Restricting Service Administration in the Solaris™ 10 Operating System

Glenn Brunette, Client Solutions

Sun BluePrints™ OnLine
June 2005

Part No. 819-2887-10
Revision 1.0
Edition: June 2005

Restricting Service Administration in the Solaris™ 10 Operating System

This Sun BluePrints™ Cookbook describes how to use the Solaris™ 10 Service Management Facility (SMF) to require specific authorizations for certain types of operations. Using this capability, it is possible to delegate access to core service management functions based on the concept of *least privilege*—if a user or service does not strictly need to have some degree of privilege, then that privilege should not be granted. SMF allows organizations to have much finer grained access control policies than was possible before the Solaris™ 10 Operating System (Solaris OS).

While this type of information is generally discussed in the Solaris OS product documentation and other forums, this article describes how to limit service administration in a practical context using a real service as an example.

Apache2 Service Example

The example used in this article is the Apache2 service that is available in Solaris 10. This Apache2 service will run inside a Solaris container. The details of how to configure and install a Solaris container are outside of the scope of this document. The container used in this example is based on the default Sun template—it was changed only to define the container root directory (that is `zonepath`) and to define network interfaces. Unless otherwise specified, all of the commands used later in this article are run from a root shell within the container.

The Apache2 service is available in the SUNWapch2r and SUNWapch2u packages. It is identified by the SMF Fault Management Resource Identifier ("FMRI") as: `svc:/network/http:apache2`. In this article, the abbreviated name of `apache2` will be used to refer to this service in any SMF commands.

Why Restrict Service Administration?

The answer to this question requires an understanding of how services were managed prior to Solaris 10 and the introduction of SMF. In previous Solaris releases, services were started by a run-control script, `inetd(1M)`, or perhaps directly by `init(1M)`. Root access was likely required to change a service configuration managed by any of these mechanisms. In addition, root access was usually required to add, remove, enable, disable, start, stop, or restart a service.

Another issue in previous Solaris versions was that root access often needed to be granted to a program, such as an editor, in order to change service configuration parameters, such as a command line option for a given service. Alternatively, an administrator could have used file-based access control lists to grant write access to an otherwise protected file. Each of these approaches was problematic because it allowed a user to modify any part of the run-control script, adding, removing, or changing content without restriction.

Obviously, this model posed quite a challenge for organizations wanting to take control of their services, enforce strong change management policies, and maintain a degree of assurance with respect to their service configurations. The introduction of SMF in Solaris 10 breaks this mold by creating new capabilities and opportunities for organizations to manage access to service management tasks.

Authorizations

SMF leverages the Solaris 10 role-based access control ("RBAC") facility to delegate access to SMF administrative tasks. By integrating with RBAC in this way, a consistent approach for privilege delegation is applied throughout the system, whether a service is started by SMF or a command is run by a user or role who has been given specific authorizations or privileges. This section describes the primary methods for granting access to SMF administrative functions.

Main Authorizations for Granting Privileges

SMF uses two primary authorizations to grant additional privileges to users or roles.

TABLE 1 Primary Authorizations for Granting Additional Privileges to Users or Roles

Authorization	Description
`solaris.smf.modify`	Used to add, delete, or modify services, service instances, or their properties. This is a very powerful authorization that is typically reserved for service administrators. Be sparing in how this authorization is granted, as it can be used to facilitate a privilege escalation attack by allowing a user (who has been granted this authorization) to effectively run arbitrary commands as root (or as any other user) by simply manipulating a service's configuration.
`solaris.smf.manage`	Used to request restart, refresh, or other state modification of any service instance. This authorization is typically reserved for service operators who are not given the authority to add or remove services, or to alter the way in which services are configured. By default, Solaris 10 also includes several service specific management authorizations, such as `solaris.smf.manage.bind` and `solaris.smf.manage.power`, that apply the `solaris.smf.manage` authorization to a given service (as the name suggests).

These authorizations represent an important advancement in Solaris 10 because they allow organizations to more easily separate the traditional roles of administrator and operator.

Property Group-Specific Authorizations

To provide additional granularity, SMF offers the following property group-specific authorizations. These can be used to subdivide the authority of the `solaris.smf.modify` authorization to more finely control which settings can be changed.

TABLE 2 Property Group-Specific Authorizations

Authorization	Description
`solaris.smf.modify.method`	Permits changing values or creating, deleting, or modifying a property group of type `method`. Each service or service instance must define a set of methods that start, stop, and (optionally) refresh the service. See the `smf_method(5)` for a more complete description of the method conventions for `svc.startd(1M)` and the similar `fork(2)` and `exec(2)` restarters.
`solaris.smf.modify.dependency`	Permits changing values or creating, deleting, or modifying a property group of type `dependency`. Service instances may have dependencies on services or files. Those dependencies govern when the service is started and automatically stopped.
`solaris.smf.modify.application`	Permits changing values or creating, deleting, or modifying a property group of type `application`. This property group is reserved to store application-specific properties.
`solaris.smf.modify.framework`	Permits changing values or creating, deleting, or modifying a property group of type `framework`.

Determining the Property Group Type

To determine the type of a given property group, use the `svccfg(1M)` command, as shown in the following example:

```
# svccfg -s cron listpg
usr             dependency
ns              dependency
general         framework
dependents      framework
startd          framework
start           method
stop            method
tm_common_name  template
tm_man_cron     template
tm_man_crontab  template
```

Note that the granularity of these authorizations is at the level of property group type. However, this does not restrict what service can be modified. As a result, granting one of these authorizations (such as `solaris.smf.modify.method`) would allow a user or role to add, change, or remove any property for any service—as long as the property group being manipulated was a member of a property group of type `method`.

RBAC Rights Profiles

By default, Solaris 10 ships with the following two new RBAC rights profiles, which can be used to leverage these capabilities out of the box.

TABLE 3 RBAC Rights Profiles

RBAC Rights Profile	Description
Service Management	A user or role assigned this rights profile can manipulate any service in any way. It corresponds to the `solaris.smf.manage` and `solaris.smf.modify` authorizations.
Service Operator	A user or role assigned this rights profile has the ability to enable or disable any service instance on the system, as well as to request that its `restart` or `refresh` method be executed. It corresponds to the `solaris.smf.manage` and `solaris.smf.modify.framework` authorizations.

For more information, see the `smf_security(5)` manual page.

Service-Specific Property Group Authorizations

Many organizations typically want more fine-grained control over service management functions. To accommodate those organizations, SMF provides a greater level of access control through the use of service-specific property group authorizations, which can be bound to specific property groups of a service. This technique can be used to further limit which property groups within which services can be created, modified, or deleted. To use service-specific property group authorizations, create one or more of the following properties under the property group to which access should be controlled.

TABLE 4 Service-Specific Property Group Authorizations

Authorization	Description
`action_authorization`	Permits a user to perform a state modification request (such as restart, refresh, mark or clear) for a given service. This is the least privileged of the authorizations because it does not permit service profile modification. Note that this authorization is meaningful only when it is defined within the general property group of a service.
`modify_authorization`	Permits the addition, deletion, or modification of properties within the property group. This is the most privileged authorization because it permits the creation, modification, and removal of service properties.
`value_authorization`	Permits changing the values of any existing property within the property group, with the exception of the `modify_authorization` property.

The authorization type to use depends on the level of access that should be provided. For example, to allow an administrator to change command line flags associated with a service's `start` method, use `value_authorization`. Doing so will prevent the administrator from adding or deleting properties in the `start` method, as well as manipulating other property groups of that service. For more information, see the `smf_security(5)` manual page.

Configuring Service Administration for the Apache2 Service

This section describes the following steps to configure service administration for the example Apache2 service:

- Step 1: Create Separate Administrative Roles
- Step 2: Create Authorizations for webadm
- Step 3: Configure the Apache2 Service With the Required Authorizations and Reduced Privileges
- Step 4: Configure and Enable the Apache Service
- Step 5: Verify that the Apache2 Service Has Been Started Correctly

Instead of running the service management commands as the root user, the instructions in this section configure two new roles (`svcadm` and `webadm`) for the purpose of separating service administration and web server management, respectively.

Step 1: Create Separate Administrative Roles

This step involves creating the `svcadm` and `webadm` roles and assigning them to an existing account (`gmb`). An organization should assign the roles to those users who are responsible for performing service administration and web server administration tasks, respectively. In this step, the `svcadm` role will be assigned the Service Management rights profile, described in "RBAC Rights Profiles" on page 4, in order to grant to `svcadm` the privileges necessary to be a service administrator.

Create the Roles

As root, create the new roles:

```
# mkdir -p -m 755 /export/home
# roleadd -P "Service Management" -d /export/home/svcadm -m svcadm
# passwd svcadm
New Password:
Re-enter new Password:
passwd: password successfully changed for svcadm

# roleadd -g webservd -d /export/home/webadm -m webadm
# passwd webadm
New Password:
Re-enter new Password:
passwd: password successfully changed for webadm

# usermod -R svcadm,webadm gmb
```

Verify the Roles

To verify that the roles have been correctly created and assigned, use the following commands.

```
# getent passwd svcadm webadm
svcadm:x:104:1::/export/home/svcadm:/bin/pfsh
webadm:x:102:80::/export/home/webadm:/bin/pfsh

# egrep "^svcadm:|^webadm:" /etc/user_attr
webadm::::type=role;profiles=All
svcadm::::type=role;profiles=Service Management

# roles gmb
svcadm,webadm
```

This output confirms that the `svcadm` and `webadm` roles have been created and assigned to the `gmb` user.

Step 2: Create Authorizations for webadm

This step creates two new authorizations that will be granted to the `webadm` role:

- `sunw.smf.manage.http/apache2`. This authorization will be used to allow `webadm` to request restart, refresh, or other state modification of the Apache2 service. It will be assigned to the `general/action_authorization` property of the Apache2 service.
- `sunw.smf.modify.application.http/apache2`. This authorization will be used to allow `webadm` to change application-specific properties of the Apache2 service. For example, the Apache2 service provides an application property to enable or disable the use of SSL. This authorization will be assigned to the `httpd/value_authorization` property of the Apache2 service.

Note – These names are just arbitrary text strings. In general, authorization strings should be self-descriptive and begin with characters that uniquely identify your organization in order to avoid potential conflicts with authorizations provided by Sun or other companies.

Create the Authorizations

To create these authorizations, open the `/etc/security/auth_attr` file in an editor and add the following lines:

```
# grep '^sunw.smf.manage.http/apache2' /etc/security/auth_attr
sunw.smf.manage.http/apache2:::Manage the Apache2 Service::

# grep '^sunw.smf.modify.application.http/apache2' /etc/security/auth_attr
sunw.smf.modify.application.http/apache2:::Modify the Apache2 Application
Properties::
```

Define and Assign Authorizations to root

In order for the root user to be able to grant these authorizations using the `rolemod(1M)` command, two other authorizations must be defined and granted to root: `sunw.*` and `sunw.grant`. Otherwise, the root account would not have authority to grant any of the `sunw.*` authorizations to other users or roles on the system.

Note – To add the following authorizations, simply open the `/etc/security/auth_attr` file in an editor and add the lines manually. There is no command line interface available in Solaris today to automate the creation of authorizations.

After adding these new authorizations, the `/etc/security/auth_attr` file should include:

```
# grep "^sunw\." /etc/security/auth_attr
sunw.*:::My Sample Authorizations::
sunw.grant:::Grant My Sample Authorizations::
sunw.smf.manage.http/apache2:::Manage the Apache2 Service::
sunw.smf.modify.application.http/apache2:::Modify the Apache2
Application Properties::
```

These two new authorizations must now be assigned to the root user by manually modifying the `/etc/user_attr` entry for that account:

```
# grep "^root:" /etc/user_attr
root::::auths=solaris.*,solaris.grant,sunw.*,sunw.grant;profiles=We
b Console Management,All;lock_after_retries=no
```

Grant SMF-Specific Authorizations to webadm

This step uses the `rolemod(1M)` command to grant SMF-specific authorizations to the `webadm` user.

```
# rolemod -A sunw.smf.manage.http/apache2,sunw.smf.modify.application.http/
apache2 webadm
```

Use the `auths(1)` command to verify that the new authorizations have been added.

```
# auths webadm
webadm : sunw.smf.manage.http/apache2,sunw.smf.modify.application.http/
apache2,solaris.device.cdrw,solaris.profmgr.read,solaris.jobs.users,solaris.mail.
mailq,solaris.admin.usermgr.read,solaris.admin.logsvc.read,solaris.admin.fsmgr.re
ad,solaris.admin.serialmgr.read,solaris.admin.diskmgr.read,solaris.admin.procmgr.
user,solaris.compsys.read,solaris.admin.printer.read,solaris.admin.prodreg.read,s
olaris.admin.dcmgr.read,solaris.snmp.read,solaris.project.read,solaris.admin.patc
hmgr.read,solaris.network.hosts.read,solaris.admin.volmgr.read
```

Note that there are many authorizations defined that were not listed in the role's `user_attr` entry. These additional authorizations are by default given to all users on the system and are defined in the `AUTHS_GRANTED` and `PROFS_GRANTED` parameters in the `/etc/security/policy.conf` file.

Step 3: Configure the Apache2 Service With the Required Authorizations and Reduced Privileges

This step configures the Apache2 service to:

- require authorizations for specific functions
- operate with reduced privileges

These actions will be performed by the Service Administrator (`svcadm`). Unless otherwise specified, the `svcadm` and `webadm` roles used in this document will be assumed directly by the `gmb` user, although the actual `su(1M)` command will not be mentioned.

The `svcadm` role is used to control overall service administration in order to prevent granting too much privilege to application administrator accounts such as `webadm`. If the necessary authorizations were given to `webadm` to perform the following steps, it would be possible for `webadm` to change key properties under the `start`, `stop`, or `restart` methods that, in turn, would allow `webadm` to execute arbitrary commands as root or other users.

The Apache2 service is configured using the `svccfg(1M)` command.

```
svcadm$ svccfg -s apache2
```

Install the Properties

Use the `svccfg` command to install the `httpd/value_authorization` and `general/action_authorization` properties.

```
svc:/network/http:apache2> setprop httpd/value_authorization = astring:
sunw.smf.modify.application.http/apache2
svc:/network/http:apache2> setprop general/action_authorization = astring:
sunw.smf.manage.http/apache2
```

Configure Reduced Privileges

Use the following commands to configure the Apache2 service to operate with reduced privileges.

```
svc:/network/http:apache2> setprop start/user = astring: webservd
svc:/network/http:apache2> setprop start/group = astring: webservd
svc:/network/http:apache2> setprop start/privileges = astring:
basic,!proc_session,!proc_info,!file_link_any,net_privaddr
svc:/network/http:apache2> setprop start/limit_privileges = astring: :default
svc:/network/http:apache2> setprop start/use_profile = boolean: false
svc:/network/http:apache2> setprop start/supp_groups = astring: :default
svc:/network/http:apache2> setprop start/working_directory = astring: :default
svc:/network/http:apache2> setprop start/project = astring: :default
svc:/network/http:apache2> setprop start/resource_pool = astring: :default
```

After changing the Apache2 service configuration, exit the `svccfg` command.

```
svc:/network/http:apache2> end
```

Refresh the Apache Service

After making configuration changes, refresh the Apache2 service so that the changes take effect.

```
svcadm$ svcadm refresh apache2
```

Verify Configuration Changes to Property Groups

Verify configuration changes using the `svcprop(1)` command. Rather than list every property, the following commands simply list the properties associated with the `general`, `start` and `httpd` property groups—that is, the property groups that were actually modified:

```
svcadm$ svcprop -p general apache2
general/enabled boolean true
general/action_authorization astring sunw.smf.manage.http/apache2
general/entity_stability astring Evolving
svcadm$ svcprop -p start apache2
start/exec astring /lib/svc/method/http-apache2\ start
start/timeout_seconds count 60
start/type astring method
start/user astring webservd
start/group astring webservd
start/privileges astring
basic,!proc_session,!proc_info,!file_link_any,net_privaddr
start/limit_privileges astring :default
start/use_profile boolean false
start/supp_groups astring :default
start/working_directory astring :default
start/project astring :default
start/resource_pool astring :default
svcadm$ svcprop -p httpd apache2
httpd/ssl boolean false
httpd/stability astring Evolving
httpd/value_authorization astring sunw.smf.modify.application.http/apache2
```

Step 4: Configure and Enable the Apache Service

In the default configuration, the Apache2 service is disabled.

```
svcadm$ svcs apache2
STATE          STIME    FMRI
disabled       13:34:10 svc:/network/http:apache2
```

Before the Apache2 service can be enabled, it needs to be configured. Before configuring the service, several steps must be completed so that a non-root user can perform the service configuration.

Note – The following steps closely mirror those described in the Sun BluePrints article entitled *Limiting Service Privileges in the Solaris 10 Operating System* (Glenn Brunette, Sun BluePrints Online, May 2005). However, a few deviations from those directions will be necessary in order to delegate access specifically to the `webadm` role.

Change the Ownership of /etc/apache2 to webadm

The first task is to change the ownership of `/etc/apache2` (and its contents) so that the `webadm` account can configure and customize the Apache2 service configuration without requiring additional root access. Because the `/etc/apache2` directory is owned by—and is only writable for—the root account, the following command must be executed as root.

```
# chown -R webadm:webservd /etc/apache2
```

By default, this command affects the following files:

```
# ls -lR /etc/apache2
/etc/apache2:
total 256
-rw-r--r--   1 webadm   webservd    1987 Apr  8 10:42 highperformance-std.conf
-rw-r--r--   1 webadm   webservd    1987 Apr  8 10:42 highperformance.conf
-rw-r--r--   1 webadm   webservd   37519 Apr  8 10:42 httpd-std.conf
-rw-r--r--   1 webadm   webservd   37661 Jan  8 05:38 httpd.conf-example
-rw-r--r--   1 webadm   webservd   12959 Apr  8 10:42 magic
-rw-r--r--   1 webadm   webservd   15020 Apr  8 10:42 mime.types
-rw-r--r--   1 webadm   webservd   10759 Apr  8 10:42 ssl-std.conf
-rw-r--r--   1 webadm   webservd   10996 Apr  8 10:42 ssl.conf
```

Create a new Directory Owned by webservd

The next task is to create a new directory (`/var/apache2/run`) that will be owned by the `webservd` account. Because `/var/apache2` is also owned by—and is only writable for—the root account, the following commands must be executed as root.

```
# mkdir -p -m 775 /var/apache2/run
# chown webservd:webservd /var/apache2/run
```

In the Sun BluePrint article entitled *Limiting Service Privileges in the Solaris™ 10 Operating System* (Glenn Brunette, Sun BluePrints Online, May 2005), the ownership of the log files under `/var/apache2/logs` was also changed to `webservd`. If an Apache2 service has never run on the system, then that directory should contain no files. If access and error log files exist in `/var/apache2/logs`, then the following command must also be performed as root:

```
# chown -R webservd:webservd /var/apache2/logs
```

Install the Apache2 Configuration File

Note – For the sake of simplicity, this article uses the default Apache2 configuration file (`httpd.conf-example`) that is shipped in the `/etc/apache2` directory. Depending on an organization's internal policies and requirements, it might be necessary to customize this file before deploying the Apache2 service in an actual environment.

To install the Apache2 configuration file, as the `webadm` user, run the following commands:

```
webadm$ cp /etc/apache2/httpd.conf-example /etc/apache2/httpd.conf
webadm$ ls -l /etc/apache2/httpd.conf
-rw-r--r--   1 webadm   webservd   37661 Apr  6 16:45 /etc/apache2/httpd.conf
```

Ensure that Files Can Be Created, Modified, and Removed

The `LockFile` and `PidFile` parameters must be modified in the `/etc/apache2/httpd.conf` file in order to ensure that the files defined by those parameters can be created, modified and removed by an Apache2 service that is started as `webservd`. The `/etc/apache2/httpd.conf` file should be modified as follows:

```
webadm$ egrep "^PidFile |^LockFile " /etc/apache2/httpd.conf
LockFile /var/apache2/logs/accept.lock
PidFile /var/apache2/run/httpd.pid
```

Enable the Apache2 Service

After completing these configuration changes, enable the Apache2 service using the `svcadm` role. The `svcadm` account is used as a matter of policy and preference. (Note that, alternatively, this authority could have been granted to the `webadm` role through the assignment of a `general/value_authorization`).

To grant the ability to enable or disable the Apache2 service to web server administrators, complete the following commands.

```
webadm$ svccfg -s apache2 setprop general/value_authorization =
astring: sunw.smf.manage.http/apache2
webadm$ svcadm refresh apache2
```

After the appropriate action authorization has been installed and the service has been refreshed, enable the service using the `svcadm` role.

```
svcadm$ svcadm -v enable -s apache2
svc:/network/http:apache2 enabled.
```

Step 5: Verify that the Apache2 Service Has Been Started Correctly

To confirm that the Apache2 service has been successfully started, use the following command:

```
svcadm$ svcs apache2
STATE          STIME    FMRI
```

```
online         20:04:52 svc:/network/http:apache2
```

Verify that the Apache2 Service Did Not Start as root

Next, verify that the service did not start as root by confirming that there is no httpd parent process running as root (all of the processes should be running as `webservd`).

```
svcadm$ ps -aef | grep httpd | grep -v grep
webservd 11773 11770   0 20:04:53 ?           0:00 /usr/apache2/bin/httpd -k start
webservd 11771 11770   0 20:04:53 ?           0:00 /usr/apache2/bin/httpd -k start
webservd 11775 11770   0 20:04:53 ?           0:00 /usr/apache2/bin/httpd -k start
webservd 11783 11770   0 20:05:06 ?           0:00 /usr/apache2/bin/httpd -k start
webservd 11774 11770   0 20:04:53 ?           0:00 /usr/apache2/bin/httpd -k start
webservd 11770 10965   0 20:04:52 ?           0:00 /usr/apache2/bin/httpd -k start
webservd 11772 11770   0 20:04:53 ?           0:00 /usr/apache2/bin/httpd -k start
```

Attempt to Connect to the Apache2 Service

Next, attempt to connect to the service to verify that it is operational.

```
svcadm$ telnet localhost 80
Trying 127.0.0.1...
Connected to localhost.
Escape character is '^]'.
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Thu, 07 Apr 2005 00:05:12 GMT
Server: Apache/2.0.52 (Unix) DAV/2
Content-Location: index.html.en
Vary: negotiate,accept-language,accept-charset
TCN: choice
Last-Modified: Fri, 04 May 2001 00:01:18 GMT
ETag: "346ee-5b0-40446f80;34704-961-8562af00"
Accept-Ranges: bytes
Content-Length: 1456
Connection: close
Content-Type: text/html; charset=ISO-8859-1
Content-Language: en
Expires: Thu, 07 Apr 2005 00:05:12 GMT

Connection to localhost closed by foreign host.
```

Verify Privileges for the Apache2 Service

Finally, verify that the Apache2 service is in fact running with the privilege set that was configured earlier in this article. Because the `webadm` and `svcadm` roles do not have the necessary privilege to perform this check, you must execute the following command as root or as a similarly authorized user or role:

```
# ppriv -S 11770
11770:  /usr/apache2/bin/httpd -k start
flags = <none>
        E: net_privaddr,proc_exec,proc_fork
        I: net_privaddr,proc_exec,proc_fork
        P: net_privaddr,proc_exec,proc_fork
        L: zone
```

This output confirms that the Apache2 service is running exactly as intended.

Notes About the webadm Role

Now that the Apache2 service is up and running, the `webadm` role is now in a position to exercise its capabilities. This section provides an overview of the capabilities and restrictions of the `webadm` role. This discussion highlights the benefits of using Solaris 10 RBAC to restrict access to service management functions.

Capabilities

Based on the configuration outlined above, the `webadm` role is permitted to perform a variety of service management actions, such as restart, refresh, mark, and clear.

```
webadm$ id -a
uid=102(webadm) gid=80(webservd) groups=80(webservd)
```

The `webadm` role can query the Apache2 service state.

```
webadm$ svcs -v apache2
STATE          NSTATE        STIME    CTID   FMRI
online         -             8:42:02    413 svc:/network/http:apache2
```

The `webadm` role can refresh the Apache2 service. This change can be verified by looking at the service time (`STIME`) parameter, the value of which changes when a service enters its new state.

```
webadm$ svcadm refresh apache2
webadm$ svcs -v apache2
STATE          NSTATE        STIME    CTID   FMRI
online         -             8:42:20    413 svc:/network/http:apache2
```

The `webadm` role can restart the Apache2 service. This change can be verified by looking at both the service time (`STIME`) and contract identifier (`CTID`) parameters, both of which should change when the service is restarted.

```
webadm$ svcadm restart apache2
webadm$ svcs -v apache2
STATE          NSTATE        STIME    CTID   FMRI
online         -             8:42:29    416 svc:/network/http:apache2
```

The webadm role can put the Apache2 service into maintenance mode. As shown below, the STIME and CTID parameters are modified. The CTID parameter has no value defined because the service is not running when it has been put into maintenance mode.

```
webadm$ svcadm mark -I maintenance apache2
webadm$ svcs -v apache2
STATE          NSTATE        STIME    CTID   FMRI
maintenance    -             8:42:42      - svc:/network/http:apache2
```

The webadm role can clear a service, returning it from maintenance into normal operation.

```
webadm$ svcadm clear apache2
webadm$ svcs -v apache2
STATE          NSTATE        STIME    CTID   FMRI
online         -             8:42:57    418 svc:/network/http:apache2
```

Restrictions

The webadm role, however, is not permitted to enable or disable the Apache2 service—this is reserved for the service administrator (svcadm).

```
webadm$ svcadm -v disable -s apache2
svcadm: svc:/network/http:apache2: Couldn't modify "general" property
group (permission denied).
webadm$ svcadm -v enable -s apache2
svcadm: svc:/network/http:apache2: Couldn't modify "general" property
group (permission denied).
```

Similarly, the webadm role is not permitted to add, modify or remove any of the Apache2 service properties—with one exception. The webadm role is permitted to modify values associated with existing properties under the httpd property group.

Attempts to add a new property will fail.

```
webadm$ svccfg -s apache2
svc:/network/http:apache2> setprop general/foo = astring: bar
Permission denied.
```

Attempts to remove an existing property will fail.

```
svc:/network/http:apache2> delprop general/action_authorization
Permission denied.
```

Attempts to modify properties outside of the httpd property group will fail.

```
svc:/network/http:apache2> setprop start/user = root
Permission denied.
```

However, attempts to modify a property within the httpd property group will succeed.

```
svc:/network/http:apache2> setprop httpd/ssl = true
svc:/network/http:apache2> quit
```

Benefits of Limiting the webadm Role

Using the instructions in this article, the `webadm` role has been granted sufficient privileges to successfully perform web server administration for the Apache2 service instance only. The `webadm` role is not able to perform service management functions on other services. This is a significant improvement over the model implemented in previous Solaris OS releases, where such fine-grained control was not possible.

Flexibility with Role Privileges

While the restrictions on the `webadm` role were added to illustrate the model and steps for limiting service access, such restrictions are not required in all cases. In fact, SMF is extremely flexible and can be configured to share authorizations between property groups, services, or even across a range of services. Permissions are determined by how the system is configured. When configuring security, carefully consider which privileges are granted, and to whom they are granted, so that excess privileges are not given away inadvertently.

Depending on an organization's own site policy and requirements, additional roles might need to be created, or privileges granted, in a way that differs from what is described in this article. Some organizations prefer fewer roles with greater authority, while others prefer greater separation of duty. This article takes a middle of the road approach to illustrate this new Solaris 10 capability and to show how it could be applied in any environment.

Complete Apache2 Service Manifest

The following code listing shows the complete updated Apache2 service manifest. The service manifest contains all of the SMF parameters associated with the Apache2 service. It is provided only for completeness for those who might be interested. The service manifest was generated using the `svccfg(1M)` command with its `export` option. This format can also be easily saved, modified, validated, and even imported back into the Service Management Facility.

```
<?xml version='1.0'?>
<!DOCTYPE service_bundle SYSTEM '/usr/share/lib/xml/dtd/service_bundle.dtd.1'>
<service_bundle type='manifest' name='export'>
  <service name='network/http' type='service' version='0'>
    <instance name='apache2' enabled='true'>
      <dependency name='loopback' grouping='require_all' restart_on='error' type='service'>
        <service_fmri value='svc:/network/loopback:default'/>
      </dependency>
      <dependency name='physical' grouping='optional_all' restart_on='error'
        type='service'>
        <service_fmri value='svc:/network/physical:default'/>
      </dependency>
      <exec_method name='start' type='method' exec='/lib/svc/method/http-apache2 start'
        timeout_seconds='60'>
        <method_context>
          <method_credential user='webservd' group='webservd'
            privileges='basic,!proc_session,!proc_info,!file_link_any,net_privaddr'/>
        </method_context>
      </exec_method>
      <exec_method name='stop' type='method' exec='/lib/svc/method/http-apache2 stop'
```

```
        timeout_seconds='60'>
        <method_context/>
      </exec_method>
      <exec_method name='refresh' type='method' exec='/lib/svc/method/http-apache2 refresh'
        timeout_seconds='60'>
        <method_context/>
      </exec_method>
      <property_group name='httpd' type='application'>
        <stability value='Evolving'/>
        <propval name='value_authorization' type='astring'
          value='sunw.smf.modify.application.http/apache2'/>
        <propval name='ssl' type='boolean' value='false'/>
      </property_group>
      <property_group name='startd' type='framework'>
        <propval name='ignore_error' type='astring' value='core,signal'/>
      </property_group>
      <property_group name='general' type='framework'>
        <propval name='action_authorization' type='astring'
          value='sunw.smf.manage.http/apache2'/>
      </property_group>
    </instance>
    <stability value='Evolving'/>
    <template>
      <common_name>
        <loctext xml:lang='C'>Apache 2 HTTP server</loctext>
      </common_name>
      <documentation>
        <manpage title='apache2' section='1M'/>
        <doc_link name='apache.org' uri='http://httpd.apache.org'/>
      </documentation>
    </template>
  </service>
</service_bundle>
```

Conclusion

While this article focused on the Apache2 service, this approach can apply to just about any service. By understanding how to use SMF to control access to its services and functions, organizations are in a better position to delegate access on their systems. Using the authorization capabilities provided in the Solaris 10 OS and integrated with SMF, organizations can select as loose or strict a policy as is needed. This flexibility allows organizations to easily define and apply a policy that meets their requirements and priorities.

References and Related Sources

Publications

- Brunette, Glenn. *Limiting Service Privileges in the Solaris 10 Operating System*. Sun BluePrints OnLine, May 2005.

  http://www.sun.com/blueprints/0505/819-2680.pdf
- Faden, Glenn. "Authorization Infrastructure in Solaris." Sun Developer Network, August 2001.

  http://developers.sun.com/solaris/articles/ais.html
- Sun Microsystems, Inc. "Roles, Rights Profiles, and Privileges. Solaris 10 Product Documentation.

  http://docs.sun.com/app/docs/doc/816-4557/6maosrjfe?a=view
- Sun Microsystems, Inc. "Managing Services". Solaris 10 Product Documentation.

  http://docs.sun.com/app/docs/doc/817-1985/6mhm8o5q9?a=view

Web Sites

- Glenn Brunette's Solaris 10 OS Security Weblog

  http://blogs.sun.com/gbrunett?catname=Solaris%2010%20Security
- Sun's BigAdmin Predictive Self Healing Portal

  http://www.sun.com/bigadmin/content/selfheal/

Command Manual Pages

- Sun Microsystems, Inc. "auths(1) Manual Page", Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5165/6mbb0m9bq?a=view
- Sun Microsystems, Inc. "exec(2) Manual Page", Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5167/6mbb2jafk?q=exec%282%29&a=view
- Sun Microsystems, Inc. "fork(2) Manual Page", Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5167/6mbb2jag7?a=view
- Sun Microsystems, Inc. "inetd(1M) Manual Page", Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-0211/6m6nc66se?q=inetd&a=view
- Sun Microsystems, Inc. "init(1M) Manual Page", Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-0211/6m6nc66sj?a=view
- Sun Microsystems, Inc. "rolemod(1M) Manual Page", Sun Solaris 10 OS Product Documentation

http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqdn?q=rolemod%281M%29&a=view

- Sun Microsystems, Inc. “smf_method(5) Manual Page”, Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5175/6mbba7f3p?a=view

- Sun Microsystems, Inc. “smf_security(5) Manual Page”, Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5175/6mbba7f3r?a=view

- *Sun Microsystems, Inc. “svccfg(1) Manual Page”, Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqhs?a=view

- Sun Microsystems, Inc. “svccfg(1M) Manual Page”, Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqhs?q=su%281M%29&a=view

- Sun Microsystems, Inc. “svcprop(1) Manual Page”, Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5165/6mbb0m9tg?a=view

- Sun Microsystems, Inc. “svc.startd(1M) Manual Page”, Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqhu?a=view

- Sun Microsystems, Inc. “su(1M) Manual Page”, Sun Solaris 10 OS Product Documentation

  http://docs.sun.com/app/docs/doc/816-5166/6mbb1kqhg?q=su%281M%29&a=view

About the Author

Glenn Brunette is a Sun Distinguished Engineer with nearly 15 years’ experience in information security. Glenn works in the Client Solutions division as the Chief Security Architect for the Global Data Center Practice. In this role, Glenn is responsible for global security strategy, as well as for improving the quality and security of consulting solutions delivered to Sun's customers.

Glenn is the co-founder of the Sun Solaris Security Toolkit software and a frequent author and contributor to the Sun BluePrints program. Glenn works closely with teams across Sun on the development of security strategy, products, services, methodologies, best practices, training, certifications, and tools.

Externally, Glenn is currently the Vice-Chair of the Enterprise Grid Alliance Security Working Group and has served as Champion for the Common Configurations Working Group of the National Cyber Security Partnership's Technical Standards and Common Criteria Task Force. Glenn is also an active contributor to the Center for Internet Security's Unix Benchmark team.

Glenn is a Certified Information Systems Security Professional (CISSP) and has been trained in the National Security Agency's INFOSEC Assessment Methodology (IAM).

Acknowledgements

The author would like to thank the following people for their inspiration, technical feedback, and overall support in the development of this article: Jonathan Adams, Ganesh Chaudhari, Mikael Lofstrand, Liane Praza, Mike Ramchand, and Collin Sampson.

Ordering Sun Documents

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

Accessing Sun Documentation Online

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

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