Understanding SD-UX ACLs

Al Miller

Hewlett Packard Company
3404 E Harmony Road, MS57
Fort Collins CO 80528

(970) 898-7466 Telephone
(970) 898-2838 Fax

al_miller@hp.com

Abstract

This presentation explains how system administrators can use HP Software Distributor (SD-UX) Access Control Lists (ACLs) to control access to SD depots, to enable remote management of systems, and to delegate software management authority to non-superusers. Detailed topics include:

* The SD "swacl" command
* Distinctions between "hosts" and "roots"
* The meaning of the five SD ACL permissions
* The distributed software management capabilities provided by SD and the HP ServiceControl Manager (SCM) product
* The relationship between SD ACLs and SCM

Real-world examples will show HP-UX system administrators how to use SD ACLs to protect systems and depots, how to delegate tasks to non-superusers, and how to use SCM or the SD-UX "push" capability to manage software on remote systems via a central management server.

Introduction

Software Distributor allows system administrators to package, distribute, install, verify, and remove software on their systems. While the single most common use model is for the local superuser to perform these operations, SD allows for more flexibility. Using SD Access Control Lists, the system administrator may authorize non-superusers, including users on remote systems, to perform SD tasks. As data center environments become more complex, the utility provided by these features may become increasingly appealing.

Prior to HP-UX 11i, some Software Distributor features, including the ability to manage remote systems and schedule jobs, were only available to customers who purchased the HP OpenView Software Distributor license. On 11i, these features may now be enabled for free, and using ACLs helps fully realize the capabilities.

The ServiceControl Manager product may also be used to perform software distribution tasks. Installation of SCM causes special ACLs to be installed on the system, removing the need for system administrators to manually manage ACLS when using applications through SCM.

The examples shown in this paper were developed primarily using HP-UX 11.00 (`swcrunch`), and HP-UX 11i (`swelter`) systems, and the quoted documentation is from HP-UX 11i. Complete documentation may also be found on http://docs.hp.com .

More information about Software Distributor may be found at http://hp.com/go/sd and at http://docs.hp.com. More information about ServiceControl Manager is located at http://hp.com/go/servicecontrol.

The `swacl` command

Software Distributor ACLs are displayed and managed using the `swacl` command, which requires at minimum an argument indicating 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] ...
```

The majority of this paper will discuss the `level` and `acl_entry` arguments, and demonstrate how they may be used to control access between local and remote systems.

Example 0: 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 filesystem is owned by the `root` user, and that as such, the owner has full ACL permissions (`crwit`). Additionally, all other users may read SD information about this root filesystem using the `swlist` command. The default ACLs have changed slightly on HP-UX 11i compared to earlier releases: The group ACL for `swadm` has been deleted, and the ‘`t`’ permission removed from `any_other`.

ACL levels: Hosts, Roots, Depots, and Products and Templates

SD host objects correlate to an individual system, and may contain roots and depots. The most common (in fact universal) example of a root is "/". All software installed by SD is written to a root filesystem, and recorded in the Installed Product Database. Software installed on a root is organized into products, filesets, subproducts, and bundles, whose properties may be displayed with the `swlist` command.

Other roots may exist elsewhere, and are known as "alternate roots". Alternate roots were utilized for diskless cluster support on HP-UX 10.X. Because diskless clusters are not supported after HP-UX 10.20, alternate roots are no longer commonly used. It is important to note that alternate roots are not used to install software to locations other than the default location. For this feature, the `is_locatable` attribute is used when the software is packaged, and the application must be implemented to support locateability.

In addition to roots, hosts may also contain SD depots, which are repositories from which software is installed. Like roots, depots are protected by ACLs. Unlike roots, products within depots also contain ACLs, allowing administrators to allow or deny access to products contained within depots.

ACLs protecting hosts, roots, depots, and products within depots affect objects which already exist on the system. In addition, there are ACLs called "templates" which affect objects to be created in the future. These templates are global_soc_template, global_product_template, and product_template.

global_soc_template provides default ACLs which will apply to all new depots and roots added to the host. global_product_template is the template used to initialize 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.

There are five ACL permissions, described as follows, as described in the `swacl(1M)` man page:

```
 r ead       Grants permission to read the object.  On host,
               depot, or root objects, read permission allows swlist
               operations.  On products within depots, read
               permission allows product files to be installed or
               copied with swinstall or swcopy.

  w rite      Grants permission to modify the object itself.
               +  On a root object (e.g. installed root filesystem),
                  this also grants permission to modify the products
                  installed (contained) within it.
               +  On a depot object, it does not grant permission to
                  modify the products contained within it. Write
                  access on products is required to modify products
                  in a depot.
               +  On a host container, write permission grants
                  permission to unregister depots.  It does not
                  grant permission to modify the depots or roots
```

```
                 contained within it.
   i nsert     On a host object, grants permission to create
              (insert) a new software depot or root filesystem
              object, and to register roots and depots.  On a depot
              object, grants permission to create (insert) a new
              product object into the depot.
   c ontrol    Grants permission to modify the ACL using swacl.
   t est       Grants permission to perform access checks and to
              list the ACL.
   a ll        A wildcard which grants all of the above permissions.
              It is expanded by swacl to crwit.
```

It is important to remember that the meaning of some of these permissions depends subtly upon which type of object the ACL is protecting. In particular, pay close attention to the difference between write and insert when applied to host objects. In this context, write permission on a host object simply allows an existing depot to be unregistered. To create a new depot, or register an existing depot, insert permission is required.

ACL Entries

The `acl_entry` field supports several `entry_types`: `any_other, group, host, object_owner, object_group, other,` and `user`. Particular attention should be paid to the distinction between the `user` and `host` entry types. `user` entry types control actions performed by a particular user running SD controller commands such as `swinstall`, `swremove`, and `swcopy`. However, `host` entry types affect the permissions for SD agents (`swagent` processes). ACLs with `entry_type host` are used primarily to control access to software depots.

Example 1. In this example, root will give user `allen` insert permission on the host. This will allow him to create new depots (and alternate roots), but he will not be able to install software to the default root “/”. For the first example, all command output from the SD commands will be displayed. For subsequent examples, only the command lines and highly relevant output will be kept.

```
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 will then package into a depot:

```
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

swelter : allen $ swreg -l depot /simple_1.depot

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

       * Session started for user "allen@swelter".

       * Beginning Selection
       * Targets:                swelter
       * Objects:                /simple_1.depot
       * Selection succeeded.



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

swelter : allen $ swlist -d @ /simple_1.depot
# Initializing...
# Contacting target "swelter"...
#
# Target:  swelter:/simple_1.depot
#

#
# No Bundle(s) on swelter:/simple_1.depot
# Product(s):
#

  test_product                          very simple test product

swelter : allen $ swacl -l depot @ /simple_1.depot
#
# swacl    Depot Access Control List
#
# For depot:  swelter:/simple_1.depot
#
# Date:  Wed Feb 28 16:54:42 2001
#

# Object Ownership:  User= allen
#                    Group=users
#                    Realm=swelter.fc.hp.com
#
# default_realm=swelter.fc.hp.com
object_owner:crwit
any_other:-r---

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

```
=======  02/28/01 16:58:50 MST  BEGIN swinstall SESSION
         (non-interactive) (jobid=swelter-0010)

       * Session started for user "allen@swelter".

       * Beginning Selection
ERROR:   "swelter:/":  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 "swelter:/".
ERROR:   More information may be found in the daemon logfile on this
         target (default location is swelter:/var/adm/sw/swagentd.log).
       * Selection had errors.



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


swelter : allen $ swinstall -s /simple_1.depot test_product @ /altroot/

=======  02/28/01 17:00:06 MST  BEGIN swinstall SESSION
         (non-interactive) (jobid=swelter-0012)

       * Session started for user "allen@swelter".

       * Beginning Selection
       * "swelter:/altroot/":  This target does not exist and will be
         created.
       * Source connection succeeded for "swelter:/simple_1.depot".
       * Source:                 /simple_1.depot
       * Targets:                swelter:/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 "swelter:/altroot/".
       * Analysis and Execution succeeded.


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

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

swelter : 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  HPUX 11.11 Support Tools Bundle

swelter : allen $ swlist @ /altroot
# Initializing...
# Contacting target "swelter"...
#
# Target:  swelter:/altroot
#

#
# No Bundle(s) on swelter:/altroot
# Product(s):
#

  test_product                         very simple test product
```

Example 2. Enable user `dennis` to install software into the default root.

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

"`r`" allows `dennis` the ability to open the root for reading, and "`i`" gives him the permission to insert the new product into the root object.

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

`dennis` may also remove this and any other product installed on the root filesystem.

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

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. The ability of these users to list installed software, access depots, and even install and remove software may be flexibly controlled.

Example 3. Disallow remote systems to `swlist` the local system.

By default, the “`any_other:-r---`“ root ACL allows all users on your network to `swlist` 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
```

This behavior may be modified by removing 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 4. Allow a remote user (`allen@swelter`) to fully manage the root filesystem 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 5: 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 @ /simple_1.depot
swelter : root $ swacl -l depot -M other:r @ /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 are now able to access this depot as a result of the "`other`" ACL, but remote users are refused. Notice above, that even though user 'root' is not the owner of the `/simple_1.depot` object, and was not given any ACLs, it was able to perform this action. This is because ACLs are not checked when SD commands are invoked by a local superuser.

Example 6. Allow only user `shelly` on host `swcrunch` to access software in a depot located on `swelter`:

At first glance, it may appear that adding a `user` ACL for `shelly` would be sufficient for allowing access to this depot:

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

However, this alone is not enough. An attempt by `shelly` to access this depot will fail with a security violation. This is because, in addition to identifying the user (`shelly` on system `swcrunch`), SD also requires that SD agents (the `swagent` process) contacting depot servers be authorized via a "`host`" ACL `entry_type`:

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

Now, `shelly`'s `swinstall` will succeed, presuming she has been given appropriate ACL permission to install software on `swcrunch`:

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

However, another user, `teresa`, also on `swcrunch`, still may not access this depot:

```
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 `user` ACLs are required in addition to the host ACL for depot access, a wildcard '*' option was host ACL `entry_type` beginning with HP-UX 11i:

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

This eliminates the need for multimple "`-M host`" `swacl` command invocations, and `user` ACL `entry_types` are still required.

Example 7. Demonstrate that remote `root` users are treated like any other remote user.

In Example *4* above, the `any_other` ACL was removed from `/simple_1.depot` on `swelter`, and it was demonstrated that the local `root` user still had unrestricted access to the depot. However, remote `root` users are not granted the same privilege. In the present state, `root@swcrunch` will be denied access to this depot, and must be granted ACL permission just like any other remote user.

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

These examples have demonstrated some of the uses of SD ACLs, and have attempted to clarify the distinctions between some of their use models. The `swacl(1M)` man page on HP-UX 11i contains additional examples, particularly with applying ACLs to individual products within depots.

ServiceControl Manager and SD ACLs

ServiceControl Manager allows single-point, multi-system, configuration management, providing a number of sophisticated management tools. It provides mechanisms for managing tools already designed to deal with multiple systems, such as Ignite UX and Software Distributor, and removes the need for system administrators to deal with many of the individual ACL tasks described above. However, it may be useful for the system administrator to understand some of the SD ACL management steps performed by SCM.

SCM environments consist of Central Management Servers and Managed Nodes. For the following examples, `swelter` will be the CMS, and is running HP-UX 11i. By default, SCM is delivered and installed with 11i, although some configuration is necessary. `swcrunch`, the managed node, is running HP-UX 11.00, must have SCM installed. SCM is available for free from http://hp.com/go/servicecontrol. After installing SCM and the prerequisite patch bundle (also available at the above web site), `swcrunch` is then configured to be a managed node. SCM managed nodes grant authorization to the CMS by installing a special fileset, `AgentConfig.SD-CONFIG` from a depot on the CMS system. That depot is `swelter:/var/opt/mx/depot11`.

Example 8. Examination of the `host` and `root` ACLs on `swcrunch` before installation of this product show:

```
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. Now we install the `AgentConfig.SD-CONFIG` fileset:

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

and 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
```

In addition to these ACLs, the ACLs for `global_soc_template` and `global_product_template` are also modified in an analogous manner by this installation.

A major feature of SCM is that when an application is integrated into SCM on the CMS system, the SCM Role facility is used to completely manage all necessary authorizations, including SD ACLs. SD ACLs become completely hidden from the SCM user's view. When an SCM managed node is configured, installation of the `SD-CONFIG.AgentConfig` fileset sets up the ACLs as described above, and from that point on the CMS system may authorize non-root users through SCM Roles, rather than directly through underlying technologies such as SD ACLs. Therefore, when managing systems through SCM, direct manipulation of ACLs is not necessary.

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
```

For more information on how SCM may be used for a broad array of tasks, see the SCM website http://hp.com/go/servicecontrol. In addition, there are several other papers presented in this volume which describe SCM at a depth beyond the scope of this paper.

Conclusions

Software Distributor provides a powerful ACL facility which allows system administrators to control access to SD host, root, and depot objects in a flexible manner. The administrator may assign authorization to non-superusers to perform SD operations, and may allow or deny access to users on remote systems.

This paper has shown several common use cases of SD ACLs, with the goal of illuminating some simple but commonly misunderstood aspects of their behavior. In particular, the difference between `host` and `root` ACL entries was demonstrated, and compared to the host `entry_type`.

The new HP ServiceControl Manager product also provides an interface to software distribution, and in fact completely hides the necessity for system administrators to directly manipulate ACLs. The SCM role-based authorization system allows the manager of a central management server to delegate responsibility for a wide range of tasks to a flexible array of users.