Managing Data Centers With Sun™ Management Center Change Manager

John S. Howard, Enterprise Engineering

Sun BluePrints™ OnLine—October 2002

http://www.sun.com/blueprints

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95045 U.S.A.
650 960-1300

Part No. 817-0183-10
Revision 1.0, 10/01/02
Edition: October 2002

Please
Recycle

Managing Data Centers With Sun™ Management Center Change Manager

Deploying and updating software are two of the most challenging and time consuming tasks facing data center managers. The Sun™ Management Center (SunMC) Change Manager provides a framework and tools for data center personnel to quickly and efficiently deploy, replicate, update, and manage software over a large number of systems.

This paper presents techniques and best practices for using SunMC Change Manager. The software utilizes Solaris™ Flash archives and Solaris™ Live Upgrade (LU) technology to manage software stacks. This paper details the following topics:

- Overview of Solaris Flash archives and LU technology
- Creation of software stacks
- Uses of the SunMC Change Manager deployment engine
- Provisioning and reprovisioning of systems
- Installation and management of software patches

Overview of Solaris Flash Archives and Live Upgrade Technology

LU 2.0 software and later versions allow you to use a Solaris Flash archive to install an alternate boot environment (ABE). The following sections provide a high-level overview of the Solaris Flash technology and detail the LU flash installation procedures.

Overview of Solaris Flash Technology

Solaris Flash technology provides a mechanism by which you can archive a specific or reference installation of the Solaris™ Operating Environment (Solaris OE). You can then use that archive to install the Solaris OE. The reference installation is created from the on-disk Solaris OE, which includes all installed software. This system is designated as the master machine. The reference installation can be a Solaris OE installed by any means, for example, with JumpStart™ software, from CD, or by an interactive installation.

After you identify the master machine, capture the reference installation in a Solaris Flash archive. A central feature of Solaris Flash software, this archive is essentially a point-in-time snapshot of the Solaris OE, software patches, and applications on the master machine. To create an archive, execute the `flarcreate`(1m) command on the master machine.

Solaris Flash extensions enable you to install an archive from a Network File System (NFS) server, a Hypertext Transfer Protocol (HTTP) server, or a traditional JumpStart server. Additionally, you can access the archive from a disk device (including CD-ROM) or from tape device that is local to the installation client. When you install an archive, it is transmitted over the network to the installation client and is written to the disk. After the archive is written to the installation client's disk, any necessary archive modifications are performed. For example, configuration files on the installation client, such as the `/etc/nsswitch.conf` file, might need to vary from the file on the master machine. The Solaris Flash mechanism enables you to automate modifications and allows for differences in kernel architecture or device differences between the master machine and the installation client.

Additionally, Solaris Flash software enables the automatic resolution of partitioning differences between the master machine and the installation client. For example, if an archive was created on a system with a single root (`/`) partition, and the installation client has separate `/` and `/var` partitions, the archive automatically customizes itself to the installation client. Remember, the installation client partitioning must be correctly specified in the JumpStart software profile.

A flash archive is a snapshot of a system and, as such, includes all specified files on a system. If an archive is created from a system that is in use, you will need to clean up or zero out some files after the flash archive is installed. Examples of these types of files include log files, such as those found in `/var/adm`, and any files in the `/var/tmp` directory.

Modify the finish script to zero out log files after installation of the Solaris Flash archive. To exclude temporary directories, such as the `/var/tmp` directory, exclude the directory when you create the flash archive. See "Inducing System Amnesia" on page 8 for additional details.

Create the flash archive after installing all software, but before placing the system into production. Depending on the software installed and the intended use for the system, you might need to create the flash archive after installing the software, but before configuring it. For example, you should create archives for database servers or Lightweight Directory Access Protocol (LDAP) servers after installing the database management software, but before creating and populating the databases.

Installing the Solaris OE with a flash archive can be dramatically faster than with other mechanisms, depending on network traffic and disk speeds.

You can find further details on the use of Solaris Flash software in the *Solaris 9 OE Advanced Installation Guide* and the Sun BluePrints™ book, *JumpStart Technology: Effective Use in the Solaris Operating Environment* by John S. Howard and Alex Noordergraaf (ISBN 0-13-062154-4).

Overview of Live Upgrade Software

LU 2.0 software was introduced with the Solaris 8 10/01 OE (Update 6). On the Solaris 8 10/01 OE media, LU 2.0 packages are located in the Easy Access (EA) directory of the Solaris media CD marked “Software 2 of 2.” With the Solaris 8 01/02 OE (Update 7), LU 2.0 software was moved from the EA area to the product area, and is bundled with the OE packages.

LU 2.0 software was also released as a web release (08/01) that is available at `http://www.sun.com/solaris/liveupgrade`. LU 2.0 software works with, and can be installed on, all releases of the Solaris OE versions 2.6, 7, 8, and 9. LU 2.0 software is the first general-availability release of the software. It is recommended over the use of LU 1.0 software, which must never be used in a production environment or on a production server.

Creating and Managing Boot Environments

The concept of a boot environment (BE) is central to the operation and implementation of LU software. A BE is a group of file systems and their associated mount points. LU software uses the term “boot environment” instead of “boot disk” because a BE can be contained on one disk, or can be spread over several disks. LU provides a command-line interface and a character-based user interface (CUI) to create, populate, manipulate, and activate BEs.

Note – The CUI has a few restrictions: it is neither localized, nor internationalized. Also, the existing CUI does not provide access to the full functionality of the LU software.

You can create BEs on separate disks, or you can create them on the same disk; however, a single root (/) file system is recommended for the Solaris OE.

The active BE is the one that is currently booted and active; all other defined BEs are considered inactive. Inactive BEs, are referred to as ABEs (alternate BEs).

BEs can be completely self-contained, or they can share file systems. Only file systems that do not contain any OE-specific data and that must be available in any OE should be shared among BEs. For example, users' home directories on the `/export/home` file system are good candidates to share among several BEs.

If you used multiple file systems for the Solaris OE, such as separate file systems for `/kernel`, `/usr`, `/etc`, `/`, and so forth, do not share OE-specific file systems among BEs. In addition, do not split certain file systems (such as `/kernel`, `/etc`, `/dev`, or `/devices`) from `/`. If you split them onto a separate file system from `/`, the BE that is created might not be bootable.

Additionally, LU provides a mechanism to synchronize individual files among several BEs. This feature is especially useful for maintaining files such as `/etc/passwd` in one BE and then propagating changes to all BEs.

To back up BEs created with LU, use the `ufsdump` or `fssnap` commands. Consult the man pages for information about the uses of these commands.

Upgrading Systems

To appreciate the value of using LU software to upgrade a system, consider the common situation of having to upgrade a production server from the Solaris 8 OE to the Solaris 9 OE. Most likely, you could not take the server down to do the upgrade. Additionally, site change control procedures likely require that you provide a back-out plan to restore the initial Solaris 8 OE in the case of any unforeseen upgrade failures or software incompatibilities. Using LU, you can complete this upgrade while the Solaris 8 OE is up and live. The LU framework also provides for the retention of the Solaris 8 OE as a fallback in case of a failure during the upgrade procedure.

▼ To Upgrade a System Using LU Software

1. **Create and populate a new BE by cloning the current OE.**
2. **Upgrade the new BE.**
3. **Install (or upgrade) unbundled software, patching as necessary, in the new BE.**
4. **When you are ready to cut over to the new version of the OE, activate the new BE and reboot into the new BE.**

Rather than using slice 7 of the boot disk for the `/export` file system, use this slice for the clone OE or as an ABE.

Installing Flash Archives With LU 2.0 and Solaris Flash Software

When using Solaris Flash with LU 2.0 software, the specified ABE is not upgraded; instead the contents of the flash archive are extracted and installed in the specified ABE.

▼ To Install a Flash Archive Using LU and Solaris Flash Software

1. **Create and populate a new BE by cloning the current OE.**
2. **Upgrade the new BE to Solaris 8 OE 10/01 using a flash archive.**
3. **Activate the new BE.**

Creating Software Stacks

The combined tasks of installation and configuration are typically encompassed by the term hardware and software integration. Integration is typically concerned not only with correctly installing and configuring a software product and binding it to a hardware platform, but installing and configuring several software products to correctly function and interoperate with each other. The result of integrating several software products is typically referred to as an integrated software stack, or simply, a software stack.

The use of a software stack helps diminish the total cost of ownership (TCO) of a system. With a software stack, the configured system implements published best practices and site standards that have been validated before the system is deployed in the data center.

The use of a software stack ensures that all such systems are identically configured. This consistency leverages the experiences of the data center personnel. Their experiences on one system are now applicable across many systems. This helps decrease the TCO as well as speeding system recovery in the event of a failure. In the

case of a software load, each system must be individually configured by data center personnel. Even if an attempt is made at automating this task, the possibility of human error still exists and system consistency might be lost.

In order to achieve maximum flexibility and robustness of software stacks, a rigorous separation of software installation and configuration information should be maintained at all times.

The SunMC Change Manager uses a customizable flash archive to implement and deploy software stacks. The system that is used as the archetype from which the software stack is created is referred to as the master system.

Selecting the Master System

For most systems, a software stack should be created shortly before the system goes into production. For example, a database server or LDAP server should have its archive created after the database management software has been installed, but before the databases have been created and populated.

Software that stores configuration information outside of a UNIX® file system might not be correctly configured on the flash installed client. For example, logical volume management software like VERITAS Volume Manager (VxVM) or Solstice DiskSuite™ software store metainformation (such as logical volume layout and RAID configuration information) in raw partitions, outside of a file system. Installing a system from a flash archive created on a master machine that uses VxVM with an encapsulated and mirrored boot disk is not possible. Because archive creation does not (and cannot) access the metainformation in the VxVM private regions, any subsequent installation from that archive would be incomplete and unbootable. The configuration of system software such as this is specified by the SunMC Change Manager parameters file. The configuration of software is done by processing this parameters file after the flash archive is installed.

To implement software stacks in a flash archive, create the archive on the master machine after installing all software, but before configuring the software. Using the example of a master machine with a VxVM encapsulated and mirrored boot disk, you would create the flash archive after installing the Solaris OE, after adding VxVM packages, and before executing `vxinstall` to configure the VxVM software. In this example, you can configure the VxVM installation client, including encapsulating and mirroring the boot disk, from a finish script after the flash archive is installed. The values required to complete this configuration, such as the disk to use as the root mirror, are specified in the parameters file as user defined keyword and value pairs.

Building Software Stacks

When selecting a system to be used as the master system and when building the software stack, pay attention to the types of hardware where the stack will be deployed. All software that might be necessary on the installation clients must be contained in the software stack.

For example, consider a Peripheral Component Interconnect (PCI)-based system selected as a master system. Depending on the choices made at installation time, the SBus driver software might not have been installed. Consequently, any resulting software stacks created from this system will not have the SBus drivers available and any SBus hardware will be unavailable to the installation client.

As another example, consider a flash archive created on a Sun Fire™ 15K domain. In most instances, the domain will not have a graphics frame buffer installed, and consequently, no drivers for any graphic frame buffers. This will not prohibit the flash archive from being correctly deployed onto a wide range of platforms. However, if one of those platforms is a Sun Blade™ 1000 workstation, the graphics monitor and frame buffer will not be available due to the lack of frame buffer drivers in the flash archive. To avoid this issue, either:

- Ensure that all possible drivers and Solaris OE software that might be needed on any potential client are on the master system (and in the flash archive) or that any missing software is installed from a JumpStart finish script after the flash archive is installed.
- Or, deploy the software stack to only those systems that are appropriate for that stack.

The first approach is the recommended solution. You can easily install all Solaris OE software by installing the Entire Distribution plus OEM Software (`SUNWCXall`) package meta-cluster, as well as any third-party or specialized device drivers, on the master system.

Unconfiguring Software

Just as some software applications require specific information and procedures to complete their configuration, some software applications have specific de-installation and unconfiguration procedures. Typically, this unconfiguration can consist of removing host specific information, such as host or device names, from configuration files.

Correctly unconfiguring software might be necessary before creating a flash archive. Unconfiguration is necessary to help ensure that the software stack is completely generalized and does not contain any host specific information from the master system. The procedure of unconfiguring software is commonly referred to as inducing system amnesia.

Inducing System Amnesia

A flash archive created for deployment on many systems needs to be given amnesia—it needs to lose or forget its identity. To induce system amnesia, run the `flarcreate`(1m) command, which essentially runs the `sys-unconfig`(1m) command on the master system's flash archive. For most of the Solaris OE, this is sufficient. However, there are a number of other applications that do not register with the use of the `sys-unconfig` command. (For details, consult the `sysidconfig`(1m) man page.) A flash archive containing such applications might require additional work to remove traces of the master system's identity before the flash archive is created. A list of items to consider are:

- **Configuration files.** Some applications store their configuration information in files, which might not be cleared by the `sys-unconfig` command. Of particular note are configuration files that contain authentication or authorization information.
- **Log files.** Often, applications write identifying information to log files. This might include host names, Internet Protocol (IP) addresses, user names, and so forth. Clean these files to ensure that the clone system does not have log records from the master system. Examples of log files include:
  - `/var/adm/lastlog`
  - `/var/adm/messages.*`
  - `/var/adm/sulog`
- **State files.** Some applications might retain state information in files. These could include files used to flag events or configuration files. If application state information is retained in files, reconcile these files on the master system prior to executing the `flarcreate` command.
- **Backup files.** Some applications that modify files create backup copies of the files before modifying them. For example, the `useradd`(1m) command creates backup copies of the `/etc/passwd` and `/etc/shadow` files. If these backup files exist, and if they contain information that identifies the master system, reconcile them.
- **Temporary files.** Some applications create temporary files that are intended to be persistent across reboots. These files might be placed in spool directories or in application-specific directories. In particular, exclude the `/var/tmp` directory from the flash archive or empty it before including it in the flash archive.
- **Queue files.** Some applications copy files or data to a queue directory. Examples include the `sendmail`(1m) file and the Solaris print service. These queue directories are not cleared by `sys-unconfig`. Clear these directories of data files before creating the flash archive.
- **Mail subsystem files.** The `sys-unconfig` command does not clear the `/var/mail` directory or user mail files therein. Clear the user mail files from this directory before creating the flash archive.
- **System accounting information.** System accounting information might not be cleared by the `sys-unconfig` command. If this is the case, clear the accounting data from the system accounting directory (typically `/var/adm/sa`).

It is also important to keep in mind that if any locally developed applications or tools utilize any of the preceding file types, those files must also be cleaned. To help enable locally developed system applications and tools to automatically clean up after themselves on a re-configuration boot, register them with the `sysidconfig` command. Consult the `sysidconfig`(1m) man page for details about registering applications.

Deploying Software Stacks

The SunMC Change Manager provides a framework that helps enable the rapid deployment of software stacks. These software stacks are managed and deployed in the form of flash archives. When used with LU, the SunMC Change Manager framework enables you to deploy and manage software stacks while the managed systems (for example, the installation target or clone systems) are up and running, even within a production environment. Further, the framework provides a grouping mechanism with which you can group similar managed systems together and manipulate them as one. This grouping mechanism helps to ensure that systems performing similar functions are not only installed identically, but that configuration changes to any of those systems are made to all systems in the group. This mechanism is a powerful tool to help prevent configuration drift among systems.

Component Systems of the SunMC Change Manager Implementation

The SunMC Change Manager implementation utilizes three distinct types of systems:

- **SunMC Change Manager server.** The system that runs the SunMC Change Manager and also acts as a repository for flash archives (software stacks), profiles for installing those stacks, and configuration files.
- **Managed systems.** Systems where software stacks are deployed. The SunMC Change Manager controls these systems by remotely executing and monitoring commands for the installation, upgrading, and auditing of software and their systems.
- **Master systems.** Systems used as templates to build software stacks. Master systems must have SunMC Change Manager client software packages installed on them to ensure that software stacks contain the components required to deploy and manage the stacks.

Component Software of the SunMC Change Manager Server

The SunMC Change Manager server provides a Solaris boot image to the managed systems. This boot image is the hardware architecture neutral Solaris OE miniroot. For initial installations, the miniroot boots the managed host and begins the installation.

The SunMC Change Manager also provides a software stack audit tool to create inventories of software stacks running on clone systems. This tool helps validate the contents of deployed stacks by comparing the contents of a managed host's file systems (the deployed and installed software stack) to a known good reference configuration. These auditing features are referred to as the `bart` subsystem. For details on the usage of the auditing tool, consult the `bart`(1m) man page.

Provisioning and Reprovisioning Systems

In addition to using the SunMC Change Manager to rapidly deploy or provision systems, you can use it to reprovision systems. For example, if a farm of web servers has a hardware failure disabling one of its servers, you could use the SunMC Change Manager to rapidly reprovision a host from a free hardware pool to replace the failed servers.

You can also use the software to implement rolling upgrades, that is, software upgrades of a large number of systems. You can deploy upgrades to managed systems and to systems that are scheduled to reboot into the upgraded system at a convenient time. This approach helps ensure a smooth transition to the new software, with minimal impact to the end user.

Whether you use it to provision or reprovision systems, the SunMC Change Manager server uses a shared profile template to drive the software installation or upgrade. A shared profile is a configuration profile that can be used for multiple managed systems or for groups of managed systems. The shared profile is used to specify configuration information such as disk partitioning, name services, and network interfaces. This is essentially information that will be passed on to the JumpStart framework to build the `sysidcfg` file and the JumpStart profile. You can create these profile templates using an ASCII text editor or through the SunMC Change Manager graphical user interface by selecting parameters and then supplying relevant values for those parameters.

Additionally, you can use a system profile to specify per system information. The values specified in this profile override or complete information from the shared profile, and are applied only to the specified individual host.

The following example shows the shared profile template and system profile template used to install a software stack on a system named `barossa`.

Note – These profile templates use the JumpStart framework profile keyword `rootdisk`. Using this keyword avoids the need to know the Small Computer Systems Interface (SCSI) or fibre disk configuration of the target system.

The system disk layout specified by the shared template is suitable for a 36-gigabyte disk:

- Root device: `rootdisk.s0`
- Root size: 8192
- Swap device: `rootdisk.s1`
- Swap size: 2048

These templates also specify that an LU ABE is to be created when the software stack is installed. The ABE location information specified is:

- ABE root device: `any`
- ABE root size: 8192

The shared profile used to install barossa is as follows.

```
# sharedProf.template

#Wed Jul 29 13:06:44 PDT 2002
base_config_be_1_root_size=8192
base_config_sysidcfg_nameservice=NIS
base_config_sysidcfg_terminal=dtterm
base_config_be_1_root_device=any
base_config_sysidcfg_domainname=EE_Lab.West.Sun.COM
base_config_sysidcfg_ipv6=NO
base_config_sysidcfg_rootpw=4KjH13s4L0do2
base_config_sysidcfg_timezone=US/Pacific
PERMS=rwx
base_config_be_0_root_size=8192
base_config_be_0_swap_device=rootdisk.s1
base_config_sysidcfg_systemlocale=C
base_config_sysidcfg_netmask=255.255.255.0
FNAME=default.foubar
base_config_flar_archive=/jumpstart/FlashArchives/s9-server.flar
base_config_be_0_swap_size=2048
base_config_sysidcfg_timeserver=localhost
base_config_sysidcfg_defaultroute=FIND_ONE
CONTENTS=changed in shared profile
base_config_sysidcfg_networkinterface=PRIMARY
base_config_be_0_root_device=rootdisk.s0
base_config_sysidcfg_dhcp=NO
```

The profile template used to install barossa follows. However, barossa has only an 18-gigabyte disk drive, so this profile overrides the following specifications from the shared template:

- Root size: 4096
- ABE root device: rootdisk.s5
- ABE root size: 4096
- Network timeserver host: timehost

The specifications that have been overridden are in bold.

```
# barossaProf.template

#Wed Jul 29 13:42:29 PDT 2002
base_config_be_1_root_size=4096
base_config_sysidcfg_nameservice=NIS
base_config_sysidcfg_terminal=dtterm
base_config_be_1_root_device=rootdisk.s5
base_config_sysidcfg_domainname=EE_Lab.West.Sun.COM
base_config_sysidcfg_ipv6=NO
base_config_sysidcfg_rootpw=4KjH13s4L0do2
base_config_sysidcfg_timezone=US/Pacific
PERMS=rwx
base_config_be_0_root_size=4096
base_config_be_0_swap_device=rootdisk.s1
base_config_sysidcfg_systemlocale=C
base_config_sysidcfg_netmask=255.255.255.0
FNAME=default.foubar
base_config_flar_archive=/jumpstart/FlashArchives/s9-server.flar
base_config_be_0_swap_size=2048
base_config_sysidcfg_timeserver=timehost
base_config_sysidcfg_defaultroute=FIND_ONE
CONTENTS=changed in shared profile
base_config_sysidcfg_networkinterface=PRIMARY
base_config_be_0_root_device=rootdisk.s0
base_config_sysidcfg_dhcp=NO
```

Installing and Managing Software Patches

In addition to using it to install or upgrade system software, you can use the SunMC Change Manager to manage and deploy software patches.

To deploy a patch or set of patches using the SunMC Change Manager, install, verify, and validate the patches on a single system. The verification and validation of the patches on a single system is very important. At this time, you should validate not only that patches address the issue for which they were designed, but also that they do not negatively interact with other third-party software and that they do not negatively affect software performance.

After you have tested and validated the patches, use the patched system as a master system for creating a software stack that contains the system software, any other installed software, and all installed patches. You can then deploy this software stack on all similar systems or on all systems that require the patched software stack.

Using the `bart` auditing tool, you can deploy the patched software stack automatically. You can also use the tool to determine which systems need the patched software stack, but do not yet have the patched software stack installed. After making this determination, schedule the patched software stack for installation on the necessary systems. You can schedule this installation manually or automatically.

It is also important to keep in mind that because LU technology is being used, you can deploy the patched software stack to an ABE. If you later determine that a problem exists with the patched software stack, the previous software stack is still available in the previously activated BE. This fact might also be used to mitigate exposure to risk during the deployment of new versions of system software or application software.

About the Author

John S. Howard is a Senior Staff Engineer with Sun's Enterprise Engineering group. He has over 19 years experience in software engineering and systems administration on a diversity of platforms. John is currently working on projects for enhancing system availability and serviceability.

John is the author of numerous technical papers and co-author of the books "JumpStart Technology: Effective Use in the Solaris™ Operating Environment" and "Boot Disk Management: A Guide for the Solaris™ Operating Environment."

Acknowledgements

The author would like to thank Julie Snow, Richard Elling, and Barry Greenberg for their suggestions, comments, and criticisms of this paper.

Ordering Sun Documents

The SunDocs℠ program provides more than 250 manuals from Sun Microsystems, Inc. If you live in the United States, Canada, Europe, or Japan, you can purchase documentation sets or individual manuals through this program.

Accessing Sun Documentation Online

The `docs.sun.com` web site enables you to access Sun technical documentation online. You can browse the `docs.sun.com` archive or search for a specific book title or subject. The URL is `http://docs.sun.com/`

To reference Sun BluePrints OnLine articles, visit the Sun BluePrints OnLine Web site at: `http://www.sun.com/blueprints/online.html`