

UNDERSTANDING SD-UX ACLS

by Al Miller

Software Distributor (SD) allows system administrators to package, distribute, install, verify, and remove software on their systems. Most commonly, a local superuser performs these operations. But SD gives you much more flexibility. Using SD Access Control Lists (ACLs), the system administrator can authorize nonsuperusers—including users on remote systems—to perform SD tasks. As data centers get more complex, these features can save you time and effort.

Prior to HP-UX 11i, you needed HP's OpenView product to use some Software Distributor features, especially the ability to "push" software to remote systems and schedule jobs. With 11i, you can enable these features for free—and using ACLs helps you fully administer these capabilities.

You can also use the ServiceControl Manager product to perform SD tasks. Installing SCM on a system also installs special ACLs that free you from having to manage ACLs manually when you use applications through SCM.

The examples shown in this article were developed primarily using HP-UX 11.00 (swcrunch), and HP-UX 11i

(swelter) systems, and the quoted documentation is from HP-UX 11i.

The *swacl* Command

The *swacl* command is your primary tool for displaying and managing Software Distributor ACLs. At minimum, this command requires only an argument that indicates the level of ACL to be operated upon. The usage statement of the *swacl* command is:

```
Usage: swacl -l level [options] [software_selections] [@target_selections]
```

```
-l level          level of ACLs to view/modify, one of "host",
                  "depot", "root", "product", "product_template",
                  "global_soc_template", "global_product_template"
```

Options include:

```
-M acl_entry      add acl_entry to ACL or replace existing entry
-D acl_entry      delete acl_entry from ACL
-F acl_file       replace ACL with entries in this file
-X option=value   set the option to value
-X option_file    read option definitions from this file
-f software_file  read product selections from this file
-t target_file    read target selections from this file
```

An *acl_entry* is specified as:

```
entry_type[:key]:permissions
```

Software selections are specified as:

```
product[,version] ...
```

where version is either

```
[r=revision][,a=arch][,v=vendor]
instance_id
```

Target selections are specified as:

```
@ [host][:][path] ...
```

This discussion focuses on the *level* and *acl_entry* arguments, demonstrating how you can use them to control access between local and remote systems. Now let's look at an example.

Example 1: Display the default *root* ACLs on a newly installed HP-UX 11i system:

```
swelter : root $ swacl -l root

#
# swacl    Installed Software Access Control List
#
# For host: swelter:/
#
# Date:    Wed Feb 28 14:58:02 2001
#
# Object Ownership:  User= root
#                   Group=sys
#                   Realm=swelter.fc.hp.com
#
# default_realm=swelter.fc.hp.com
object_owner:crwit
any_other:--r---
```

This ACL indicates that the `root` user owns the file system. As `root`, the owner has full ACL permissions (*crwit*). (We'll talk more about these permissions later.) Additionally, all other users can read SD information about this root file system using the *swlist* command. (Note that the default ACLs have changed slightly on HP-UX 11i versus earlier HP-UX releases: the group ACL for *swadm* has been deleted and the *t* permission has been removed from *any_other*.)

ACL LEVELS: HOSTS, ROOTS, DEPOTS, AND PRODUCTS AND TEMPLATES

SD host objects correlate with an individual system and may contain roots and depots. The universal example of a root is `/`. SD writes all software that it installs to a root file system and records these actions in the Installed Product Database. Software installed on a root is organized into products, file sets, subproducts, and bundles, and you can display the properties of these objects using the *swlist* command.

A quick but important aside: other roots known as "alternate roots" may exist elsewhere. These alternate roots enable diskless clusters, which are not commonly used after HP-UX 10.20. Don't make the mistake of trying to use alternate roots to install software to locations other than the default location. (To do that, the software packager must have set the *is_locatable* attribute when the software was packaged and the software itself must function properly from an alternate location.)

In addition to roots, hosts may also contain SD depots, which are repositories from which SD installs software. As with roots, ACLs also protect depots. Unlike roots, products within a depot also contain ACLs, which lets you allow or deny access to individual products in the depot.

ACLs protecting hosts, roots, depots, and products within depots affect those objects that already exist on the system. However, additional "template" ACLs affect objects when they are

created. These templates are *global_soc_template*, *global_product_template*, and *product_template*.

global_soc_template provides default ACLs that apply to all new depots and roots added to the host. *global_product_template* initializes the *product_template* of future depots added to the host, and *product_template* affects the ACLs of future products added to a depot.

The SD ACL Permissions

Let's take another look at the five permissions used in ACLs:

- r* (read) Grants permission to read an object:
 - for hosts, depots, or roots: permits *swlist* operations
 - for products within depots: permits installation or copying of product files by *swinstall* or *swcopy*
- w* (write) Grants permission to modify an object:
 - for roots (e.g., installed root file system): permits modification of the products installed or contained within the root
 - for depots: really does very little. It does *not* permit modification of the products inside the depot.
 - for products: permits modification of a specific product within a depot
 - for hosts: permits unregistration of depots. It does *not* grant permission to modify depots or roots on the host.
- i* (insert) Grants permission to create new objects:
 - for depots: permits creation of new products in the depot
 - for hosts:

- permits creation of new software depot or root file system objects
 - permits registration of roots and depots
- c* (control) Grants permission to modify the ACL using *swacl*
- t* (test) Grants permission to perform access checks and to list the ACL
- a* (all) A wildcard that grants all of the above permissions (*swacl* expands this to *crowit*.)

The meaning of these permissions can change subtly depending on which type of object the ACL protects. In particular, pay close attention to the difference between the *write* and *insert* permissions when they apply to host objects. *Write* permission on a host only lets you unregister an existing depot. To create a new depot or register an existing depot, you also need *insert* permission.

ACL Entries

The *acl_entry* field, used with the *swacl*'s *-M* and *-D* options, lets you specify more complex classes of permissions. These are the available values for *entry_type*:

- any_other*
- group*
- host*
- object_owner*
- object_group*
- other*
- user*

Pay particular attention to the distinction between the *user* and *host* entry types. A *user* entry type controls actions performed by a particular user running SD commands such as *swinstall*, *swremove*, and *swcopy*. By contrast, the *host* entry type affects the permissions for SD agents (*swagent*) processes. ACLs with an *entry_type* of

host are used primarily to control access to software depots.

Example 2. In this example, root gives user *allen* insert permission on the host. This lets *allen* create new depots (and alternate roots), but does not let him install software to the default root, "/". Examples 2 and 3 show that you can separate the task of packaging software from that of installing the same software. (Note that Example 2 shows all command output from the SD commands. Subsequent examples will show only the command lines and the most relevant output.)

```
swelter : root $ swacl -l host -M user:allen:i
swelter : root $ swacl -l host
#
# swacl Host Access Control List
#
# For host: swelter
#
# Date: Wed Feb 28 16:56:16 2001
#
# Object Ownership: User= root
#                   Group=sys
#                   Realm=swelter.fc.hp.com
#
# default_realm=swelter.fc.hp.com
user:allen:---i-
any_other:--r---
```

Next, *allen* has prepared a simple SD Product Specification File (PSF), which he displays and then packages into a depot using the *swpackage* command:

```
swelter : allen $ cat simple_1.psf
product
tag test_product
title "very simple test product"

fileset
tag test_a
title "fileset a"

directory . = /product_top
file simple_1.psf

end

end
swelter : allen $ swpackage -s simple_1.psf @
                               /simple_1.depot

===== 02/28/01 16:53:26 MST BEGIN swpackage SESSION

* Session started for user "allen@swelter.fc.hp.com".

* Source:          swelter:simple_1.psf
* Target:          swelter:/simple_1.depot
* Software selections:

*

* Beginning Selection Phase.
* Reading the Product Specification File
  (PSF)"simple_1.psf".
* Reading the product "test_product" at line 1.
* Reading the fileset "test_a" at line 6.
```

```
NOTE:   Creating new target depot "/simple_1.depot".
* Selection Phase succeeded.

* Beginning Analysis Phase.
NOTE:   The estimated free disk space required on
filesystem "/" is 1 Kbyte blocks. This
requirement will leave 103798 Kbyte blocks
of free disk space on the filesystem after
the packaging session completes.
* Analysis Phase succeeded.

* Beginning Package Phase.
* Packaging the product "test_product".
* Packaging the fileset "test_product.test_a".
* Package Phase succeeded.

NOTE:   You must register the new depot "/simple_1.depot"
to make it generally available as a source
for swinstall and swcopy tasks. To register it,
execute the command

        swreg -l depot /simple_1.depot

===== 02/28/01 16:53:28 MST  END swpackage SESSION
```

Now *allen* registers the depot, uses *swlist* to list the depot contents, and then uses *swacl* to list the ACL for the newly created depot:

```
swelster : allen $ swreg -l depot /simple_1.depot
===== 02/28/01 16:54:08 MST  BEGIN swreg SESSION (non-
interactive)
* Session started for user "allen@swelster".
* Beginning Selection
* Targets:                swelster
* Objects:                /simple_1.depot
* Selection succeeded.

===== 02/28/01 16:54:08 MST  END swreg SESSION
(non-interactive)

swelster : allen $ swlist -d @ /simple_1.depot
# Initializing...
# Contacting target "swelster"...
#
# Target:  swelster:/simple_1.depot
#
#
# No Bundle(s) on swelster:/simple_1.depot
# Product(s):
#
test_product                very simple test product

swelster : allen $ swacl -l depot @ /simple_1.depot
#
# swacl  Depot Access Control List
#
# For depot:  swelster:/simple_1.depot
#
# Date:  Wed Feb 28 16:54:42 2001
#
# Object Ownership:  User= allen
#                   Group=users
#                   Realm=swelster.fc.hp.com
#
# default_realm=swelster.fc.hp.com
object_owner:crwit
any_other:-r---
```

Now *allen* tries—and fails—to install his product from the depot to the root file system:

```
swelster : allen $ swinstall -s /simple_1.depot test_product
===== 02/28/01 16:58:50 MST  BEGIN swinstall SESSION
(non-interactive) (jobid=swelster-0010)
* Session started for user "allen@swelster".
* Beginning Selection
ERROR:  "swelster:/": You do not have the required
permissions to select this target.
Check permissions using the "swacl"
command or see your system administrator for
assistance. Or, to manage applications
designed and packaged for nonprivileged
mode, see the "run_as_superuser" option in
the "sd" man page.
* Target connection failed for "swelster:/".
ERROR:  More information may be found in the daemon
logfile on this target (default location
is swelster:/var/adm/sw/swagentd.log).
* Selection had errors.

===== 02/28/01 16:58:51 MST  END swinstall SESSION
(non-interactive)
(jobid=swelster-0010)
```

The above command points out that while *allen* has ACL permissions to create and register depots, those permissions do not also let him install depots to the default root. However, *allen* tries again—and finds that his permissions do let him install his depot to an alternate root:

```
swelster : allen $ swinstall -s /simple_1.depot test_product
@ /altroot/
===== 02/28/01 17:00:06 MST  BEGIN swinstall SESSION
(non-interactive) (jobid=swelster-0012)
* Session started for user "allen@swelster".
* Beginning Selection
* "swelster:/altroot/": This target does not
exist and will be created.
* Source connection succeeded for
"swelster:/simple_1.depot".
* Source:                /simple_1.depot
* Targets:                swelster:/altroot/
* Software selections:
test_product.test_a
* Selection succeeded.

* Beginning Analysis and Execution
* Session selections have been saved in the file
"/home/allen/.sw/sessions/swinstall.last".
* The analysis phase succeeded for "swel-
ter:/altroot/".
* Analysis and Execution succeeded.

NOTE:   More information may be found in the agent
logfile using the command "swjob -a log
swelster-0012 @ swelster:/altroot/".

===== 02/28/01 17:00:09 MST  END swinstall SESSION
(non-interactive)
(jobid=swelster-0012)
```

To see what he has installed, *allen* runs two *swlist* commands. His product isn't on the root file system, but it has installed just fine on the alternate root:

```
swelster : allen $ swlist
```

```
# Initializing...
# Contacting target "swelter"...
#
# Target:  swelter:/
#
#
# Bundle(s):
#
CDE-English      B.11.11      English CDE Environment
FDDI-00          B.11.11.01  PCI FDDI;Supptd
GigEther-00     B.11.11.14  PCI/HSC GigEther;Supptd
HPUX11i-OE      B.11.11      HP-UX Internet Operating
                Environment
HPUXBase32      B.11.11      HP-UX 32-bit Base OS
HPUXBaseAux     B.11.11      HP-UX Base OS Auxiliary
OnlineDiag      B.11.11.00.04 HP-UX 11.11 Support Tools
                Bundle
```

```
swelter : allen $ swlist @ /alroot
# Initializing...
# Contacting target "swelter"...
#
# Target:  swelter:/alroot
#
#
# No Bundle(s) on swelter:/alroot
# Product(s):
#
test_product      very simple test product
```

Example 3. Using the software package from the previous example, the administrator enables user *dennis* to install software into the default root.

```
swelter : root $ swacl -l root -M user:dennis:ri
```

The *r* (read) permission lets *dennis* open the root for reading, and *i* (*insert*) permission lets him insert the new product into the root object.

```
swelter : dennis $ swinstall -s /simple_1.depot
test_product
```

dennis may also remove this or any other product installed on the root file system:

```
swelter : dennis $ swremove test_product
swelter : dennis $ swremove Xserver
```

Note, however, that *dennis* cannot create registered depots and therefore cannot install the software that he has packaged himself. The SD administrator must change the ACL for *dennis*, or another user with the proper authorization must register the depot.

MANAGING REMOTE SYSTEMS

Much of the utility of SD ACLs comes from the ability to control the access of remote users

to a system's software. ACLs let you flexibly control the ability of remote users to install or remove software, list installed software, or access depots.

Example 4. Disallow remote systems from using *swlist* on the local system.

By default, the *any_other:r- - root* ACL lets all users on your network use *swlist* on the contents of your root:

```
swelter : allen $ swlist @ swcrunch
# Initializing...
# Contacting target "swcrunch"...
#
# Target:  swcrunch:/
#
#
# Bundle(s):
#
HPUXEng32RT      B.11.00      English HP-UX 32-bit
                Runtime Environment
XSWGR1100        B.11.00.45  HP-UX Extension Pack,
                May 1999
```

To disallow this default behavior, you must remove the *any_other* ACL on *swcrunch*:

```
swcrunch : root $ swacl -l root -D any_other

swelter : allen $ swlist @ swcrunch
# Initializing...
# Contacting target "swcrunch"...
ERROR: "swcrunch:/" : You do not have the required
permissions to select this target. Check
permissions using the "swacl" command or
see your system administrator for assistance.
Or, to manage applications designed and packaged
for nonprivileged mode, see the "run_as_superuser"
option in the "sd" man page.
ERROR: More information may be found in the daemon
logfile on this target (default location is
swcrunch:/var/adm/sw/swagentd.log).
```

Example 5. Allow a remote user (*allen@swelter*) to fully manage the root file system on *swcrunch*:

```
swcrunch : root $ swacl -l root -M user:allen@swelter:a
swelter : allen $ swinstall -s /simple_1.depot/
* @ swcrunch
swelter : allen $ swremove Xserver @ swcrunch
```

Example 6: Disallow all remote users from accessing */simple_1.depot* on *swelter*, but allow local users to access the depot:

```
swelter : root $ swacl -l depot -D any_other @
/swelter/simple_1.depot
swelter : root $ swacl -l depot -M other:r @
/swelter/simple_1.depot
swelter : root $ swacl -l depot @ /simple_1.depot
#
# swacl Depot Access Control List
```

```
#
# For depot: swelter:/simple_1.depot
#
# Date: Thu Mar 1 16:19:57 2001
#
# Object Ownership: User= allen
#                   Group=users
#                   Realm=swelter.fc.hp.com
#
# default_realm=swelter.fc.hp.com
object_owner:crwit
other:-r---
```

Local users can now access this depot because of the *other* ACL, but remote users are refused. Also notice that user *root* can perform this action on the */simple_1.depot* object even though it does not own the object or have any ACLs. This is because SD never checks ACLs for local superusers.

Example 7. Allow only user *shelly* on host *swcrunch* to access software in a depot located on *swelter*:

At first glance, you may think that all you need to do is add a user ACL for *shelly*:

```
swelter : root $ swacl -l depot -M user:shelly@swcrunch:r @
                                     /simple_1.depot
```

However, this alone is not enough. If *shelly* attempts to access the depot, she fails with a security violation. Why? You must enable the host as well as the user on the host. In other words, you must authorize the user (*shelly* on system *swcrunch*) and the SD agents (the *swagent* process) that contact depot servers. To do this, use a *host ACL entry_type*:

```
swelter : root $ swacl -l depot -M host:swcrunch:r @
                                     /simple_1.depot
```

Now, *shelly*'s *swinstall* command succeeds (presuming she has appropriate ACL permission to install software on *swcrunch*):

```
swcrunch : shelly $ swinstall -s swelter:/simple_1.depot
                                     test_product
```

Because *shelly* is the only authorized user, another user, *teresa*, also on *swcrunch*, cannot access the same depot with the same command:

```
swcrunch : teresa $ swinstall -s swelter:/simple_1.depot
                                     test_product
```

```
===== 03/01/01 16:57:59 MST BEGIN swinstall SESSION
(non-interactive)
* Session started for user "teresa@swcrunch".
* Beginning Selection
* Target connection succeeded for "swcrunch:".
ERROR: "swelter:/simple_1.depot": You do not have the
required permissions to perform this SD operation.
Please check to see that you have the required
permissions using the "swacl" command or see your
system administrator for assistance.
* Source connection failed for
"swelter:/simple_1.depot".
WARNING: More information may be found in the daemon
logfile on this target (default location is
swelter:/var/adm/sw/swagentd.log).
* Selection had errors.
===== 03/01/01 16:58:09 MST END swinstall SESSION
(non-interactive)
```

Because granting depot access requires that you set up *user* and *host* ACLs, you may find it useful to use the wildcard (*) option with the *host ACL entry_type* (this feature was added to SD with HP-UX 11i):

```
swelter : root $ swacl -l depot -M host:*:r @
                                     /simple_1.depot
```

The wildcard eliminates the need for multiple *swacl-M host* commands—although you still must add multiple user ACLs.

Example 8. Grant remote *root* users the same privileges as any other remote user.

Example 5 above removed the *any_other* ACL from */simple_1.depot* on *swelter*, but the local *root* user still had unrestricted access to the depot. However, remote *root* users are not granted the same privilege. Given the permissions set up so far, *root@swcrunch* is denied access to the sample depot. To permit access, you must grant remote *root* user ACL permissions as you would for any other remote user:

```
swelter : allen $ swacl -l depot -M user:root@swcrunch:r @
                                     /simple_1.depot
```

SERVICECONTROL MANAGER AND SD ACLS

ServiceControl Manager (SCM) comes with HP-UX 11i—although some configuration is necessary. This product allows single-point, multisystem configuration management and provides a number of sophisticated management tools. It lets you use HP-UX tools already designed to deal with multiple systems (such as

Ignite-UX and SD). It also removes the need for you to deal with many of the individual ACL tasks needed for SD administration. Even so, knowing how SCM manages SD ACLs can help you better understand and administer this product.

A few examples will help. Here are some facts and assumptions:

- SCM environments consist of a Central Management Server (CMS) and managed nodes.
- The system *swelter* is the CMS running HP-UX 11i.
- The system *swcrunch* is the managed node running HP-UX 11.00. It must have SCM and a prerequisite patch bundle installed. (Both are available from the SCM Web site.)
- *swcrunch* is configured to be a managed node by SCM.
- SCM managed nodes grant authorization to the CMS by installing a special file set, *AgentConfig.SD-CONFIG*, from a depot on the CMS system. That depot is *swelter:/var/opt/mx/depot11*.

Take a look at the *host* and *root* ACLs on *swcrunch* before installation of this product:

```
swcrunch : root $ swacl -l root

#
# swacl    Installed Software Access Control List
#
# For host: swcrunch:/
#
# Date:   Fri Mar  2 12:40:51 2001
#
# Object Ownership:  User= root
#                   Group=sys
#                   Realm=swcrunch.fc.hp.com
#
# default_realm=swcrunch.fc.hp.com
object_owner:crwit
user:shelly:crwit
user:teresa:crwit
user:allen@swelter.fc.hp.com:crwit
group:swadm:crwit
```

Example 9. Install the *AgentConfig.SD-CONFIG* file set to set up the SCM authorizations:

```
swcrunch : root $ swinstall -s swelter:/var/opt/mx/depot11
AgentConfig.SD-CONFIG
```

Now you can observe that an ACL for *root@swelter* has been added:

```
swcrunch : root $ swacl -l root
#
# swacl    Installed Software Access Control List
#
# For host: swcrunch:/
#
# Date:   Fri Mar  2 12:57:18 2001
#
# Object Ownership:  User= root
#                   Group=sys
#                   Realm=swcrunch.fc.hp.com
#
# default_realm=swcrunch.fc.hp.com
object_owner:crwit
user:shelly:crwit
user:teresa:crwit
user:allen@swelter.fc.hp.com:crwit
user:root@swelter.fc.hp.com:crwit
group:swadm:crwit
```

Installing the *AgentConfig.SD-CONFIG* file set also modifies the ACLs for *global_soc_template* and *global_product_template* in similar fashion so that changes are included in future ACLs that you may construct.

One major feature of SCM is that the SCM Role facility completely manages all necessary authorizations for integrated SCM applications on the CMS system. This includes SD ACLs and hides them from view while you use SCM. When you configure an SCM managed node by installing the *SD-CONFIG.AgentConfig* file set,

you have set up the ACLs as described above. From that point on, you can use the SCM Roles facility on the CMS system to authorize non-root users. You don't have to set up the ACLs manually using SD ACLs.

Example 10. Add user *dennis* as an authorized SCM user on the CMS, and allow him to fully manage *swcrunch*:

```
swelter : root $ mxuser -a -u dennis
swelter : root $ mxauth -a -u dennis -R "Master Role"
-n swcrunch
```

PUTTING IT ALL TOGETHER

SD ACLs give you a powerful tool for controlling access to SD host, root, and depot objects in a flexible manner. As an administrator, you can set up authorizations that let non-

superusers perform SD operations, or that allow or deny access to users on remote systems.

The examples given here have shown some of the common uses of SD ACLs. Armed with this knowledge, you can better understand ACLs and avoid the commonly misunderstood aspects of their behavior. In particular, you should remember the difference between *host* and *root* ACL entries, and know how these differ from the *host entry_type*.

HP hopes that you will find these tools helpful in running your systems efficiently and securely.

FOR MORE INFORMATION

- The *swacl(1M)* man page for HP-UX 11i contains additional examples, particularly for applying ACLs to individual products within depots.
- You can find complete HP-UX documentation at <http://docs.hp.com>, including the SD manuals, listed under the System Administration heading for each OS:

Managing HP-UX Software with SD-UX (for 10.20 and 11.00)

Software Distributor Administration Guide for HP-UX 11i

- For more information about Software Distributor, see <http://hp.com/go/sd>.
- For more information about using ServiceControl Manager for a broad array of tasks and for free downloads, see <http://hp.com/go/servicecontrol>.
- Interex members can refer to the conference proceedings for the InterWorks 2001 conference for several papers that describe SCM in greater depth. See <http://www.interex.org/conference/iworks2001/proceedings/home.html>.

Al Miller is a software development engineer with HP's UNIX Systems Enablement Lab in Fort Collins, Colorado. He has a BS degree in physics and computer science from the University of Wisconsin, River Falls, and a Ph.D. in physics from Colorado State University. He has been with HP for five years.