ISCSI Boot

Functional Specification

Version: 1.0

1 Project Description

This project is to enable Solaris to boot off iSCSI disk via network adapters rather than iSCSI adapters. Different approaches, iBFT/OBP, are adopted to implement this feature on x86/sparc platforms. This case supersedes 'iSCSI Software boot'(PSARC/2007/450).

1.1 Definition

iSCSI boot is a process whereby the operating system is initialized from a remote storage disk array across a storage area network (SAN) rather than from the locally attached hard disk drive. The functionality reduces the datacenter's total cost of ownership by contributing to lower power consumption, better solution performance, localized patch management and real estate cost savings. Currently Solaris is able to do iSCSI boot only with expensive/dedicated iSCSI adapters , and this project aims to enhance Solaris to use network adapters to archive the same funtionality.

Generally to load the booter and ramdisk for Solaris an initial iSCSI and TCP/IP network stack is needed in firmware of NIC and/or motherboard. On x86 this kind of firmware is called iBF (iSCSI Boot Firmware), and BIOS is able to loader necessary data via iBF to start the booter and then the OS. The iBF also capitulates parameters of iSCSI boot connection into a table (iBFT, iSCSI Boot Firmware Table) and stores the table into a piece of low memory. OS can use this table to configure its own iSCSI connection and then continue to initialize self until fully starts up. On SPARC, OBP will implement the iSCSI stack and store parameters of boot connection in OBP properties. So on SPARC the only difference in Solaris OS is to read OBP properties instead of scanning low memory.

Solaris automated installer will be supporting iSCSI disk to facilitate the deployment of Solaris in iSCSI environment.

A previous case, PSARC/2007/450, is superseded by this project. Major enhancements from the previous one include,

1. Support x86 only, the new case supports both x86 and sparc
2. Lack of coordinations with installer team, the new case has good coordinations with

installer team

3. Lack of security support, the new case supports CHAP for authentication
4. Dedicated dhcp/boot/bootparams server, the new case don't have any that kind of servers
5. Lack of MPxIO support, the new case supports MpxIO
6. Utilities will be improved to prompt user for negative operations against the boot connection
7. Stmsboot will be supporting iSCSI

The previous case didn't deliver anything to Solaris.

1.2 Motivation, Goals, and Requirements

iSCSI boot is required on Solaris in FY09, both x86 and SPARC platforms, because:

1. Sun customers are requesting iSCSI boot options on our Ethernet cards and Storages
2. Intel has iSCSI boot option on their standard NIC for Windows and Linux OS, and Sun can offer this today if we have iSCSI boot on Solaris
3. iSCSI boot is supported on Linux and Windows; therefore we need to reach parity on Solaris
4. iSCSI boot will be the replacement for PXE boot
5. The plan allows iSCSI boot on the standard Network cards without using expensive TOE HBAs.

1.3 Changes From the Previous Release

N/A

1.4 Program Plan Overview

1.4.1 Development

Development is going in Beijing. iBFT on x86 has been prototyped and the design is complete.

- Stmsboot changes
- Solaris automated installer

1.4.2 Quality Assurance/Testing

Dedicated QE resource will be assgined to ensure the quality of this project. Test consists of,

- Boot from Intel NIC with iBFT support on x86

- Boot from OBP with iSCSI support on Sparc
- Install Solaris on iSCSI disk with new installer
- Switch between MPxIO and non-MPxIO by stmsboot

1.4.3 Documentation

System Administration Guide: Devices and File Systems (Chap. 14, Configuring Solaris iSCSI Targets and Initiators)

1.4.4 Release Cycle

We plan to integrate the iSCSI boot on x86 in build 102 as phase I, and the support on Sparc along with the change in installer and stmsboot in build 112 as phase II.

1.4.5 Technical Support

TBD

1.4.6 Training

TBD

1.5 Related Projects

1.5.1 Dependencies on Other Sun Projects

iSCSI Openboot support

Solaris automated installer

PSARC/2003/126 iSCSI project

PSARC/2004/454 Solaris Boot Architecture

PSARC/2006/525 new-boot sparc

1.5.2 Dependencies on Non-Sun Projects

N/A

1.5.3 Sun Projects Depending on this Project

None

1.5.4 Projects Rendered Obsolete by this Project

None

1.5.5 Related Active Projects [Describe the relationship.]

None

1.5.6 Suggested Projects to Enhance this Program

None

1.6 Competitive Analysis

Currently Solaris is unable to be boot-off from iSCSI disk. This is a drawback which limits Solaris' competency in iSCSI SAN enviroment, diskless clients setup and so on. Microsoft and several Linux distributions are already supports iSCSI boot with iBFT so Solaris is significantly behind them. Sun needs this effort to pace up with them in this area.

2 Technical Description

2.1 Architecture

The iSCSI boot project consists of many different aspects. Some of them will be illustrated in section 4 and beyond.

- Overall the major modification for iSCSI boot is against the 'Kernel' stage of Solaris booting. After starting up Solaris kernel adopts iSCSI boot properties, configure the NIC/iSCSI boot disk then finally mount the rootfs. Most parts are common on x86 and sparc except the way to load iSCSI boot properties.

- On x86, iBFT is the way to pass the boot info to the kernel. IBFT is a part of ACPI 3.0b specification. In the view of Solaris kernel iBFT is a piece of memory at a fixed area and identified by a magic signature. For details about this part please refer to section 4.

- On sparc, OBP will be responsible for loading booter/kernel and passes the boot properties with standard OBP properties. For details about the OBP part please refer to section 5.

- CHAP (Challenge-handshake authentication protocol, RFC1994) is supported to ensure the data iSCSI initiator received for booting comes from the target it claims to have come from. For booting as the data being transmitted are mostly OS standard, the most important thing there is to choose the RIGHT target, therefore CHAP is the key here. IBFT supports CHAP and various vendors already support that (e.g., Intel 10G/1G series). OBP will also be supporting CHAP on iSCSI to ensure equivalent protection on sparc. Solaris software iSCSI initiator and mostly all iSCSI target also supports CHAP for authentication. IPsec in initiator side is not available during the boot but will take effect after Solaris fully starts up. This is mainly due to several userland daemon and configuration files are not available during boot.

  CHAP resides in IBF's firmware on x86 and can only be configured from there. It will also be one of OBP's iSCSI boot properties and reside there.

- Dump with iSCSI stack will not be supported in this project. Users must configure a local disk or something else which is capable as the dump device if they need to capture OS crash dump. This is a compromised decision, actually there are lot of options available to archive this goal or as a work around.

  a) Twist Solaris TCP/IP stack along with sockfs and GLDv3, able to do polled I/O

  b) Implement a small/private network stack in iSCSI software initiator, to do polled I/O

  These two solutions will lead the dump code into a complicated path and conflicts with the situation in panic which some part of the OS is not reliable. Generally the support to dump should be as simple as possible. Another reason is that they needs new interfaces from GLDv3 which may be impractical in near team.

  c) Relies on NIC vendor's support on their driver/firmware.

  This is kind of hacking, and also means a long-term special support from vendor. Also so far there is no such a support found in any NIC vendor.

  d) Using simpler network protocols (e.g. UDP) to do dump on network.

  This may be a good choice but currently no available solution in Solaris. Abandoning TCP in iSCSI stack is mostly impractical (needs target support) but more, this should be a general purpose to do network-based dump in Solaris, not only for iSCSI. Therefore it needs another case to evaluate this solution and the project team consider it out of the scope for this project.

  However this project will not make the situation worse. Solutions for similar requirements(e.g., diskless client) will also work with iSCSI boot.

- Solaris new Installer will be modified to prompt user configure iSCSI disk and then install Solaris onto it. Solaris legacy installer is able to work with iSCSI disk if the user breaks installation and configure iSCSI disk in shell. Refer to seciton 6 for details.

- On sparc, Solaris installer will interact with OBP to set/correct OBP properties related to iSCSI boot.

- Solaris iSCSI software initiator will be updated with 'scsi-self-identify' class to enable dynamic discovery for boot device.

- Solaris iSCSI software initiator will be modified to now allow to offline the last session with boot target.

- Stmsboot will be modified to support iSCSI in addition to fp and mpt.

- ISCSI boot of Xen Virtual machine is out of the scope of this project, and it is already done with Solaris hypervisor.

2.2 Interfaces

2.2.1 Exported Interfaces

None.

2.2.2 Imported Interfaces

Interface Name	Proposed Stability Classification	Specified in What Document?	Former Stability Classification or Other Comments
psm_map_new, psm_unmap	Consolidation private	PSARC/1995/422	None
ddi_prop_lookup_string	Standard	Solaris man page	None
iBFT	Standard	ACPI 3.0b specification	Microsoft License

2.3 User Interface

To configure iBFT, UI relies on implentations for iBF from different vendors. Usually it is text based.

To configure OBP, UI is the CLI provided by current OBP implementation.

To installation Solaris on iSCSI disk, the legacy installer needs no changes on UI. The new installer will be updated to allow the user input info about the iSCSI initiator/target. The design of this part is TBD.

2.4 Compatibility and Interoperability

2.4.1 Standards Conformance

On x86 the iBFT is used as the interface. It is defined in the ACPI 3.0b specification please refer to appendix for details.

2.4.2 Operating System and Platform Compatibility

Solaris Nevada on Sparc will be supported with updated OBP.

Solaris Nevada on x86 will be supported if the system has iBF on NIC's/mainboard's ROM.

2.4.3 Interoperability with Sun Projects/Products

Solaris can boot from Sun's iSCSI array and software iSCSI target.

2.4.4 Interoperability with External Products

Solaris can boot from iSCSI arrays from other vendors using standard iSCSI protocol. No special requirements from target side.

2.4.5 Coexistence with Similar Functionality

No similar functionality found.

2.4.6 Support for Multiple Concurrent Instances

Only one iSCSI boot disk will be presented during boot.

2.4.7 Compatibility with Earlier and Future Releases

N/A.

2.5 Performance and Scalability

2.5.1 Performance Goals

This project should have no effect on iSCSI IO path, and we expect no regressions there.

2.5.2 Performance Measurement

TBD.

2.5.3 Scalability Limits and Potential Bottlenecks

Scalability is not an issue for this project. Solaris only needs one boot device.

2.5.4 Static System Behavior

No dedicated logging file/database for boot support.

2.5.5 Dynamic System Behavior

System behaviors will be similar with one booted from local disk.

2.6 Failure and Recovery

2.6.1 Resource Exhaustion

User may lose the access to the root disk if the system is unable to allocate iSCSI packets due to low memory.

2.6.2 Software Failures

We don't have plan on software failures.

2.6.3 Network Failures

User may lose the access to the root disk if the network is down. ISCSI software initiator can reestablish the connection to iSCSI disk based by retry mechanism.

2.6.4 Data Integrity

File system should be responsible to ensure the data integrity.

2.6.5 State and Checkpointing

N/A.

2.6.6 Fault Detection

N/A.

2.6.7 Fault Recovery (or Cleanup after Failure)

N/A.

2.7 Security

For booting, it is loading OS specific data so the important thing is to make sure that the data come from the authenticated server/target. Solairs iSCSI boot uses CHAP (Challenge-handshake authentication protocol, RFC1994) to do ensure the data iSCSI initiator received for booting comes from the target it claims to have come from.

CHAP is a part of iBFT standard and is supported by various vendors in iBF(e.g., Intel 10G/1G series). OBP will also have properties for CHAP to ensure equivalent protection on sparc. Solaris software iSCSI initiator and mostly all iSCSI target also supports CHAP for authentication.

IPsec in initiator side is not available during the boot but will take effect after Solaris fully starts up.

2.8 Software Engineering and Usability

2.8.1 Namespace Management

We introduce no new packages or namespaces.

2.8.2 Dependencies on non-Standard System Interfaces

N/A.

2.8.3 Year 2000 Compliance

This project doesn't have this issue.

2.8.4 Internationalization (I18N)

N/A.

2.8.5 64-bit Issues

This project is 64-bit safe as a part of Solaris kernel/drivers.

2.8.6 Porting to other Platforms

N/A.

2.8.7 Accessibility

All interfaces except the installer are CLIs. Interfaces in installer will follow the way how it works today.

3 Release Information

3.1 Product Packaging

Bundled with Solaris.

3.1.1 Package Overview

No new packages will be introduced.

3.1.2 (Default) Installation Locations

SUNWcakr.i

- platform/i86pc/kernel/unix
- platform/i86pc/kernel/amd64/unix

SUNWcakrx.i

- platform/i86xpv/kernel/unix
- platform/i86xpv/kernel/amd64/unix

SUNWcakr.u
- platform/sun4u/kernel/sparcv9/unix

SUNWckr
- kernel/fs/sparcv9/sockfs
- kernel/fs/sockfs
- kernel/fs/amd64/sockfs
- kernel/misc/sparcv9/strplumb
- kernel/misc/strplumb
- kernel/misc/amd64/strplumb

SUNWiscsir
- kernel/drv/sparcv9/iscsi
- kernel/drv/iscsi
- kernel/drv/amd64/iscsi

SUNWcsu
- usr/sbin/stmsboot

3.1.3 Effect on External Environment

None.

3.2 Installation

3.2.1 Installation procedure

Binaries are installed with normal Solaris installation.

To install Solaris on an iSCSI disk,

- With legacy installer, user can exit the installer to shell before selecting the disk, use 'iscsiadm' to configure iSCSI disks and then continue the installation onto them.
- The project team has a draft out there to modify both LEGACY installer and Caiman. Details are in section 6.

3.2.2 Effects on System Files

None.

3.2.3 Boot-Time Requirements

User need to configure the iBFT on x86 or OBP on sparc.

3.2.4 Licensing

Sun Solaris/OpenSolaris

iSCSI BOOT FIRMWARE TABLE (iBFT) SPECIFICATION LICENSE, Microsoft Inc.

3.2.5 Upgrade

None.

3.2.6 Software Removal

Done with normal package tools.

3.3 System Administration

Administrator needs to learn how to configure iBFT from different vendors by referring their manuals. The way to configure iBFT can be very different in different iBF, classified by vendors.

Administrator also needs to learn how to configure OBP to support iSCSI boot.

An administration guide will be created to cover there cases.

Stmsboot will be changed to support iSCSI(currently it supports fp and mpt), administrator needs to refer updated man page for stmsboot.

4 iBFT Architecture

4.1 Description

iBFT(iSCSI Boot Firmware Table) is defined in ACPI 3.0b specification and is a block of information that contains various parameters that are useful to the iSCSI Boot process. The iBFT is the mechanism by which parameter values in iSCSI Boot Firware(iBF) are conveyed to the operating system. The iBF builds and fills in the iBFT, and iBF could be one of System ROM, Adapter ROM and Network Boot Program (NBP). The iBFT can be located by the Low RAM Method, for the table header signature in system memory between 512K and 1024K.

For Solaris kernel by scanning the low memory, it is able to know if it is doing an iSCSI boot and loading necessary paramesters then. It is iBF's responsibility to present the iSCSI disk to load OS boot loader and then the ramdisk.

4.2 Interfaces

4.2.1 User-visible

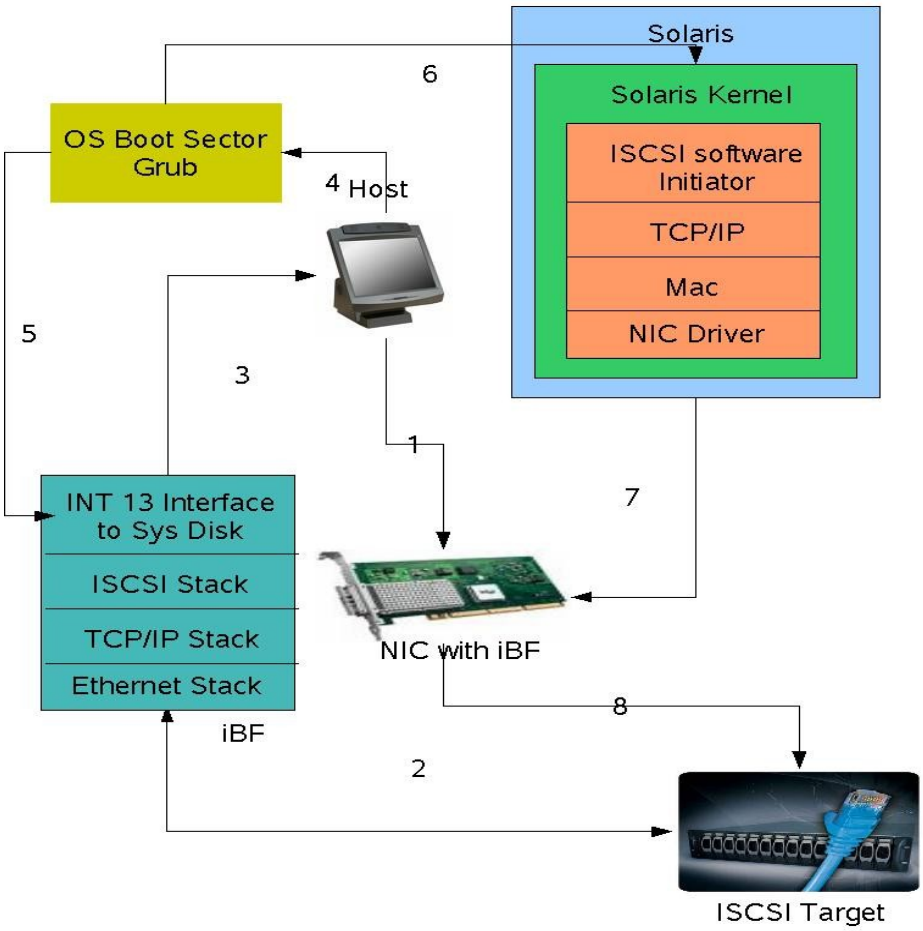
IBF interfaces to configure iBFT.

4.2.2 Internal (optional for ARC review)

IBFT.

Unified global iSCSI boot properties for x86 and sparc.

4.3 Operation



A step by step description of the boot procedure is as

1. Host is powered up/resetting and the iBF is loaded
2. iBF initializes and connects to the iSCSI target, presents iSCSI disk to BIOS
3. BIOS uses INT 13 to load MBR and OS boot sectors from the iSCSI disk
4. OS boot loader (grub) takes over the control from BIOS
5. Grub loads Solaris kernel/ramdisk

6. Grub transfers control to Solaris kernel

7. Kernel scans iBFT, configures the boot NIC, TCP/IP and iSCSI initiator to enumerate the boot disk, and then mount the rootfs

8. Kernel loads the rest of drivers/conf files as booting from a local disk

5 OBP Architecture

5.1 Description

OBP will implement an iSCSI stack and define iSCSI boot related properties. After setting up these properties the OBP can enumerate the iSCSI boot disk with as a local one and then boot from it. OBP team

has submitted case FWARC 2008/466 to add iSCSI stack and iSCSI boot properties into OBP.

5.2 Interfaces

OBP will provide interface to let user/installer configure the iSCSI boot. Solaris kernel will convert these properties into a unified global iSCSI boot property.

5.2.1 User-visible

OBP interfaces to configure iSCSI boot properties, include

- iSCSI boot nic device, nic IP, nic netmask, gateway IP
- iSCSI boot target IP and port
- CHAP name and secret sent to target, optional
- CHAP name and secret expected from target, optional

5.2.2 Internal (optional for ARC review)

Unified global iSCSI boot properties for x86 and sparc, refer to the interface documentation in appendix.

5.3 Operation

A step by setp description about this procedure is,

1. System is powered up/reseted, OBP is started after POST

2. User may configure OBP to set up iSCSI boot properties, optional
3. OBPconnects to the iSCSI target with its own iSCSI/TCP/IP stack and enumerate iSCSI luns as local disks
4. User may select one disk to boot, optional
5. OBP loads the booter from the iSCSI disk and constructs iSCSI boot properties
6. Booter reads in the boot archive and then executes it, no change needed
7. Ramdisk extracts the kernel image and excutes it, no change needed
8. Kernel loades iSCSI boot properties of the boot iSCSI target from OBP
9. With these properties, kernel configures the boot NIC, TCP/IP and iSCSI initiator to enumerate the boot disk, and then mount the rootfs
10. The rest is as booting from a local disk

6 Installer Architecture

6.1 Description

Solaris automated installer will be supporting iSCSI disk.

6.2 Interfaces

Solaris automated installer will expose GUI interfaces to user for configuring iSCSI disk, it will also check/update OBP properties on sparc to simplify the configuration for next booting. The improvement relies on

this project to ensure configured iSCSI disk is able to be boot-off.

6.2.1 User-visible

TBD

6.2.2 Internal (optional for ARC review)

None

6.3 Operation

TBD

Appendix A: Standards Supported

References

R.1 Related Projects

new-boot sparc http://sac.sfbay/PSARC/2006/525

Solaris Boot Architecture http://sac.sfbay/PSARC/2004/454

scsi-self-identifying http://sac.eng.sun.com/PSARC/2008/337

iSCSI Software boot http://sac.sfbay/PSARC/2007/450

R.2 Background Information for this Project or its Product

http://kungfu.prc/index.php/ISCSI_Team#iSCSI_Boot

Guide to configure Intel's iBF, see iBF_setup.pdf in the case directory.

R.3 Interface Specifications

IBFT specification, see the documentation in case directory.

Unified global properties for iSCSI boot, see the documentation in case dicrectory.

R.4 Project Details

None.