forma eficaz un único sistema lo determinan los requisitos de recursos totales de la aplicación que se ejecuta en todas las zonas.
Existen dos tipos de modelos de sistema de archivos root de zonas no globales: disperso y completo. El modelo de **zona root dispersa** optimiza el uso compartido de objetos. El modelo de **zona root completa** permite la máxima configuración.

10.2. Cuando se utilizan las zonas

Las zonas son idóneas para entornos que consolidan varias aplicaciones en un único servidor. Debido al coste y la complejidad de administrar varios equipos, se recomienda consolidar varias aplicaciones en servidores más grandes y escalables.

La siguiente figura muestra un sistema con tres zonas. Cada una de las zonas (`apps`, `users` y `work`) ejecuta una carga de trabajo no relacionada con las cargas de trabajo de las demás zonas, en un ejemplo consolidado. Este ejemplo ilustra que pueden ejecutarse diferentes versiones de la misma aplicación sin las consecuencias negativas de las diferentes zonas, para que cumplan los requisitos de la consolidación. Cada zona puede proporcionar un conjunto se servicios personalizados.

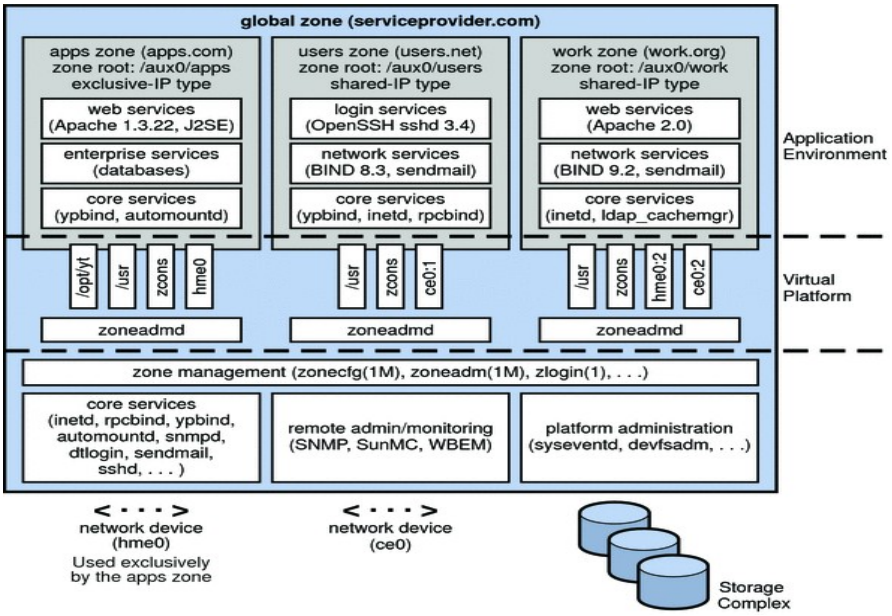

figura 27 – Ejemplo de consolidación de servidor de zonas

Las zonas permiten un uso más eficaz de los recursos en el sistema. La reasignación dinámica de recursos permite mover los recursos no utilizados a otros contenedores, según precise. El aislamiento de seguridad y fallos significa que las aplicaciones con un comportamiento anómalo no requieren un sistema dedicado e infrautilizado. Con el uso de las zonas, estas aplicaciones pueden consolidarse con otras aplicaciones.

Las zonas permiten delegar algunas funciones administrativas a la vez que se mantiene la seguridad global del sistema.

10.3. Funcionamiento de las zonas

Una zona no global sería similar a una caja. Una o varias aplicaciones pueden ejecutarse en esa caja sin interactuar con el resto del sistema. Las zonas de Solaris aíslan las aplicaciones y servicios utilizando unos límites flexibles y bien definidos. Las aplicaciones que se ejecutan en la misma instancia del sistema operativo Solaris se pueden administrar de forma independiente. De este modo, pueden ejecutarse diferentes versiones de la misma aplicación en zonas distintas, para cumplir los objetivos de la configuración.

Un proceso asignado a una zona puede manipular, supervisar y comunicarse directamente con otros procesos asignados a la misma zona. El proceso no puede llevar a cabo estas funciones con procesos que están asignados a otras zonas del sistema o con procesos que no están asignados a ninguna zona. Los procesos asignados a diferentes zonas sólo pueden comunicarse a través de las API de red.

A partir de Solaris 10 8/07, las redes IP pueden configurarse de modos distintos, en función de si la zona tiene su instancia IP exclusiva o comparte la configuración y el estado de la capa de IP con la zona global.

Cada sistema Solaris contiene una **zona global** . La zona global tiene una doble función. La zona global es tanto la zona predeterminada para el sistema, como la zona que se usa para el control administrativo de todo el sistema. Todos los procesos se ejecutan en la zona global si el **administrador global** no crea zonas **no globales** , denominadas simplemente zonas.

La zona global es la única zona desde la que se puede configurar, instalar, gestionar o desinstalar una zona no global. Sólo es posible arrancar la zona global desde el hardware del sistema. La administración de la infraestructura del sistema, como dispositivos físicos, enrutamiento o reconfiguración dinámica (DR), sólo es posible en la zona global. Algunos procesos con privilegios adecuados que se ejecuten en la zona global pueden acceder a objetos asociados con otras zonas.

Los procesos sin privilegios en la zona global podrían llevar a cabo operaciones no permitidas para los procesos con privilegios en una zona no global. Por ejemplo, los usuarios de la zona global pueden ver información sobre cada uno de los procesos del sistema. Si esta función presenta un problema para su sitio, puede restringir el acceso a la zona global.

Se asigna un nombre a cada zona, incluida la zona global. La zona global siempre tiene el nombre `global`. Cuando se inicia la zona, el sistema también asigna a cada zona un identificador numérico exclusivo. La zona global siempre se asigna al ID `0`.

Cada zona también tiene un nombre de nodo completamente independiente del nombre de la zona. El nombre de nodo lo asigna el administrador de la zona.

Cada zona tiene una ruta a su directorio raíz relativo al directorio raíz de la zona global.

La clase de planificación para una zona no global se configura como la clase de planificación para el sistema de forma predeterminada.

10.4. Características de las zonas no globales

Una zona proporciona aislamiento a prácticamente cualquier nivel de granularidad que se desee. Una zona no necesita una CPU dedicada, ni un dispositivo físico ni una porción de memoria física. Estos recursos pueden estar multiplexados a lo largo de varias zonas que se ejecuten en un único sistema o dominio. También pueden estar asignados en función de las zonas usando las funciones de gestión de recursos que estén disponibles en el sistema operativo.

Cada zona puede proporcionar un conjunto se servicios personalizados. Para aplicar el aislamiento básico de los procesos, cada uno de ellos puede ver o señalar únicamente aquellos procesos que se encuentren en la misma zona. La comunicación básica entre las zonas se lleva a cabo asignando conectividad de red a cada IP de zona. Una aplicación que se ejecute en una zona no puede observar el tráfico de red de otra zona. Este aislamiento se mantiene aunque los respectivos flujos de paquetes viajen a través de la misma interfaz física.

Cada zona cuenta con una porción de la jerarquía del sistema de archivos. Como cada zona está limitada a su árbol de la jerarquía del sistema de archivos, una carga de trabajo que se esté ejecutando en una zona concreta no puede acceder a los datos que estén en un disco de otra carga de trabajo que se ejecute en una zona diferente.

Los archivos utilizados por los servicios de nombres residen en la vista de un sistema de archivos raíz propio de una zona. De esta forma, los servicios de nombre que estén en distintas zonas estarán aislados unos de otros y podrán configurarse de forma diferente.

11. Apéndice 6 – Requisitos técnicos para la instalación de SCS

11.1. Arquitectura de Red

Diagrama de Protocolos / Puertos / Servidores

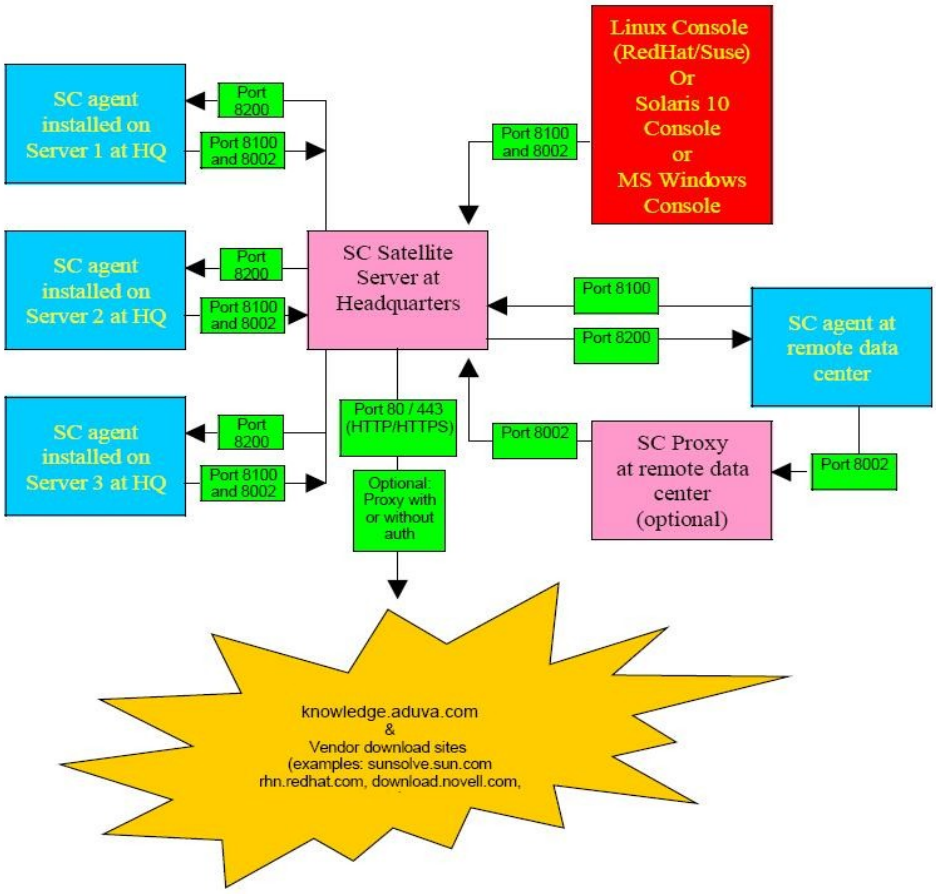

figura 28 : Diagrama de Red de Sun Connection Satellite

Protocolos de Comunicación

Los puertos y conexiones utilizadas por Sun Connection Satellite son los siguientes:

- Puerto 8100 y 8200 - Protocolo XDR TCP/IP con encriptación RSA publica/privada de 512 bits.
- Puerto 8002 - HTTPS
- A knowdlege.aduva.com – HTTPS
- A los distribuidores - HTTP / HTTPS / FTP (dependiendo de cada distribuidor)

URLS Externas

El Servidor Satélite requiere también acceder a los diferentes distribuidores vía Internet, dependiendo del sistema manejado, para bajar la actualizaciones y parches; aunque en la *KB* existe la información de estos paquetes y parches, los mismos sólo se podrán bajar de las URLS de los propios Distribuidores con el usuario que se utilice para este fin.

Estas conexiones son sólo de descarga e iniciadas por el Servidor Satélite. Ninguna información del cliente será enviada al exterior.

Site	Protocolo	Descripción
knowledge.aduva.com	HTTPS	SC Knowledgebase server
identity.sun.com	HTTPS	Sun Patch Authentication
sunsolve.sun.com	HTTP	Sun Patch downloads
www.redhat.com	HTTPS	Red Hat Authentication

Site	Protocolo	Descripción
rhn.redhat.com	HTTP/HTTPS	Red Hat downloads
download.rhn.redhat.com	HTTPS	Red Hat downloads
www.novell.com	HTTP/HTTPS	SuSE Authentication
download.novell.com	HTTP	SuSE downloads
you.novell.com	HTTPS	SuSE downloads
ftp.sunfreeware.com	FTP	Sun Freeware Downloads

Tabla 1: Relación de sitios de descarga por distribuidor y funcionalidad

Todos los accesos a las diferentes URLS antes descritas pueden ser realizados mediante conexión directa a Internet o mediante la configuración proxy que tenga el cliente en sus instalaciones .

11.2. Requisitos de Sun Connection Satellite

11.2.1. Consola

La Consola se puede instalar indistintamente y de forma simultanea en cualquiera de las plataformas que el cliente disponga en sus instalaciones y que coincidan con la siguiente arquitectura:

Sistemas Linux en Intel/AMD:
- SuSE
- Red Hat

Sistemas Microsoft Windows (2000/XP)

Sistemas Solaris 10
- SPARC
- x86/x64
- Paquetes requeridos para Solaris 10:
  SUNWfreetype2
  SUNWfontconfig-root
  SUNWfontconfig
  SUNWicu
  SUNWstsf
  SUNWxwxst
  SUNWxwxft

La consola también se puede instalar en el *System Dependency Server,* siempre y cuando se pueda exportar el DISPLAY a cualquiera de las estaciones de trabajo desde donde se desee operar.

11.2.2. *System Dependency Server* (SDS)

Acceso de red externo

Se debe verificar que exista el acceso HTTP / HTTPS y FTP desde el *System Dependency Server* a las URLS de los diferentes distribuidores mencionadas en apartados anteriores de este documento, para validar que el acceso es el correcto podremos ejecutar cualquiera de los siguientes comandos:

```
# curl -k --proxy <proxyhost[:port]> --proxy-user <user:pass>
https://knowledge.aduva.com/channels.xml
```

O

```
# wget <--proxy-user=user> <--proxy-passwd=pass>
https://knowledge.aduva.com/channels.xml
```

Acceso de red interno

Si existe algún *Corta fuegos* entre el *System Dependency Server* y cualquiera de los sistemas gestionados o agentes, se deberán abrir puertos en ambas direcciones (2 puertos desde el agente al Sistema Satélite y 1 puerto desde el Sistema Satélite al agente), según especificaciones mencionadas en el apartado 3.2 de este informe.

- Al Sistema Satélite: 8002 y 8100
- Al Agente: 8200

Arquitectura

El Sistema Satélite puede ser instalado en cualquiera de las siguientes arquitecturas:

- Intel o AMD64 para Red Hat/Suse
- Intel, AMD64 o SPARC para Solaris 10

Distribuciones

Las distribuciones soportadas donde el *System Dependency Server* puede estar instalado son:

- Sun Solaris 10 SPARC
- Sun Solaris 10 X64/X86
- Red Hat RHEL 4 AS/ES
- Novell SuSE SLES 9

Las imágenes instaladas deberán ser imágenes estándar, i.e no deberán ser imágenes minimizadas. Para Solaris 10 el cluster SUNWCall deberá ser el instalado. Los últimos requisitos podrán ser consultados en http://www.sun.com/service/sunconnection/specs.jsp

Requisitos Hardware

- Instalación: 512 megabytes mínimos de espacio libre en /usr/local o /opt/SUNWuce
- CPU debe ser como mínimo UltraSparc III 1.2GHz + y para Intel/AMD a 2 Ghz+ CPU
- 2 GB de memoria RAM es el mínimo aconsejado.
- De 7 a 10 GB de espacio libre recomendado en disco por canal.
- CD-ROM o DVD

Binarios Solaris requeridos

Se debe verificar que los binarios siguientes se encuentran instalados en Solaris 10:

- /usr/bin/unzip, /usr/bin/zipinfo (SUNWswmt)
- /usr/bin/bash (SUNWbash)
- /usr/sfw/bin/gtar (SUNWgtar)
- /usr/bin/gzip (SUNWgzip)
- /usr/bin/uncompress (SUNWesu)
- /usr/xpg4/bin/egrep (SUNWxcu4)
- /opt/sfw/bin/rpm (SFWrpm)
- /usr/sfw/bin/python (SUNWpython)
- /usr/lib/amd64/libpopt.so.0 (SUNWlibpopt)
- /usr/bin/od (SUNWtoo)

Parches SPARC

- 119254-34
- 124630-03
- 120050-05

Parches X86

- 119255-34
- 124631-03
- 120051-05

Si la Consola se va a instalar también en el SDS, los siguientes paquetes también serán necesarios ser instalados:

- SUNWfreetype2
- SUNWfontconfig-root
- SUNWfontconfig
- SUNWicu
- SUNWstsf
- SUNWxwxst
- SUNWxwxft

Binarios Linux requeridos

Se debe verificar que los binarios siguientes se encuentran instalados en Linux:

- /usr/lib/libz.so (zlib-…rpm, if 64bit box also install the 32bit version)
- /usr/bin/uncompress (ncompress-…rpm)
- /bin/tar (tar-…rpm)
- /usr/bin/unzip (unzip-…rpm)
- /usr/bin/file (file-…rpm)
- /usr/bin/md5sum (coreutils-…rpm)
- /bin/egrep (grep-…rpm)
- ncurses

Resolución de nombres

En caso de que el nombre del SDS no se pueda resolver desde cualquier agente mediante DNS, sería recomendado actualizar el /etc/hosts de cada uno de los agentes con la configuración adecuada. En cualquier caso si la asignación IP no se realiza mediante DHCP se aconseja utilizar direccionamiento estático para la Instalación del configurador de los agentes.

Otros

Para poder disponer de una *Knowdlege Base* completa se requerirán los *CDRooms* de instalación de las diferentes distribuciones que el cliente dispone en sus instalaciones o bien un repositorio de las imágenes usadas durante la instalación.

Para poder descargar todas la actualizaciones requeridas de los diferentes distribuidores serán necesarias las respectivas cuentas y passwords de RHN, Novell o Sunsolve con los privilegios adecuados.

11.2.3. Agente

Acceso de red

El ancho de banda mínimo entre el Servidor y los Agentes no deberá ser menor de 56K.

Arquitectura

El Sistema Agente puede ser instalado en cualquiera de las siguientes arquitecturas:

- Sun UltraSPARC
- AMD Opteron
- IA 32: Intel Pentium IV or equivalent

Distribuciones

Las distribuciones soportadas donde el Agente puede estar instalado son:

- Sun Solaris 8, 9, 10 SPARC
- Sun Solaris 10 X64/X86
- Red Hat RHEL 4 AS/ES
- Novell SuSE SLES 9

Requisitos Hardware

- Instalación: 256 megabytes mínimos de espacio libre en /usr/local o /opt/SUNWuce
- 512 MB de memoria RAM es el mínimo aconsejado.
- De 512 MB a 1 GB de espacio libre recomendado en /var para descargar las actualizaciones, dependiendo del tamaño de las mismas
- Se requiere privilegios de root para la instalación del Agente y conectividad IP al Servidor Satélite.

Binarios Solaris requeridos

Se debe verificar que los binarios siguientes se encuentran instalados en los sistemas Solaris:

- /usr/bin/bash (SUNWbash)
- /usr/bin/unzip (SUNWswmt)
- /usr/bin/gzip (SUNWgzip)
- /dev/random & /dev/urandom devices

Opcionalmente se deberán tener instalados los siguientes parches, aunque es preferible instalarlos antes del agente, en caso de no hacerlo, al lanzar el primer trabajo sobre el agente automáticamente serán instalados.

Parches S8 Sparc
- 110165-05 (sed patch)
- 108987-18
- 112097-06 (cpio patch)

Parches S9 Sparc
- 112951-13

Parches S10 Sparc
- 119254-34 – instalar antes agent
- 124630-03
- 122660-07 (zones patch)

Parches S10 x86
- 119255-34 – instalar antes agent
- 124631-03
- 122661-07 (zones patch)

Binarios Linux requeridos

Se debe verificar que los binarios siguientes se encuentran instalados en Linux:

- /bin/tar (tar-…rpm)
- /usr/bin/unzip (unzip-…rpm)
- /usr/bin/file (file-…rpm)
- /usr/bin/md5sum (coreutils-…rpm)
- /bin/egrep (grep-…rpm)

11.3. Grupo de sistemas seleccionados

A continuación se listan los sistemas que han sido seleccionados junto al cliente para realizar la Fase I de adopción de Sun Connection Satellite y que se detalla en este plan de proyecto, así como la función que cumple cada uno de ellos en la arquitectura de Sun Connection Satellite detallada en apartados anteriores:

Equipo	Versión Operativo	Función
sunx01	Solaris 10	Satélite
hppz68	Red Hat Enterprise Linux AS release 4 (Nahant Update 3)	Agente
hppz69	Red Hat Enterprise Linux AS release 4 (Nahant Update 3)	Agente
hppz70	Red Hat Enterprise Linux AS release 4 (Nahant Update 3)	Agente
hppz71	Red Hat Enterprise Linux AS release 4 (Nahant Update 3)	Agente
sunz42	Solaris 8	Agente
vmwz07	Red Hat Enterprise Linux AS release 3 (Taroon Update 1)	Agente
vmwz11	Red Hat Enterprise Linux AS release 3 (Taroon Update 1)	Agente
vmwz08	Red Hat Enterprise Linux AS release 3 (Taroon Update 1)	Agente
vmwz09	Red Hat Enterprise Linux AS release 3 (Taroon Update 1)	Agente
vmwz10	Red Hat Enterprise Linux AS release 3 (Taroon Update 1)	Agente
sung15	Solaris 10 Zona Global	Agente
zj2eeat23	Solaris 10 Zona local	Agente
zj2eeat25	Solaris 10 Zona local	Agente
zj2eeat19	Solaris 10 Zona local	Agente
zj2eeat27	Solaris 10 Zona local	Agente
zj2eeat31	Solaris 10 Zona local	Agente
sung16	Solaris 10 Zona Global	Agente
zj2eeat24	Solaris 10 Zona local	Agente
zj2eeat20	Solaris 10 Zona local	Agente
zj2eeat28	Solaris 10 Zona local	Agente
zj2eeat32	Solaris 10 Zona local	Agente
zj2eeat26	Solaris 10 Zona local	Agente
sunz67	Solaris 9	Agente
sunz76	Solaris 10	Agente
sunz57	Solaris 9	Agente

Tabla 2 – Sistemas Fase piloto

12. Apéndice 7 – Valoración económica del proyecto

A continuación se adjunta la valoración económica de las tareas necesarias para la implantación del proyecto en el cliente, así como los costes de las licencias del producto que deben ser contratadas, para poder implementar el proyecto en 25 Sistemas del cliente:

Cantidad	Description	Price
1	Servidor Sun Connection Satellite para Solaris 10 y servicios de instalación. Los servicios de instalación incluyen: 4 horas de formación, instalación de hasta 25 agentes e instalación del Servidor. SUBSCRIPCIÓN anual.	7.600,00 €
25	Sun Connection Provisioning Knowledge Channel. por nodo. SUBSCRIPCIÓN anual.	2.841,00 €
25	Sun Connection Update Knowledge Channel. por nodo. SUBSCRIPCIÓN anual.	2.841,00 €
1	Servicios de instalación básicos de Sun Connection Satellite hasta 25 agentes.	0,00 €
1	Piloto 25 servidores. Seleccion de 25 sistemas que cumplan los pre-requisitosde Sistema Operativo para instalar el producto. Definición de políticas y criterios para el desarrollo de 3 tipología de parcheado Implantación de las polítidas definidas en de 5 dias fuera de horas para la realización del parcheado	30.283,00 €
	Total.....	**43.565,00 €**

Tabla 5– Valoración económica del proyecto

"Los requisitos del Negocio que requieren un gran crecimiento generan mayor complejidad en los Centros de Cómputo, son los administradores quienes necesitan gestionar el creciente volumen de datos, aplicaciones, y usuarios, así como la rápida proliferación de los servidores y los diferentes sistemas operativos. En este proyecto se pretende reducir la complejidad en la gestión de los Centros de Cómputo, combinando la automatización de la gestión del ciclo de vida y todas las medidas de contingencia necesarias para mantener la integridad de los mismos."

"Business requirements that demand tremendous growth are creating more datacenter complexity, the administrators are who need to manage increasing volumes of data, applications and users, as well as the rapid proliferation of servers and operating systems. What is intendeed on this project is to reduce datacenter management complexity by combining lifecycle management automated functionalities and all the contingency measures to save the datacenters integrity."

"Els requisits del Negoci que requereixen un gran creixement generen major complexitat en els Centres de Còmput, són els administradors qui necessiten gestionar el creixent volum de dades, aplicacions, i usuaris, així com la ràpida proliferació dels servidors i els diferents sistemes operatius. En aquest projecte es pretén reduir la complexitat en la gestió dels Centres de Còmput, combinant l'automatització de la gestió del cicle de vida i totes les mesures de contingència necessàries per a mantenir la integritat dels mateixos."