Introduction

Many systems and security administrators have often longed for a practical method to generate new or upgraded operating system installations based on best security practices in conjunction with system preparation guidelines set by one's company. The process of loading an operating system and then hardening a system seemed to be 2 independent and time-consuming operations until Sun Microsystems initiating its' SOLARIS™ SECURITY TOOLS initiative formerly know as JASS 0.3. If one is in a business environment that requires new deployments or redeployment of system hardware on a regular basis, this can be a dilemma for systems and security administrators alike. The question at hand is, how does one ensure that each installation and securing of these systems were done in a consistent manor without the aid of the dreaded checklist. The phrase dreaded checklist is used for there could be items due to non-malicious intent that are skipped due to breaks in the work day, etc.

The answer to this question is automation of the installation and security hardening process. Many of the results from using SOLARIS™ SECURITY TOOLS are similar to YASSP, "Yet Another Solaris Security package". It should be emphasized that Sun Microsystems' SOLARIS ™ SECURITY TOOLS is still a beta release, with Version 0.3 being the best yet. It combines the ability of installing of operating systems with the hardening of the installation. Please bear in mind that this is not an official product yet. It is provided by Sun Microsystems for the exploration of what the JumpStart process is capable of accomplishing. There is much work to be done to make this a saleable product, but it's a great start for automating the installation and securing of a machine. The package from Sean Boran and crew, YASSP provides somewhat more secure machine when the process is complete. Ultimately this process would come in handy if someone at a senior level perfects the package and security configurations used in one's environment. The installation and securing process could then be handled by novice administrators with the assurance that the installation was completed up to corporate standards utilizing best practices

Server Installation

I would like to stress that use of SOLARIS™ SECURITY TOOLS constitutes a network install. This in itself is a problem if the install will not happen on an isolated segment of network that has no connectivity to any other network segments, especially those with Internet connectivity. It could be possible for the machine having its' operating system installed to be hijacked during the installation process. I have not verified this probability, but surmise it could be done. Although the probability is low, it may be possible. Additionally, when creating your install server, please create it in a safe environment, for instance, no network connections until the machine has been hardened security wise. This machine by its very nature will not be secured to the fullest due to the nature of some of the services required to run for totally unattended installation. The services that will be required are NIS or NIS+, and NFS. The NIS services provide the new installation process information about the time zone and locale while booting across the network. The NFS services are needed to bootstrap the operating system and populate the OS directories. The very existence of these services running during an install should be a red flag for any security administrator.

Securing the machine may be accomplished through the tried and true method of the manual process with requisite checklist, utilizing YASSP, or you may use SOLARIS™ SECURITY TOOLS Ver. 0.3 itself in standalone mode. There is a special script that has been constructed for the purpose of hardening the jumpstart server. It is called hardening-jumpstart-driver. Standalone mode assumes that you have already looked at the package's configurations in the Drivers subdirectory.

One caveat should be noted regarding the standalone method and the installation of OBSDssh.pkg. This method will not install the package in this mode. It would have trample one existing ssh_config settings if allowed to install the package. I do not wish to go into details on how to use SOLARIS™ SECURITY TOOLS in the standalone at this point in time for that is jumping ahead of myself. The first thing one must get is the SOLARIS™ SECURITY TOOLS Ver. 0.3 Package. It can be found http://www.sun.com/blueprints/tools under the link Solaris™ Security Tools. Examine the above-mentioned script and determine if it meets your needs.

I strongly recommend that one read the accompanying documentation to this package and the documentation from JumpStart Architecture and Security Scripts Version 0.2 package carefully. The JumpStart Architecture and Security Scripts package version 0.2 documentation contains the requisite background to complete the setup and installation tasks. The documentation offers a good explanation of how to configure Solaris™ Security Tools, but it is a bit vague when it comes to the mechanics of SOLARIS™ SECURITY TOOLS. I hope this paper will clear up some ambiguities for the next person using Solaris™ Security Tools for deployment of machines. The machine recommended for a jumpstart server does not need a lot of processor cycles. Therefore this machine can be any machine that can run a recent version of Solaris, preferably version 8. The machine in question will require a lot of disk space since one is overlaying a jumpstart installation tree with this jumpstart package. One should setup the server with plenty of space reserve for a jumpstart slice.

Caution should be observed when setting up the jumpstart server's disk. One should avoid at all cost using symbolic links for they will work fine for one version of the operating system. Since the Solaris Security toolkit is meant to be a focal point for installs of Solaris versions 2.5.1 through 8, it is not recommended to modify the scripts. Additionally, since this is beta package, the scripts change from release to release to incorporate new features. One is best off creating one's own scripts based on the stock scripts. If you don't intend on this facility to install more than one operating system version and are happy with the installed result then please disregard the warning.

The install of the server is started as such:

```
oliver# ./setup_install_server /jumpstart/OS/Solaris_7.0_05-99
Verifying target directory...
Calculating the required disk space for the Solaris_2.7 product
Copying the CD image to disk...
Install Server setup complete
```

Once the operating system(s) have been laid down in the nomenclature of your choosing, one should overlay this with the actual JASS package. Do this by first installing the raw package like so:

```
oliver# pkgadd -d SUNWjass-0.3.pkg

The following packages are available:
  1  SUNWjass     JASS Toolkit 0.3
                (Solaris) 0.3
Select package(s) you wish to process (or 'all' to process
all packages). (default: all) [?,??,q]: 1
Processing package instance <SUNWjass> from </export/home/kmartin/SUNWjass-
0.3.p
kg>
JASS Toolkit 0.3
(Solaris) 0.3
The selected base directory </opt/SUNWjass> must exist before
installation is attempted.

Do you want this directory created now [y,n,?,q] y
```

The product itself will install itself in /opt/jass-0.3. Copy these files and directories recursively to your jumpstart target directory. The following is a bird's eye view of the directories. Note I am including the directories that the documentation does not tell you must exist for complete package installation of various packages that are paramount to securing new installations, such as FixModes and OBSDssh.

```
 14 -r-xr-xr-x   1 root     root        6964 Jun 10 18:43 add-client
 26 -r--r--r--   1 root     root       12413 Jun 13 20:41 CHANGES
 60 -rwxr-xr-x   1 root     root       30516 Aug  2 16:24 check
  8 -r--r--r--   1 root     root        3200 Jun 15 16:12 CREDITS
  2 drwxrwsr-x   2 root     root         512 Aug  2 15:33 Documentation
  2 drwxrwsr-x   2 root     root        1024 Aug  6 14:04 Drivers
  2 drwxrwsr-x   4 root     root         512 Aug  2 15:33 Files
  2 drwxrwsr-x   4 root     root         512 Aug  2 15:33 Files.orig
  6 drwxrwsr-x   2 root     root        2560 Aug  5 17:51 Finish
  8 -r--r--r--   1 root     root        3492 Jun  8 08:39 INSTALL
 24 -r-xr-xr-x   1 root     root       12277 Jun 15 16:12 jass-execute
  6 -r--r--r--   1 root     root        2716 Jun  8 08:42 LICENSE
 16 drwx------   2 root     root        8192 Aug  2 12:03 lost+found
 10 -r-xr-xr-x   1 root     root        4160 Jun 10 18:43 make-pkg
 20 -r--r--r--   1 root     root       10225 Jun 15 16:14 MANIFEST
  4 -r--r--r--   1 root     root        1921 Jun 10 18:44 nomatch.beg
  2 drwxr-xr-x   5 root     root         512 Aug  2 16:06 OS
  2 drwxr-xr-x   2 root     root         512 Aug  3 15:08 Packages
  2 drwxr-xr-x   3 root     root         512 Aug  2 16:15 Patches
  2 drwxrwsr-x   2 root     root         512 Aug  2 16:17 Profiles
  4 -r--r--r--   1 root     root        1823 Jun  8 08:58 README
  6 -r-xr-xr-x   1 root     root        2724 Jun 10 18:43 rm-client
  6 -r--r--r--   1 root     root        2249 Aug  3 09:39 rules
  2 -rw-r--r--   1 root     root         461 Aug  3 15:09 rules.ok
  4 -r--r--r--   1 root     root        1760 Jun 10 18:44 rules.SAMPLE
  2 drwxrwsr-x   6 root     root         512 Aug  2 15:33 Sysidcfg
  2 drwxr-xr-x   3 root     root         512 Aug  6 09:37 tmp
```

add-client and rm-client are scripts that their names are very descriptive to their function in the JumpStart process. These scripts fill in the blanks as compare with the regular jumpstart scripts found under the /jumpstart/OS/osname_version/Solaris_X/Tools directory. Using the add-client script saves much typing at the risk of fat fingering the long command string. Below you will find an example of a Solaris 7 client being added to the JASS directory:

```
oliver# ./add-client spyder Solaris_7.0_05-99 sun4c oliver
saving original /etc/dfs/dfstab in /etc/dfs/dfstab.orig
Adding "share -F nfs -o ro,anon=0 /jumpstart/OS/Solaris_7.0_05-99" to /etc/dfs/d
fstab
making /tftpboot
enabling tftp in /etc/inetd.conf
starting rarpd
starting bootparamd
starting nfsd's
starting nfs mountd
updating /etc/bootparams
copying inetboot to /tftpboot
```

The check script verifies that one has a valid set of rules in place for the install. It parses the rules and places them in a format that has been checksums for validity in the file rules.ok. It is rules.ok that is used for the installation. An abbreviated set of rules looks like the following:

```
hostname apple    sparc  -  Profiles/end-user.profile    Drivers/config-apple.driver
hostname orange    sparc  - Profiles/end-user.profile    Drivers/config-orange.driver
hostname banana    sparc  - Profiles/end-user.profile    Drivers/config-banana.driver
hostname grape  sparc   -   Profiles/end-user.profile    Drivers/config-grape.driver
hostname apricot  sparc  -  Profiles/end-user.profile    Drivers/config-apricot.driver
#any          sparc   -   Profiles/end-user.profile    Drivers/config.driver
#nomatch.beg    -                                  -
```

Run through the check script and here is output of successful execution:

```
oliver# ./check
Validating rules...
Validating profile Profiles/end-user.profile...
The custom JumpStart configuration is ok.
```

The profile dictates how to layout the disk structures and which OS clusters to install. In the example above I have chosen a user install. In practice, it is best to install the least amount of software e one can and still have a useable machine. The name of this game is risk management, and by going with the minimalist approach, one mitigates the number of potential threats via different packages. This file also points to a Drivers directory. The drivers directory is a place that you can custom tailor the security hardening of individual machines depending on their use. Additionally, it is in this area that one can direct the installation of packages such as FixModes and OBSDssh. This directory holds all of the directives and configuration data. These directives are accomplished using "Finish scripts". One can observe from this file there are 5 "fruit systems" each with their own profiles and drivers. Taking a look in the drivers directory one will find hardening.driver and secure.driver..

Now this is where the beauty of the JASS package begins. During the prep work for JASS, one should drop the latest tested and verified recommended patch clusters in the Patches directory. The patches as well as any files that you drop into the Files directory will be copied over after modification of the copy files script. One may copy the example config scripts and tailor them to one's needs in their environment. A very nice feature to the JASS package was the addition of an undo driver. This is quite handy if you find the resultant of the JASS installation excludes needed services. These scripts are located in the Finish directory .The toolkit takes a modular approach to hardening the operating system. One can call other drivers from within a driver file in order to keep the individual machine configuration readable.

There are several enhancements to the package that make the environment ultimately very useful. The creation of the Open BSD SecureShell Version 2.9patchlevel 2 is one of those helpful items. The is actually another Sun Microsystems Blueprint online that instructs one
Building and Deploying OpenSSH on the Solaris[tm] Operating Environment (July 2001)
-by Jason Reid and Keith Watson details how to obtain, compile, and package this collection of binaries custom tailored to one's environment. One item of particular interest is the Psuedo Random Number Generator kit. It so happens that Solaris does not include a /dev/random. This is because the at best it provided few too many bits for a random seed. This led to easily crackable ciphers. If you do not provide a PRNGD for OBSDssh, it creates its' own randomness. It takes time for the program to create this random function, therefore there can be a significant time period before ones ssh session initiates. The startup delay may be negligible for some and others it may be an annoyance. The document and requisite package also include the scripts to start and stop the PRNGd and the sshd daemons.

What is not mentioned in the documentation provided is that the zlib-1.1.3 and openssl-0.9.6b need to be included in the installation on the client not just the server as this documentation would . These can be placed in the /jumpstart/tmp/jass-package directory. Look at the install-openssh.fin finish script for ideas on how to create a finish script that installs the zlib and openssl packages.

There are two other packages that need to be collected compiled and installed on the machine one is using as the packaging machine. The first package is zlib-1.1.3 available at http://www.freesoftware.com/pub/infozip/zlib/ .The compression routines for openssh are in this package. The other required package is OpenSSL-0.9.6b available at http://www.openssl.org/source/ or ftp://ftp.sunfreeware.com/pub/freeware/sparc/8/openssl-0.9.6b-sol8-sparc-local.gz as a precompiled package. The cryptographic routines for ssh originate in this kit.

Client Install

The following is the command line and response that is required to install the client:

```
ok boot net - install
SPARCstation 2, No Keyboard
ROM Rev. 2.9, 64 MB memory installed, Serial #4351779.
Ethernet address 8:0:20:11:21:91, Host ID: 55426723.
Rebooting with command: net - install
Boot device: /sbus/le@0,c00000   File and args: - install
1ae00
hostname: spyder
domainname: yourdomainname.com
root server: oliver
root directory: /jumpstart/OS/Solaris_7.0_05-99/Solaris_2.7/Tools/Boot
SunOS Release 5.7 Version Generic [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1998, Sun Microsystems, Inc.
Configuring devices...
Using sysid configuration file 192.168.1.30:/jumpstart/Sysidcfg/Solaris_7/sysidcfg
The system is coming up.  Please wait.
Starting remote procedure call (RPC) services: sysidnis done.
Starting Solaris installation program...
Searching for JumpStart directory...
Using rules.ok from 192.168.1.30:/jumpstart.
Checking rules.ok file...
Using profile: Profiles/end-user.profile
Using finish script: Drivers/secure.driver
Executing JumpStart preinstall phase...
Searching for SolStart directory...
Checking rules.ok file...
Using begin script: install_begin
Using finish script: patch_finish
Executing SolStart preinstall phase...
Executing begin script "install_begin"...
Begin script install_begin execution completed.
Processing default locales
        - Specifying default locale (en_US)
etc………
```

What you wind up with on reboot of the client is:

```
Finish script Drivers/secure.driver execution completed.
The begin script log 'begin.log'
is located in /var/sadm/system/logs after reboot.
The finish script log 'finish.log'
is located in /var/sadm/system/logs after reboot.
syncing file systems... done
rebooting...ý
SPARCstation 2, No Keyboard
```

```
ROM Rev. 2.9, 64 MB memory installed, Serial #4351779.
Ethernet address 8:0:20:11:21:91, Host ID: 55426723.



Rebooting with command:
Boot device: /sbus/esp@0,800000/sd@3,0   File and args:
SunOS Release 5.7 Version Generic_106541-16 [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1999, Sun Microsystems, Inc.
configuring network interfaces: le0.
Hostname: spyder
Configuring /dev and /devices
Configuring the /dev directory (compatibility devices)
The system is coming up.  Please wait.
WARNING: /tmp had incorrect ownership (sys).
WARNING: /var/tmp had incorrect ownership (sys).
starting routing daemon.
Setting /dev/arp arp_cleanup_interval to 60000
Setting /dev/ip ip_forward_directed_broadcasts to 0
Setting /dev/ip ip_forward_src_routed to 0
Setting /dev/ip ip_ignore_redirect to 1
Setting /dev/ip ip_respond_to_address_mask_broadcast to 0
Setting /dev/ip ip_respond_to_echo_broadcast to 0
Setting /dev/ip ip_respond_to_timestamp to 0
Setting /dev/ip ip_respond_to_timestamp_broadcast to 0
Setting /dev/ip ip_send_redirects to 0
Setting /dev/ip ip_strict_dst_multihoming to 1
Setting /dev/tcp tcp_conn_req_max_q0 to 4096
Setting /dev/tcp tcp_conn_req_max_q to 1024
Setting /dev/tcp tcp_smallest_anon_port to 32768
Setting /dev/tcp tcp_largest_anon_port to 65535
Setting /dev/udp udp_smallest_anon_port to 32768
Setting /dev/udp udp_largest_anon_port to 65535
Setting /dev/tcp tcp_smallest_nonpriv_port to 1024
Setting /dev/udp udp_smallest_nonpriv_port to 1024
Setting /dev/ip ip_ire_flush_interval to 60000
Setting /dev/tcp tcp_extra_priv_ports_add to 6112
Setting netmask of le0 to 255.255.255.0
syslog service starting.
starting audit daemon
Configured 229 kernel events.
Starting process accounting
The system is ready.
spyder console login: root
Password:
```

```
Aug 12 13:30:01 spyder login: ROOT LOGIN /dev/console
#########################################################################
# This system is for the use of authorized users only.                  #
# Individuals using this computer system without authority, or in       #
# excess of their authority, are subject to having all of their         #
# activities on this system monitored and recorded by system            #
# personnel.                                                    #
#                                                                       #
# In the course of monitoring individuals improperly using this         #
# system, or in the course of system maintenance, the activities        #
# of authorized users may also be monitored.                            #
#                                                                       #
# Anyone using this system expressly consents to such monitoring        #
# and is advised that if such monitoring reveals possible               #
# evidence of criminal activity, system personnel may provide the       #
# evidence of such monitoring to law enforcement officials.             #
#########################################################################
 # ps -ef
     UID   PID  PPID  C    STIME TTY      TIME CMD
    root     0     0  0 23:22:53 ?        0:03 sched
    root     1     0  0 23:22:57 ?        0:00 /etc/init -
    root     2     0  0 23:22:57 ?        0:00 pageout
    root     3     0  1 23:22:57 ?        0:06 fsflush
    root   241     1  0 23:24:05 ?        0:00 /usr/sbin/auditd
    root   125     1  0 23:23:48 ?        0:00 /usr/sbin/in.routed -q
    root   283     1  0 23:24:14 ?        0:00 /usr/lib/saf/sac -t 300
    root    44     1  0 23:23:16 ?        0:00 /usr/lib/devfsadm/devfseventd
    root    46     1  0 23:23:18 ?        0:00 /usr/lib/devfsadm/devfsadmd
    root   229     1  0 23:24:02 ?        0:00 /usr/lib/utmpd
    root   200     1  0 23:23:56 ?        0:00 /usr/sbin/inetd -s -t
    root   244     1  0 23:24:06 ?        0:02 /usr/local/sbin/prngd --cmdfile /usr/local/etc/prngd.conf --
seedfile /var/spool
    root   245     1  0 23:25:34 ?        0:00 /usr/local/sbin/sshd
    root   207     1  0 23:23:59 ?        0:01 /usr/sbin/syslogd
    root   212     1  0 23:24:00 ?        0:00 /usr/sbin/cron
    root   290   283  0 23:24:17 ?        0:01 /usr/lib/saf/ttymon
    root   285     1  1 23:24:15 console  0:01 /usr/bin/login
    root   233     1  0 23:24:03 ?        0:00 /usr/sbin/nscd
    root   355   285  1 17:05:41 console  0:00 -sh
    root   361   355  2 17:05:52 console  0:00 ps -ef
```

While this installation leaves more daemons running the YASSP, it is a far cry better than what is provided from a basic installation.

I have preempted most of the installation log for the sake of saving space in the document.

In closing, The SOLARIS™ SECURITY TOOLKIT initiative holds promise in providing secure installations and upgrades for those of us who require repeatable, quick installations and upgrades of the Solaris Operating Environment. It is important to my company to maintain a connection to the past for instance Solaris 2.6. There are legacy applications that are in the process of being ported to Solaris 8, but in the meanwhile it is good to know one can still generate a more secure version of an old operating system as well as provide facilities to move forward in a secure fashion. The facilities in the Solaris Security Toolkit enables us to move forward on some installations, yet maintain a connection with the past for installations.

I urge any readers of this document to read the documentation of the Solaris Security Toolkit carefully before attempting deployment. This document is a synopsis of what can be achieved using this tool. Version 0.3 of the toolkit also includes a facility to create your own packages. This holds promise for the future. I only wish I had time to implement it, but as of this writing, it has not been in existence very long.

Kasper, Anthony Paul. Automating Solaris Installations: A Custom Jumpstart Guide with Disk
Prentice Hall PTR 1995

Noordergraaf,Alex ,Building a JumpStart[tm] Infrastructure (April 2001)
http://www.sun.com/software/solutions/blueprints/0401/BuildInf.pdf

Noordergraaf,Alex Brunette, Glen , JumpStart[tm] Architecture and Security Scripts for the Solaris[tm] Operating Environment - Part 1 (November 2000)
http://www.sun.com/software/solutions/blueprints/1100/ssec-updt1.pdf

Noordergraaf,Alex Brunette, Glen , JumpStart[tm] Architecture and Security Scripts for the Solaris[tm] Operating Environment - Part 2 (November 2000)
http://www.sun.com/software/solutions/blueprints/1100/ssec2-updt1.pdf

Noordergraaf,Alex Brunette, Glen , JumpStart[tm] Architecture and Security Scripts for the Solaris[tm] Operating Environment - Part 3 (November 2000)
http://www.sun.com/software/solutions/blueprints/1100/ssec3-updt1.pdf

Noordergraaf,Alex, JumpStart[tm] Architecture and Security Scripts for the Solaris[tm] Operating Environment - Part 3 (September 2000)
http://www.sun.com/software/solutions/blueprints/0900/jssec3.pdf

Noordergraaf,Alex, JumpStart[tm] Architecture and Security Scripts for the Solaris[tm] Operating Environment - Part 2 (August 2000)
http://www.sun.com/software/solutions/blueprints/0800/jssec2.pdf

Noordergraaf,Alex, JumpStart[tm] Architecture and Security Scripts for the Solaris[tm] Operating Environment - Part 1 (July 2000)
http://www.sun.com/software/solutions/blueprints/0900/jssec.pdf

Reid, Jason and Watson, Keith Building and Deploying OpenSSH on the Solaris[tm] Operating Environment (July 2001)
http://www.sun.com/software/solutions/blueprints/0701/openSSH.pdf

Jaenicke, Lutz PRNGD - Pseudo Random Number Generator Daemon
http://www.aet.tu-cottbus.de/personen/jaenicke/postfix_tls/prngd.html

Gutman, Peter Random Number Generation
http://www.cryptoengines.com/~peter/06_random.pdf