



OpenSolaris の セキュリティ 入門編 その2

Jan. 2009

増月 孝信
OpenSolaris エバンジェリスト

Agenda

- スーパーユーザモデルの問題点
- Process Rights Management
 - 最小特権モデル
- User Rights Management
 - Role-Based Access Control (RBAC)

以前の UNIX(Solaris) のスーパーユーザモデル

- 従来 UNIX の特権モデルは全ての特権が uid 0 に関連付けられていた
 - 特権 (Privileges) は処理を実行する為にプロセスが必要とする個々の権利
- このモデルの欠点は **all or nothing** であることである
 - 特権の一部分だけをプロセスに与えることが出来ない
 - 例えば日付設定には file_dac_write 特権と sys_time 特権だけで十分
 - 特権を持つ各プロセスがシステム全体の権利を握る
 - 脆弱な特権プロセスを利用した不正アクセス
 - 一般ユーザの権限を拡張することが出来ない
 - 個々のプロセスが本来必要としている特権が不明確である

Process Rights Management (1)

- カーネルは uid 0 をチェックするのではなく、カーネルで定義された**特権**をチェックするように変更
- あるタスクを完了させるのに必要な特権だけをプロセスに与える
 - **最小特権モデル (Least Privileges)** と呼ぶ
 - スーパユーザモデルの問題点を改善

Process Rights Management (2)

- カーネル内の特権をグループ化して定義
 - ppriv -lv コマンドにてシステムの特権をリスト

```
$ ppriv -lv
```

```
contract_event
```

Allows a process to request critical events without limitation.

Allows a process to request reliable delivery of all events on any event queue.

```
contract_identity
```

Allows a process to set the service FMRI value of a process

contract template.

```
:
```

- 特権定義ファイル
 - /etc/security/priv_names
 - priv_names(4)
- 定義されている特権の説明
 - privileges(5)

Process Rights Management (3)

○ 主な特権の種類

○ FILE 特権

- 文字列 file で始まる特権は、ファイルシステムオブジェクトに対して作用する

○ IPC 特権

- 文字列 ipc で始まる特権は、IPC オブジェクトアクセス制御を無効にする

○ NET 特権

- 文字列 net で始まる特権は、特定のネットワーク機能へのアクセスを可能にする

○ PROC 特権

- 文字列 proc で始まる特権は、プロセスがそれ自体の限定された属性を変更できる

○ SYS 特権

- 文字列 sys で始まる特権は、各種のシステム属性に対する無制限のアクセス権をプロセスに与える

プロセス特権

○ 基本 (Basic) 特権セット

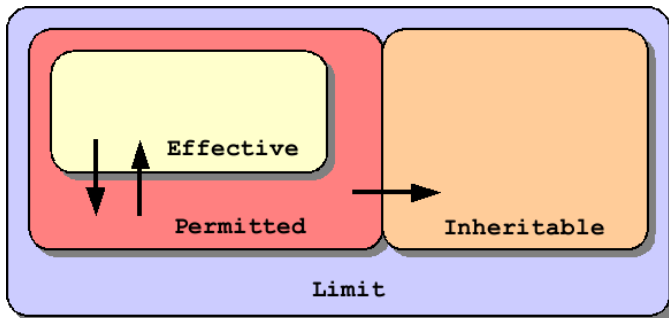
- 従来的一般ユーザプロセスと同等のアクセス権限を含む
- `proc_fork`, `proc_exec`, `proc_session`, `proc_info`, `file_link_any`

○ All 特権セット

- 従来のスーパユーザプロセスと同等のシステム全体の特権を含む

○ 各プロセスは4つの特権セットを持つ

- 実行セット (E): 現在有効な特権
- 継承セット (I): `exec` に継承される特権
- 許可セット (P): プロセスの最大特権セット
- 上限セット (L): プロセスおよびその子プロセスが取得する権限の上限。
この特権セット L に対する変更はすべて次の `exec` に影響する



プロセスの特権を確認する

- ppriv コマンドを使用するとプロセスに割り当てられた特権を確認できる

```
$ ppriv $$
```

```
737: bash
```

```
flags = <none>
```

```
    E: basic
```

```
    I: basic
```

```
    P: basic
```

```
    L: all
```

従来的一般ユーザプロセスの初期値は上記のような設定である

Privilege Aware(PA) と Non-Privilege Aware(NPA)

- 従来のアプリケーションとの互換性を保持
- 従来のプロセスは Non-Privilege Aware として識別

```
# ppriv `pgrep nscd`
```

```
99: /usr/sbin/nscd
```

```
flags = <none>
```

```
E: all
```

```
I: basic
```

```
P: all
```

```
L: all
```

```
# ppriv `pgrep nwamd`
```

```
23: /lib/inet/nwamd
```

```
flags = PRIV_AWARE
```

```
E: basic,file_chown_self,file_dac_read,file_dac_write,net_privaddr,net_rawaccess,proc_audit,proc_owner,  
proc_setid,sys_config,sys_ip_config,sys_ipc_config,sys_net_config,sys_res_config,sys_resource
```

```
I: basic,file_chown_self,file_dac_read,file_dac_write,net_privaddr,net_rawaccess,proc_audit,proc_owner,  
proc_setid,sys_config,sys_ip_config,sys_ipc_config,sys_net_config,sys_res_config,sys_resource
```

```
P: basic,file_chown_self,file_dac_read,file_dac_write,net_privaddr,net_rawaccess,proc_audit,proc_owner,  
proc_setid,sys_config,sys_ip_config,sys_ipc_config,sys_net_config,sys_res_config,sys_resource
```

```
L: all
```


プロセスの必要としている特権を調べる方法

○ ppriv コマンドを用いる

```
$ ppriv -De date 010509102009
```

```
date[1130]: missing privilege "sys_time" (euid = 6985, syscall = 192) needed at secpolicy_settime+0x16
```

```
date: cannot set date: Not owner
```

```
2009 年 01 月 05 日 (月) 09 時 10 分 00 秒 JST
```

```
$ ppriv -vl sys_time
```

```
sys_time
```

Allows a process to manipulate system time using any of the appropriate system calls: stime, adjtime, ntp_adjtime and the IA specific RTC calls.

○ その他 DTrace を利用することも可能である

- <http://www.opensolaris.org/os/community/security/files/privdebug.pl>

特権のデバッグ

- シェルに対して特権のデバッグモードを ON にする

```
$ ppriv -D $$
```

- コマンドを実行すると、デバッグ情報が出力される

```
$ date 010509102009
```

```
date[1154]: missing privilege "sys_time" (eid = 6985, syscall = 192) needed at secpolicy_settime+0x16
```

```
date: cannot set date: Not owner
```

```
2009 年 01 月 05 日 (月) 09 時 10 分 00 秒 JST
```

```
$ touch /testfile
```

```
touch[1157]: missing privilege "ALL" (eid = 6985, syscall = 225) needed at zfs_zaccess+0x1dc
```

```
touch: cannot touch `/testfile': Permission denied
```

- 特権のデバッグモードを OFF にする

```
$ ppriv -N $$
```

```
ppriv[1160]: missing privilege "proc_owner" (eid = 6985, syscall = 5) needed at secpolicy_proc_access+0x16
```

```
ppriv[1160]: missing privilege "proc_owner" (eid = 6985, syscall = 5) needed at secpolicy_proc_access+0x16
```

```
ppriv[1160]: missing privilege "proc_owner" (eid = 6985, syscall = 5) needed at secpolicy_proc_access+0x16
```

```
ppriv[1160]: missing privilege "proc_owner" (eid = 6985, syscall = 5) needed at secpolicy_proc_access+0x16
```

```
ppriv[1160]: missing privilege "proc_owner" (eid = 6985, syscall = 5) needed at secpolicy_proc_access+0x16
```

最小特権を設定する方法 (1)

○ SMF を用いる

```
/var/svc/manifest/system/sar.xml
```

```
:  
<exec_method  
  type='method'  
  name='start'  
  exec='/lib/svc/method/svc-sar %m'  
  timeout_seconds='60'  
  <method_context>  
    <method_credential  
      user='sys'  
      group='sys'  
      privileges='basic,file_dac_write' />  
    </method_context>  
</exec_method>  
:
```

最小特権を設定する方法 (2-1)

○ ppriv コマンドを用いる

`/usr/bin/ppriv -s [AEILP] [+ -] priv1, priv2, ... PID`

- -s : 特権を設定
- A : All
- E : 実行セット
- I : 継承セット
- L : 上限セット
- P : 許可セット
- + : 追加
- - : 削除
- priv1, priv2, ... : カンマで区切られた特権リスト
- PID : プロセスの ID

最小特権を設定する方法 (2-2)

○ ppriv の簡単な事例

```
$ ppriv $$
```

```
1238: -bash
```

```
flags = <none>
```

```
  E: basic
```

```
  I: basic
```

```
  P: basic
```

```
  L: all
```

```
# ppriv -s EIP+sys_time 1238
```

```
$ ppriv $$
```

```
1238: -bash
```

```
flags = <none>
```

```
  E: basic,sys_time
```

```
  I: basic,sys_time
```

```
  P: basic,sys_time
```

```
  L: all
```

最小特権を設定する方法 (3)

- RBAC を用いる
 - 詳細は後述
- システム全体のセキュリティ構成ファイル `/etc/security/policy.conf` を設定
 - 詳細は後述

Process Rights Management まとめ

- カーネルの定義する特権によってプロセスに与える権限をきめ細かに制御可能である
- システムデーモンなどに必要以上の特権を与えなくても良いので、セキュリティリスクを軽減させることができる
- Process Rights Management を学習することによって、より OpenSolaris の動作を理解することが出来る
- まずは `ppriv(1M)` コマンドを実行しながら復習！

User Rights Management

Role-Based Access Control (RBAC)

- 管理タスクを一般ユーザに権限委譲するフレームワーク
 - root パスワードを共有する必要がなくなる
 - 管理者の設定により一般ユーザが特別な権限を持つコマンドを実行可能
 - システムすべての権限を与えるのではない
 - 複数のユーザをコントロールする

ユーザとグループ GUI

The image displays the Solaris User and Group GUI. The main window, titled "ユーザとグループ", shows a list of users. The "masu" user is selected, and the "ユーザ masu の設定" (User masu Settings) dialog is open. The "ユーザのプロファイル" (User Profile) tab is active, showing a list of permissions. The "root" permission is checked, and the "Apache 22 Administration" role is selected in the "ユーザに与えることが可能な権限" (Permissions that can be granted to the user) list.

ユーザとグループ

ユーザ	ホーム・ディレクトリ	ユーザの詳細
svctag	/	Service tag UID
nobody	/	NFS Anonymous Access User
noaccess	/	No Access User
nobody4	/	SunOS 4.x NFS Anonymous Access U
masu	/export/home/masu	Masuzuki
demo	/export/home/d	
admin	/export/home/a	

ユーザ masu の設定

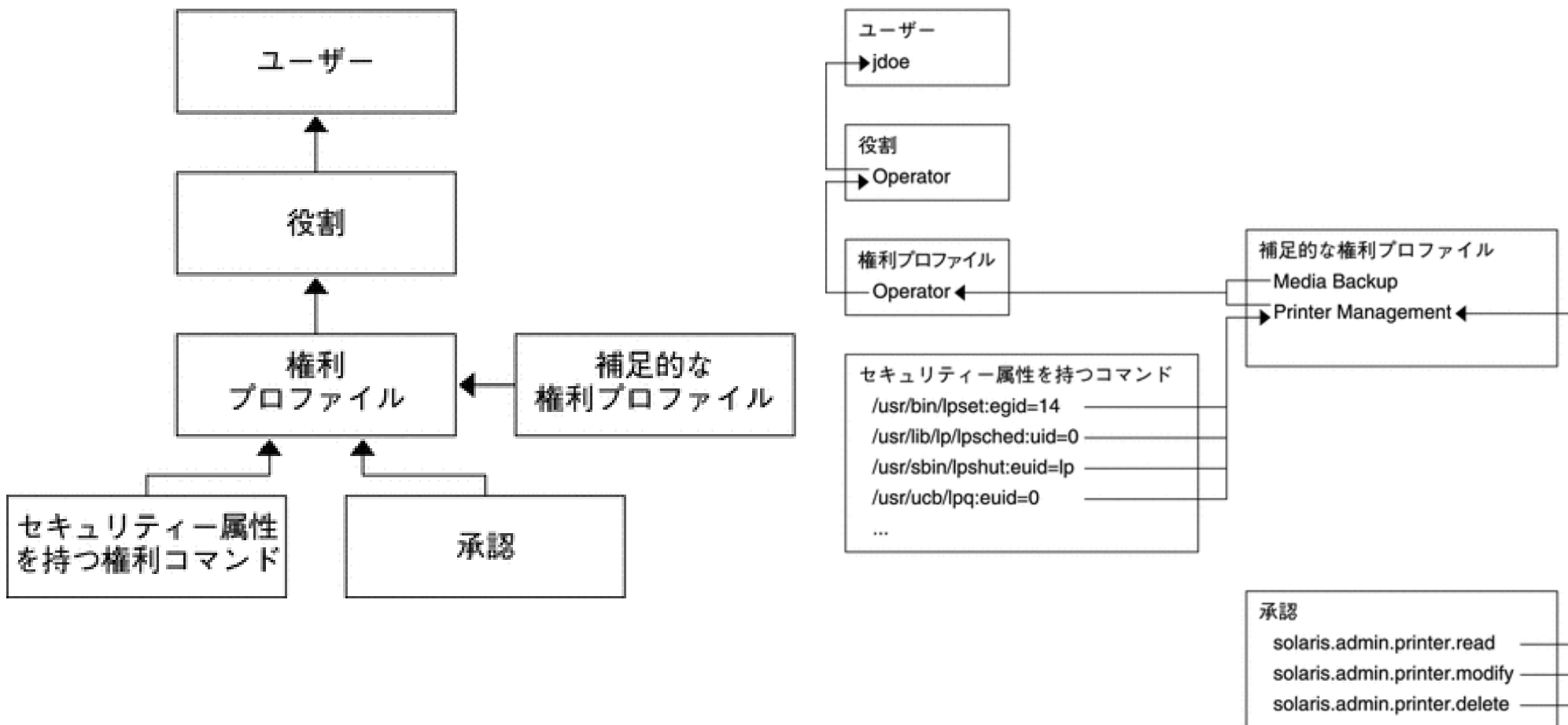
ユーザに与えることが可能な権限:

- admin
- postgres
- root
- zfsnap

ユーザに与えることが可能な権限:

- Apache 22 Administration ()
- Application Server Management (Administrator of Application S
- Audit Control (Configure Solaris Auditing)
- Audit Review (Review Solaris Auditing logs)
- Basic Actions (A minimum set of actions necessary to login thro
- Basic Solaris User (Automatically assigned rights)
- Brightness (For authorized users to Control LCD Brightness)
- CPU Power Management (For authorized users to manage CPU
- CUPS Administration ()

RBAC の動作構成



役割 (Role)

- システムの特殊なアカウントで UID, ホームディレクトリ, パスワードなどを持つ
 - /etc/passwd にエントリを持つ
- ログインして使用は出来ない
 - 役割 (role) は使用を許可されているユーザが、su コマンドにて利用することを前提にしている
 - pam_roles(5) モジュールによりログインを制限
- 通常は特定の作業を意味する
 - データベース管理、ユーザ管理、

承認 (Authorization)

- 承認とはセキュリティに影響を及ぼす可能性のある操作をユーザまたは役割が実行可能にするアクセス権
- 特権はカーネルレベルでセキュリティポリシーを適用するのに対し**承認はアプリケーションレベルにて定義**される
- /etc/security/policy.conf に一般ユーザが CD-ROM デバイスにアクセスを可能にする承認 `solaris.device.cdrw` が定義されている

```
AUTHS_GRANTED=solaris.device.cdrw
```

○ policy.conf

- システム全体のセキュリティ関連のデフォルト設定が定義
- 例：全てのユーザに割り当てられる権利プロファイル
`PROFS_GRANTED=Basic Solaris User`

承認 (Authorization)

○ 承認は `/etc/security/auth_attr` に定義されている

`authname:res1:res2:short_desc:long_desc:attr`

<code>authname:</code>	承認を識別するユニークな任意の文字列。 <code>prefix.[suffix]</code> の形式で指定。 OpenSolaris では <code>prefix</code> に solaris を使用する。 その他については承認を作った組織のインターネットドメイン名を逆さにして利用 (ex. <code>com.xyz</code>)。 <code>suffix</code> には承認する機能や操作を表現する。
<code>res1, res2:</code>	予約フィールド。
<code>short_desc:</code>	承認の短い説明。
<code>long_desc:</code>	承認の目的、アプリケーション、想定するユーザなどを説明。
<code>attr:</code>	<code>key=value</code> 形式にてオプションの属性を指定する。 キーワード <code>help</code> は <code>/usr/lib/help/auths/locale/C/</code> に置かれる HTML 形式のヘルプファイルを指定する。

役割の生成

○ 役割 admin の追加

```
# roleadd -m -d /export/home/admin -P "System Administrator" admin
/etc/passwd
admin:x:6986:1::/export/home/admin:/bin/pfsh
```

○ 役割 admin に承認 solaris.role.* を割り当てる

```
# rolemod -A solaris.role.* admin
/etc/user_attr
admin::::type=role;auths=solaris.role.*;profiles=System Administrator
```

○ ユーザ demo に役割 admin を割り当てる

```
# usermod -R admin demo
/etc/user_attr
demo::::type=normal;roles=admin
# roles demo
admin
```

プロファイルシエル

- 役割を実行するには、プロファイルシエルが必要である
- 役割の権利プロファイルをチェックし、最初に一致するエントリを実行する
- OpenSolaris にて提供されているプロファイルシエル
 - /usr/bin/pfsh
 - /usr/bin/pfcsh
- pfexec
 - 権利プロファイルに定義されるコマンドを実行
 - コマンドはプロファイルシエルによって呼び出される
 - -P オプションにより特権プロファイルでユーザに与えられた追加特権を取得出来る

/etc/user_attr (1)

○ /etc/user_attr にユーザの拡張属性を設定

```
user:qualifier:res1:res2:attr
```

user :	/etc/passwd に設定されたユーザ名または役割名
qualifier, res1, res2:	予約フィールド
attr:	セミコロンで区切られたオプションフィールド。 key=value 形式にてユーザがコマンド実行した時に適用されるセキュリティ属性を指定する。

/etc/user_attr (2)

○ attr フィールドで使用可能な key を以下に示す

- type : **normal** は通常ユーザ、 **role** は役割。
- auths : auth_attr にて定義される承認を指定する。アスタリスク '*' を用いてワイルドカード指定が可能。複数指定する場合はコンマで区切る。
- profiles : prof_attr にて定義される権利プロファイル名を指定する。複数の指定をコンマで区切って指定可能で、UNIX シェルの PATH のように動作する。
- roles : ユーザに割り当てる役割を指定する。複数指定する場合はコンマで区切る。 type=role で設定されているエントリーには使用出来ない。
- defaultpriv : ログイン時の初期特権を割り当てる。
- limitpriv : ユーザの上限特権を定義。
- lock_after_retries : アカウントのロックポリシーを設定。

その他の key については man ページ user_attr(4) を参照

権利プロファイル (Rights Profiles)

- 役割またはユーザに割り当てることが出来る管理権限の集まり
- 権利プロファイルの構成
 - 承認による構成
 - セキュリティ属性を持つコマンドによる構成
 - 他の権利プロファイルによる構成
- 権利プロファイルは次のファイルにて構成
 - `/etc/security/prof_attr`
 - `/etc/security/exec_attr`

/etc/security/prof_attr (1)

○ 権利プロファイルの定義ファイル

```
profname:res1:res2:desc:attr
```

profname : 権利プロファイル名。大文字小文字の区別があることに注意。

res1, res2 : 予約フィールド

desc : 権利プロファイルの説明。

attr : セミコロンにて区切って実行時のセキュリティ属性をオプション指定する。 key=value 形式

/etc/security/prof_attr (2)

○ attr フィールドで使用可能な key を以下に示す

- help : auth_attr と同様の扱い。
- profiles: prof_attr の他の profname を指定する。
- auths : auth_attr に定義される承認を指定する。アスタリスク '*' を用いてワイルドカード指定が可能。
- privs : 役割またはユーザに割り当てる特権 (priv_names(4) を参照) を指定。

profiles, auths, privs の何れも複数の指定する場合は、コンマで区切って指定可能

/etc/security/exec_attr

○ 実行プロファイルの定義ファイル

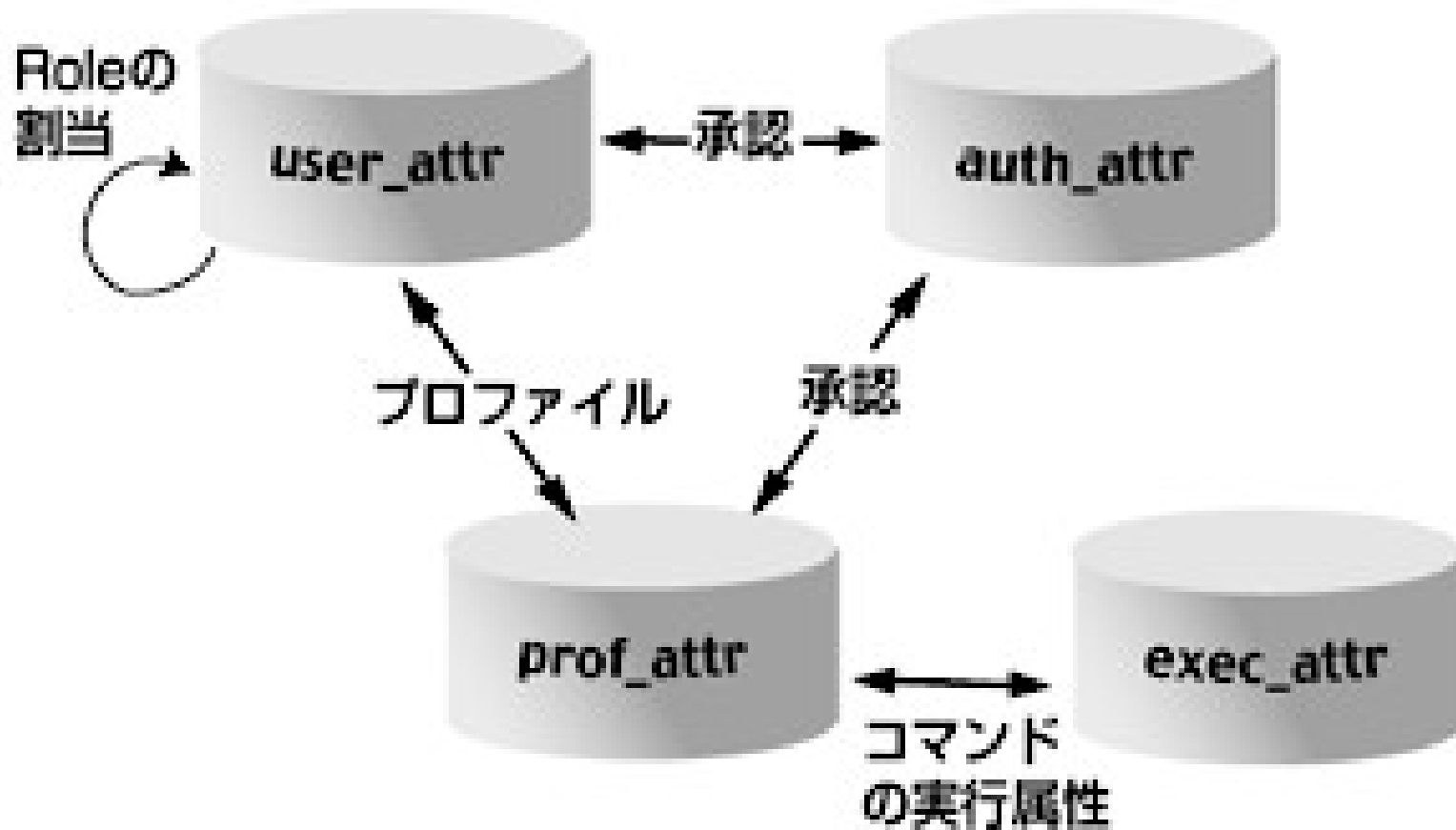
name:policy:type:res1:res2:id:attr

- name : prof_attr に定義される権利プロファイル名
- policy : セキュリティポリシーを指定。 **suser**(standard Solaris superuser) と **solaris** の 2 種類が指定可能。 solaris の場合は特権 (privileges(5)) を認識するが、 suser は特権を認識しない。
- type: **cmd** は id フィールドに実行するコマンドやスクリプトを指定する。 **act** は Trusted Extensions にて使用され、 id フィールドには CDE アクションを指定する。
- res1, res2 : 予約フィールド
- id : 実行するコマンドをフルパスで指定する。コマンドのオプションがある場合には、別にスクリプトを記述して、そのスクリプトを指定する。
- attr : suser の場合、 euid, uid, egid, gid の指定が可能。 solaris の場合、 privs, limitprivs が指定可能。

権利プロファイルとユーザの割り当て

- ユーザ demo に権限プロファイル File System Management を割り当てる
usermod -P "File System Management" demo
- /etc/user_attr に以下のエントリが追加される
demo::::type=normal;profile=File System Management
- 割り当てられたプロファイルの確認
\$ id
uid=6985(demo) gid=1(other) groups=1(other)
\$ profiles
File System Management
SMB Management
VSCAN Management
SMBFS Management
Basic Solaris User
All
- ユーザ demo でもルートディレクトリに新規ディレクトリを作成可能になる
\$ pfexec /usr/bin/mkdir /testdir

RBAC 構成ファイルの関係図



RBAC 構成ファイルの関係

```
[global 0]: $ cat /etc/security/auth_attr
[...]
```

**solaris.admin.usermgr.manage::Manage Users and Roles\
::help=AuthUsermgrManage.html**

```
solaris.admin.usermgr.read::View Users and Roles\  
::help=AuthUsermgrRead.html  
solaris.admin.usermgr.write::Modify Extended Security Attributes  
::help=AuthUsermgrWrite.html
```

```
[global 0]: # more /etc/security/prof_attr
[...]
```

User Management::Manage users, groups, home directory:
auths=**solaris.profmgr.read,solaris.admin.usermgr.write,solaris.admin**
.usermgr.read,solaris.admin.usermgr.manage;help=RtUserMngmnt.html

```
[...]
```

```
[global 0]: # more /etc/user_attr
[...]
```

**root:::type=role;auths=solaris.*;solaris.grant;profiles=All;\
lock_after_retries=no;min_label=admin_low;clearance=admin_high**

```
[...]
```

```
[global 0]: $ cat /etc/security/exec_attr
[...]
```

User Management; solaris:cmd:: /usr/sbin/usermod:euid=0

```
[...]
```

RBAC まとめ

- 管理タスクを行うのにスーパーユーザである必要はなくなった
 - ホストを複数の人が利用する場合に、root パスワードを共有する必要がなくなり、セキュリティリスクが軽減出来る
- /etc/security/prof_attr には様々なプロファイルが既に定義されているので、新規 role を作成するのに便利である



Thank You.

Jan. 2009

Takanobu Masuzuki
OpenSolaris Evangelist

参考文献

- Solaris のシステム管理ドキュメント
 - <http://docs.sun.com/app/docs/doc/819-0383/prbactm-1?l=ja&a=view>
- SDC の記事
 - <http://sdc.sun.co.jp/solaris/solaris10/security.html>
- 各種ファイルおよびコマンドの man ページ
- Sun Tech Days 2008 in Tokyo "Developing & Deploying Securely by 野崎宏明"
 - http://sdc.sun.co.jp/techdays/presentations/2008/PDF/2-S-6_Security.pdf
- OpenSolaris Security Project
 - <http://www.opensolaris.org/os/community/security/projects/rbac/>
- BigAdmin
 - http://www.sun.com/bigadmin/xperts/sessions/16_prm/
- Casper Dik's Blog
 - http://blogs.sun.com/casper/entry/solaris_privileges

RBAC Step by Step (reboot role)

```
$roles
No roles
$/usr/sbin/reboot
reboot: permission denied
# roleadd -m -d /export/home/reboot reboot
64 blocks
# passwd reboot
New Password:
Re-enter new Password:
passwd: password successfully changed for reboot
# grep reboot /etc/passwd
reboot:x:102:1::/export/home/reboot:/bin/pfsh
# usermod -R reboot demo
UX: usermod: demo is currently logged in, some
changes
may not take effect until next login.
# grep "demo" /etc/user_attr
demo:::type=normal;roles=reboot
```

```
# echo "REBOOT:::profile to reboot:help=reboot.html"
>> /etc/security/prof_attr
# rolemod -P REBOOT reboot
# grep reboot /etc/user_attr
reboot:::type=role;profiles=REBOOT
demo:::type=normal;roles=reboot
# echo "REBOOT:suser:cmd:::/usr/sbin/reboot:euid=0"
>> /etc/security/exec_attr
$ roles
reboot
$/usr/sbin/reboot
reboot: permission denied
$ su reboot
Password:
$/usr/sbin/reboot
.....
```

OpenSolaris にて root のログイン

- OpenSolaris の初期設定では root は role として設定されている
 - root ではログイン出来ない
 - インストール時に作成する一般ユーザが root role を使用できるように設定される
- OpenSolaris で root でログインするには
 - root role が割り当てられているユーザから設定を解除
 - GUI "ユーザとグループ" をシステムメニューから起動して、インストール時に作成したユーザアカウントの User Role から root のチェックを外す
 - rolemod コマンドにて root を role から normal に変更


```
# rolemod -K type=normal root
```

Solaris RBAC vs sudo

特徴	Solaris RBAC	Sudo
Authorizations	O	X
PAM	O	O
Cross Platform	X	O
Kerberos Support	O	O
Solaris BSM Audit	O	X
RUID	O	O
EUID	O	X
RGID	O	X
EGID	O	X
Hierarchical Profiles	O	X
Network Wide Policy	O	O
Host Specific Policy	O	O
Netgroup Policy	X	O
Require Password	X	O
Allow no Password	O	O
Cached Authentication	X	O
Restrict Users	O	X
Profile Shells	O	X
Control cmd arguments	X	O
Privileges/Capabilities Aware	O	X
Authenticate as Self	X	O
Control Sensitive Environment Variables	O	O
Control UMASK	X	O
Fine grained Policy Admin	O	X
Default Profiles for OS Admin	O	X