Defining Administrative and User Profiles

Glenn Faden

Secure Software Engineering, Sun Microsystems, Federal, Inc.

ABSTRACT

Trusted Solaris provides a configuration facility that allows administrators to define profiles consisting of specific sets of UNIX commands, CDE actions, and Administration Tools. These profiles include the security attributes to be applied to each command and action. The capabilities of individual users may be defined in terms of these profiles in NIS+ tables.

This mechanism can be used to configure turn-key environments for novice users, or to break down the administration of the enterprise into specific areas of responsibility called roles. Solstice AdminSuite is recast in terms of CDE actions and roles. The CDE environment is modified so that only allowed actions are made available to users and administrative roles. A restricted shell is also provided which enforces these restrictions for UNIX commands.

Introduction

In any enterprise, the computer skills and responsibilities of individuals will vary. It is generally desirable to be able to describe their capabilities in terms of their actual responsibilities within the organization. For example, some users should be allowed to set security attributes, or to maintain certain subsystems, such mail, printing, or file archival. This customizing has been difficult to do in traditional UNIX because it is based on a model of a single superuser, and it lacks the fine grained security attributes which are necessary for specifying and controlling behavior. Trusted Solaris is an extended version of Solaris which provides this fine grained control.

All UNIX systems support an access control policy based on a user's identity and membership in groups, which is commonly referred to as Discretionary Access Control, or *DAC*. Recently some UNIX systems, such as Solaris, have added Access Control Lists (ACLs) which provide additionally flexibility in specifying which files can be accessed or executed by specific users or groups. While DAC controls are very useful in this regard, additional controls are provided by Trusted Solaris:

- a *role mechanism* that allows administrative functions to be broken apart into discrete functions, so that administrative users have only the powers that are necessary and sufficient to perform their jobs
- ~~a *sensitivity based access control system* that is represented by labels, and used to provide access based on content rather than ownership. This mechanism is commonly referred to a Mandatory Access Control or MAC, because normal users have no discretion as to how the labels protect the data~~
- a *privilege mechanism* to replace the traditional UNIX reliance on the User ID zero being the superuser

Copyright © 1995-1997 by Sun Microsystems, Inc. All Righs reserved.

- an *authorization mechanism* to specify the security relevant attributes that a user may affect

In addition, Trusted Solaris provides:

- a graphical user interface for defining and making available these capabilities
- a set of configuration tools that can be used to specify which policies are enforced

This paper we describes how the capabilities of user and roles are specified, and how these security controls are applied to users and their actions.

Separation of Administrative Roles

In Trusted Solaris, administrative tasks are performed using roles, which are assumed by authorized users. Roles are generally associated with specific duties, and have unique responsibilities. The operations performed in a role are restricted so that no unnecessary commands are available, and auditable to provide a record of actions. Naturally, determining what is necessary and sufficient is a subjective decision, so that each role's capabilities need to be tunable. However, no role is allowed to modify its own set of capabilities. Therefore a minium of two roles is required, so that each role can configure the other.

User and Roles

Users and roles are closely related, and therefore easily confused. The term *user* corresponds to a specific person, whereas a *role* corresponds to a specific functional responsibility. Common role names include: Security Officer, System Administrator, Operator, and Software Installer. The precise list for a given site is determined by the operational needs of the site.

Individual roles may be assigned to one or more users. To perform the responsibilities of the role, the user must first perform a specific action known as *assuming a role*. This action is comparable to the UNIX command **su -**, because the user must enter a password, before being granted the powers associated with that role. One of the distinctions from **su** is that the capabilities of the role are limited, and are not equivalent to those of the superuser. On the other hand, if a superuser-like role is desired, it can be configured, and made available using the role configuration mechanism.

Another distinction is that the user interface for roles, including command line and GUI based actions are performed in a private CDE workspace. Doing so protects these actions from interference by untrusted applications.

Since roles correspond to functions rather than individuals, their actions are accountable to the actual user who assumed the role. Since more than one user may be assigned to a particular role, the auditing records associated with the actions of a role contain the audit ID of the original user. Although role names do appear in the password file, and other NIS+ databases, they are not allowed to log in directly. This restriction ensures that their actions remain accountable to individuals.

Copyright © 1995-1997 by Sun Microsystems, Inc. All rights reserved.

Both users and roles are granted specific security attributes, ~~including a restricted label range~~, a set of authorizations, and a set of available commands and actions. Users may also be given the right to assume one or more administrative roles.

~~*Sensitivity Label Ranges*~~

~~Labels are used to restrict access to data based on its sensitivity. A label range has a lower bound or minimum label, and an upper bound or clearance. Each process in Trusted Solaris executes at a single label which it inherits form its parent process. The user interface enables both users and roles the ability to execute processes at any label which is within their label range.~~

~~*Protecting Security Databases*~~

~~Trusted Solaris uses both user-based and sensitivity-based access control to protect data. Most administrative files, such as those found in /etc are assigned an administrative label, admin_low, which is lower than any normal user label. Since writing down to a file whose label is lower than the process' label is disallowed, users cannot modify these databases, even if the are granted discretionary write access. Individual administrative roles may be able to modify such files if they can execute a process at the admin_low label. Access control lists can be used to provide specific write access to those roles that need to modify specific administrative files.~~

Privilege

Trusted Solaris does not grant any special powers to the User ID zero. Instead, it provides a fine-grained set of privileges which correspond to specific policy enforcement conditions. In order to override a system policy, a process must hold the specific privilege for that condition. Normally processes have no privileges associated with them, and therefore cannot override the security policy of the system.

Privileges can be associated with executable files, so that the privileges become effective when the program is executed, or privileges may be inherited from a parent process. Associating privileges with executable files is generally undesirable because it presents some of the same security problems as **setuid root** programs in normal UNIX. Instead, a small set of privileged programs are provided which pass the appropriate privileges to programs that must override normal policies.

Authorizations

Unlike privileges, authorizations are associated with users rather than programs. They specify a set of special capabilities that users have been granted based on their identity. Authorizations are general in nature and do not correspond to specific programs. Rather, they are interpreted by privileged programs to determine if a specific user should be allowed to perform a restricted action. For example, a user may have the authorization ~~to set file labels, or~~ write data to removable media. Without the required authorizations, trusted programs deny such requests.

Copyright © 1995-1997 Sun Micosystems, Inc. All rights reserved.

Since authorizations are associated with users, they follow the user throughout the system. They are maintained, along with other per-user attributes, in a NIS+ table.

Restricted Execution

Another way to define user capabilities is to restrict the set of commands which a user may execute. Although this can be done with either the MAC or DAC mechanisms, these approaches don't scale well, because they require the administrator to modify the attributes of many files, each time a user is added to the system. Furthermore, these attributes might need to be maintained on each machine which the user has access to.

Restricted Shells

An alternative approach is to mandate the use of a restricted shell, such as rksh, which limits the paths that a user may search to find executable files. This is not a very flexible approach, however, because the granularity of the restriction is that of an entire directory.

CDE Actions

An even more restrictive approach is to disallow shells altogether, and provide a GUI interface for all user interaction. With the development of the Common Desktop Environment, and its Actions Database, this approach is quite workable. Actions are the verbs in which CDE functions are described. Most actions have an icon associated with them so that they can be accessed from the File Manager or Application Manager. Other actions are used internally to provide various CDE functions, in an object oriented paradigm.

Defining Actions

Using the CDE Create Action GUI, it is a straightforward process to provide a graphical interface to most commonly used UNIX commands. Such interfaces are easier to understand because they are named in the local language, as well as represented graphically. More and more users of UNIX systems today, are not familiar with UNIX commands, nor should they be. The complexity of UNIX can be hidden behind the graphical metaphors of actions and types. Users see files and commands as graphical entities in the File Manager or the Application Manager. They use localized names, rather than deal with UNIX program names (grep, awk, sed) and complex syntax.

Restricting Actions

The current CDE product has some support for restricting the actions that a user can perform, but it is based on pathnames, rather than specific actions. It is also not particularly restrictive, in that any UNIX executable can be launched by double clicking on it in the File Manager.

Copyright © 1995-1997 by Sun Microsystems, Inc. All rights reserved.

Defining Execution Profiles

A more refined approach is to enumerate the complete list of commands, actions and authorizations which are appropriate for a particular type of user. Such lists are referred to as *execution profiles* in Trusted Solaris. Trusted Solaris provides a graphical user interface for constructing execution profiles from commands, actions, and authorizations, and for associating profiles with users and roles.

Although it is possible to maintain a list of users for each command, as can be done with ACLs, it is far more useful to maintain a list of commands for each user. One difficulty in using ACLs is that they are stored with the files, rather then in a NIS+ database. Furthermore, there is no ACL equivalent for specifying a list of users who can invoke a CDE action. Execution profiles provide a per-user mechanism for associating arbitrary security attributes with the allowed set of commands, actions, and authorizations.

For example, in Trusted Solaris, a discrete set of privileges can be specified for each allowed command for each individual user or role. Thus user joe could be allowed to run **cp**, but with no privileges, but user jane could be allowed to run **cp** with several privileges. ~~In addition, a restrictive label range can be specified so that the command can only be invoked when the user is operating at a label within the range.~~ If the program should be run with a special effective User ID or effective group ID, that can be specified as well.

The Trusted Shell

Roles typically perform a set of tasks which require both command line and graphical interfaces. For commands, roles must use a trusted version of the Bourne shell, **pfsh**, which has been modified to support both the restrictive and enabling aspects of the command list profiles. **pfsh** prevents the execution of commands that are not included in the profile. ~~In addition it ensures that the programs with a restrictive label range are not executed outside of that range.~~ Finally it sets the effective privileges, user ID, and group ID that are specified in the profile. **pfsh** does not allow shell scripts to be executed unless they are listed in the profile, or are themselves **pfsh** scripts.

Although **pfsh** is primarily intended for use by administrative roles, it can also be specified as the shell for normal users. This meets the requirement of sites where it is desirable to use the profile database to limit the commands that a user can invoke. The selection of a user’s default shell is specified by an administrator acting in a role.

Invoking Actions

CDE actions are defined in a database which is constructed from a set of types files. A special search path is used to locate all the types files where actions are defined. The specification of the search path is protected in Trusted Solaris so that users cannot interfere with the definitions of the actions.

The actions database is loaded, and interpreted by a shared DeskTop Services library called **DtSvc**. In Trusted Solaris, a modified version of this library is provided, and used by trusted CDE components, such as the window manager, and the File Manager. The modified library provides the equivalent functionality of the trusted **pfsh**, but

Copyright © 1995-1997 Sun Micosystems, Inc. All rights reserved.

applies the specified security attributes to actions instead of commands. For example, actions that do not appear in the user's profile ~~or are have a restricted label range which does not include the current label,~~ are not available to the user. Actions whose method is a command, and are specified to require special privileges, user IDs, or group IDs, are executed with the required attributes The security attributes are not applicable to actions whose method is a Tool Talk message.

In keeping with the concept of restricting execution, normal users cannot create their own actions, nor can they execute programs via the File Manager, except through the action database. The action Execute, which is the method for running arbitrary programs, is not normally allowed.

Solstice AdminSuite

The Solstice AdminSuite is extended in Trusted Solaris to provide additional GUIs for its unique security attributes. While AdminSuite provides an extension framework for adding new administrative tools, it does not provide a convenient means for customizing a distinct tool set for separate administrative roles. Each member of the sysadmin group is presented with the same set of tools when AdminSuite is launched.

Trusted Solaris recasts the individual admintools into CDE actions. By wrapping the tools in actions Trusted Solaris is able to provide a customized set of tools to each administrative role, and to provide unique security attributes for these actions. The actual AdminSuite launcher is replaced by a folder view in the CDE Application Manager.

Configurability

Such configurability is not without costs. A great deal of analysis must be done to determine which commands and actions should be associated with each profile. For each command and action, there may be a need to specify one or more privileges~~, or a restricted label range~~. It is also possible to specify whether a special user ID or group ID should be used when executing the command or invoking the action.

Predefined Execution Profiles

Rather than expose this complexity to the customer, Trusted Solaris comes configured with a set of execution profiles for tasks such as installing packages, adding users, performing backups, or configuring devices.

User profiles delivered with the system include one for novices who don't know UNIX, one with all commands and actions.

Associating Execution Profiles with Users

The Solstice User Manager is extended in Trusted Solaris to provide an interface for assigning execution profiles to users. Multiple profiles can be assigned to a single user; this has the effect of combining multiple profiles into larger sets.

Copyright © 1995-1997 by Sun Microsystems, Inc. All rights reserved.

The Profile Manager

Execution profiles are defined using a new Profile Manager. The Profile Manager supports drag and drop from the File Manager or Application Manager to construct lists of allowed commands and actions. The Profile Manager also provides scrollable lists of potential command and actions, and dialog windows for setting specific security attributes for each allowed command or action.

Copyright © 1995-1997 Sun Micosystems, Inc. All rights reserved.

The Profile Manager has two interfaces, one for actions and one for commands. The action view provides a description of each action, both in English and in terms of UNIX commands. A typical screen is shown below.

Figure 1 Profile Manager - Action View

Copyright © 1995-1997 by Sun Microsystems, Inc. All rights reserved.

The command view displays the executable files in a specified search path from which a profile can be constructed. Unique security attributes can be associated with each allowed command.

Figure 2 Profile Manager - Command View

Profiles Are Not for Everyone

Since the restricted environment may not be appropriate for some individuals and sites, the administrator may determine when it should be applied. Users are only affected by execution profiles if their default shell is specified as tsh. Other UNIX shells do not use the profile database. However, the CDE action database can still be defined in terms of profiles even if the restricted tsh is not used. A special authorization can be associated with users to allow them to provide their own set of actions.

Copyright © 1995-1997 Sun Micosystems, Inc. All rights reserved.

Conclusion

Although the profile mechanism was designed to meet the US federal government requirements for compartmented mode workstations[1], it is an extensible mechanism. Enterprises with large sets of users frequently need to provide a turnkey environment for some sets of users, or multiple sets of turnkey environments. The profile design provides a powerful but easy to use tool for constructing these environments.

Sites with more than one administrator can use the profile mechanism to distribute various administrative tasks to different set of administrators. Tasks can be delegated and controlled based on the functional responsibilities.

References

[1] Compartmented Mode Workstation Evaluation Criteria. Defense Intelligence Agency, DDS-2600-6249-91. November 1991.

Copyright © 1995-1997 by Sun Microsystems, Inc. All rights reserved.