J. Comuzzi (DEC)
February 1994

ADDING GROUP OVERRIDE SUPPORT TO DCE SECURITY

1. INTRODUCTION

The DCE 1.0 security facility permits UNIX hosts to override account information from the security registry on a per host basis. This RFC provides a proposal to extend this facility in DCE 1.1 to support overriding group information and adding a mechanism to lookup account and group information using the overridden UID or GID as a key.

1.1. Statement of the Problem

DCE 1.0 security includes a password override facility, which uses the password override file. This file, which exists in `/opt/`*dcelocal*`/etc/passwd_override`, has a syntax similar to the syntax of the `/etc/passwd` file which exists on most UNIX systems. `/opt/`*dcelocal*`/etc/passwd_override` permits using the DCE registry (`secd`) as the repository for local account information, but allows specific information to be overridden on the local system (a ''customizibility'' feature). This is particularly useful because without it all UNIX hosts would have to share the same conventions on pre-existing system accounts (i.e., pre-DCE, low UID), which would be inconvenient since there are many different, incompatible, conventions for the names and UID's of such pre-existing accounts.

No analogous mechanism currently exists to reconcile differences between the pre-existing groups in the registry and pre-existing groups that might be present on a given UNIX system. Further, no mechanism exists to access registry information starting with the overridden UID or GID as a key.

1.2. Outline of the Proposal

This proposal adds a file, `/opt/`*dcelocal*`/etc/group_override`, and adds a new API routine, `sec_rgy_pgo_get_by_effective_unix_num()`. This routine is the analogue of `sec_rgy_pgo_get_by_unix_num()`. It returns account or group information that use UID's and GID's that are subject to override information in `/opt/`*dcelocal*`/etc/passwd_override` and `/opt/`*dcelocal*`/etc/group_override` files. The latter file would permit the specification of the name to GID association specific to this system.

2. DETAILED PROPOSAL

The `sec_rgy_pgo_get_by_unix_num()` API routine is implemented as an RPC to the security registry, `secd`. However the routine `sec_rgy_pgo_get_by_effective_unix_num()` would be implemented as a local routine, which calls a new override interface, followed by an RPC to the registry server. The new override routine would be part of the local machine's `sec_clientd`, where a database of account override information is already maintained in memory. An additional database maintained as a linked list of `struct override_group_buffer`'s would be added.

2.1. Addition of Group Override Information to sec_clientd

All processing of the `passwd_override` information is localized in the file `src/security/sec_clientd/roverride.c`. The data structure maintained and accessed therein (the list of `struct override_buffer`'s) would be augmented by an additional list of `struct override_group_buffer`'s and `struct override_group_member`'s. See Appendix A for details.

The private routine `check_update_group_overrides()` would be added to reinitialize the `override_group_buffer` lists if the `group_override` file changed. This is exactly analogous to `check_update_overrides()` maintenance of the override_buffer's list. Similarly, a new routine `parse_group_override_buffer()` would be added to parse the syntax of the `group_override` file — see Appendix B for details on this syntax.

Four new routines (`roverride_get_by_unix_num()`, `roverride_get_group_info()`, `roverride_check_group_passwd()` and `roverride_is_group_passwd_overridden()`) would be added. These routines would be added as additional routines to the IDL file `roverride.idl`. Since they are being added at the end, only the interface minor version number need be incremented. Indeed, since this IDL interface is *only* used to communicate among the DCE library, `dce_login` and `passwd_export` and `sec_clientd` on *one* machine, all of which would be replaced in any upgrade to DCE 1.1, it would not matter if the interface major version number was changed. [Note that there is no conceivable reason for calling this interface remotely, since the outputs from the existing three routines are only interpretable on the local host. However, if changing the interface version is perceived as a problem, the three routines could be exported via a new IDL interface (say `rgroupoverride.idl`) at the cost of one additional entry in the end-point mapper. Again, see Appendix A for details on the API of these routines.]

The routine `roverride_get_by_unix_num()`, would implement a case on the `name_domain`. For the domain `sec_rgy_domain_person`, the `override_buffer` list would be searched for an entry with matching UID. If one is found, the function returns true, with the name of the corresponding account; otherwise it returns false. For the domain `sec_rgy_domain_group`, the `override_group_buffer` list would be searched for an entry with matching GID. If one is found, the function returns true, with the name of the corresponding group; otherwise it returns false. For the domain `sec_rgy_domain_org`, the fuction returns false. In both the searches, only entries with keyfield's of name (i.e, either principal name or group name) are examined. If the name field is not present in an override entry, then the correct information can be obtained from the registry.

The routines `roverride_get_group_info()`, `roverride_check_group_passwd()` and `roverride_is_group_passwd_overridden()` would be almost identical to the existing corresponding routines, except that the `override_group_buffer` list would be searched.

Finally, in addition to the changes required in `src/security/server/sec_cliend/roverride.c`, the above entry points would have to be added in `src/security/client/rca/override.c` and `override.acf`. These entry points are a simple pass-thru's of the arguments.

2.2. Addition of sec_rgy_pgo_get_by_effective_unix_num()

Implementation of this `libdce` routine would parallel `sec_rgy_login_get_effective()`. It would consist of a call to `roverride_get_by_unix_num()` to obtain the name of the account or group if it is overridden. If `roverride_get_by_unix_num()` indicates this UID or GID has been overridden, the returned name is used in a call to `sec_rgy_pgo_get_by_name()`. If it has not been overridden, `sec_rgy_pgo_get_by_unix_num()` is called. The signature of `sec_rgy_pgo_get_by_effective_unix_num()` is identical to `sec_rgy_pgo_get_by_unix_num()`, and is listed in Appendix A.

2.3. Addition of Group Override Mechanism to passwd_export

`passwd_export` currently supports using information in `/opt/`*`dcelocal`*`/etc/passwd_override` in its generation of `/etc/passwd`. It will need a several line modification to access `roverride_get_group_info()`, so that information in `/opt/`*`dcelocal`*`/etc/group_override` will override the registry in its generation of `/etc/group`.

2.4. Documentation

There are three documentation tasks associated with this proposal. First, the new public API routine `sec_rgy_pgo_get_by_effective_unix_num()` would need to be documented. This would be done by cloning the `sec_rgy_pgo_get_by_unix_num()` documentation and making minor changes. Second, the format of the `group_override` file would need to be documented. This has been included in Appendix B. Third, the man page for `passwd_export` would need to be updated to reflect the fact that unless the `-n` switch is present, both the `passwd_override` and `group_override` files will effect the created files.

Note that the four new `roverride` routines would be given the same documentation treatment as the existing routines in `roverride.idl`; currently, those are not documented in the DCE, but they may be documented in the future.

3. OTHER NOTES

Noticeably absent in the above proposal is any additional implementation to support ‘‘organizations’’. This is because organizations are not a UNIX concept, and are not required to achieve a clean integration with UNIX.

There are several possible other API routines that could be added. These include `sec_rgy_pgo_effective_unix_num_to_id()` and `sec_rgy_pgo_effective_unix_num_to_name()`. It is not proposed to add these additional API routines, since their functionality can be obtained with `sec_rgy_pgo_get_by_effective_unix_num()`.

APPENDIX A. DATA STRUCTURES AND ROUTINE SIGNATURES

A.1. struct override_group_buffer

```
typedef unsigned32                sec_group_override_fields_t;
const sec_group_override_fields_t sec_group_override_none
                                                      = 0x0;
const sec_group_override_fields_t sec_group_override_gr_gid
                                                      = 0x1;
const sec_group_override_fields_t sec_group_override_gr_passwd
                                                      = 0x2;
const sec_group_override_fields_t sec_group_override_member
                                                      = 0x4;

typedef struct override_group_member {
    sec_rgy_name_t                  member;
    struct override_group_member    *next;
} override_group_member_t;

typedef struct override_group_buffer {
    sec_rgy_name_t                  gr_name;
    sec_rgy_unix_passwd_buf_t       gr_passwd;
    signed32                        gr_gid;
    struct override_group_member    *members;
    sec_group_override_fields_t     overridden;
    sec_override_key_t              key_fields;
    struct override_group_buffer    *next;
} override_group_buffer_t;
```

A.2. Signature of roverride_get_by_unix_num()

```
boolean32
roverride_get_by_unix_num (
    [in]            handle_t                    h,
    [in]            sec_rgy_domain_t            name_domain,
    [in]            signed32                    unix_id,
    [out, string]   char                        *name,
    [out]           error_status_t              *status );
```

A.3. Signature of roverride_get_group_info()

```
void
roverride_get_group_info (
    [in]            handle_t                    h,
    [in, string]    char                        *gr_name,
    [in, out]       signed32                    *gr_gid,
    [out]           sec_rgy_unix_passwd_buf_t  gr_passwd,
    [in]            signed32                    max_members,
    [in, out]       sec_rgy_cursor_t            *member_cursor,
    [out, v1_array, length_is(*number_supplied),
     size_is(max_members)]
                    sec_rgy_member_t            member_list[],
    [out]           signed32                    number_supplied,
    [out]           signed32                    number_members,
    [out]           sec_group_override_fields_t *overridden,
    [out]           error_status_t              *status );
```

A.4. Signature of roverride_check_group_passwd()

```
boolean32
roverride_check_group_passwd (
    [in]            handle_t                    h,
    [in, string]    char                        *gr_name,
    [in]            signed32                    *gr_gid,
    [in]            sec_rgy_unix_passwd_buf_t  gr_passwd,
    [out]           error_status_t              *status );
```

A.5. Signature of roverride_is_group_passwd_overridden()

```
boolean32
roverride_is_group_passwd_overridden (
    [in]            handle_t                    h,
    [in, string]    char                        *gr_name,
    [out]           signed32                    *gr_gid,
    [out]           sec_rgy_unix_passwd_buf_t  gr_salt,
    [out]           error_status_t              *status );
```

A.6. Signature of sec_rgy_pgo_get_by_effective_unix_num()

```
[idempotent] void
sec_rgy_pgo_get_by_effective_unix_num (
    [in]            sec_rgy_handle_t            context,
    [in]            sec_rgy_domain_t            name_domain,
    [in]            sec_rgy_name_t              scope,
    [in]            signed32                    unix_id,
    [in]            boolean32                   allow_aliases,
    [in, out]       sec_rgy_cursor_t            *item_cursor,
    [out]           sec_rgy_pgo_item_t          *pgo_item,
    [out]           sec_rgy_name_t              name,
    [out]           boolean32                   overriden,
    [out]           error_status_t              *status );
```

APPENDIX B. FORMAT OF /opt/dcelocal/etc/group_override FILE

The information in this section is intended as the man page for the `group_override` file.

Purpose:

The registry database group override file.

Synopsis:

`/opt/`*`dcelocal`*`/etc/group_override`

Description:

The `/opt/`*`dcelocal`*`/etc/group_override` administrative file lets you override the UNIX group ID for a group similar to manner in which the `passwd_override` file permits overriding information in the network registry database.

The `group_override` file is stored on each machine. Any changes you make to it are in effect for the local machine only, and have no effect on the centralized registry. You may find `group_override` especially useful overriding the default group definitions supplied with the registry if they do not match your local UNIX system.

The `group_override` File Format:

The format of the `group_override` entries is similar to entries in the UNIX `group` file. The format is:

`group_name``:`*`passwd`*`:`*`group_uid`*`:`*`members`*

In the override entry, *`group_name`* and *`group_uid`* are keyfields. You must enter one to identify the group to which the override applies. The keyfield is used to perform a lookup in the override file. The lookup is performed in order as the entries are specified in an override entry: first by group name, then by group UNIX ID. If you specify more than one keyfield in an override entry, the first keyfield specified is used as the lookup key; subsequent keyfields are used as overrides.

Field Descriptions:

`group_name`

A keyfield that contains the name that identifies the group to which the override applies.

`passwd`

This field specifies the encrypted password. If you specify an override in this field, the password you enter is in effect for this local machine only.

You can also specify `OMIT` in the passwd field to disallow `newgrp`'s to this group on the local machine. The use of `OMIT` in conjunction with an option to the `passwd_export` command also prevents the inclusion of this group in the group file created by `passwd_export`. (See the section entitled ''Using `OMIT`'', later in this command reference, for details.)

`group_uid`

A UNIX group ID. This filed can function as a keyfield, when no other keyfields are entered, or as a field containing an override, when entered in conjunction with group name. This field specifies the local override of the group ID supplied by the network registry server.

`members`

This field specifies a comma-separated list of members of the group. The contents of this field will override information in the registry when `passwd_export` creates an `/etc/group` file. Note that to specify a null membership, as opposed to indicating no override is required, use an asterisk (`*`) for this field.

Leaving Fields Blank:

If you do not want to override an item, leave its field blank, separating each blank field with a colon (`:`). Note that to override a group with a null membership list, enter an asterisk (`*`) for the *`members`* field.

Using `OMIT`:

If you enter either the word `OMIT` or another invalid password string (such as an asterisk or `NO GOOD`) in the `passwd` field, users will not be able to issue a `newgrp` to this group on the local machine. If you specify `OMIT` and run `passwd_export` with the `-x` option, the named group will not appear in the

`/etc/group` file produced by `passwd_export`.

You should also be aware that, if you have omitted groups from the `/etc/group` file, information about those groups will not be available to any programs that use the group file. For example, the `ls -lg` command accesses the group file to obtain further information about a group. If the group is omitted, no group entry will exist and no information will be available. For this reason you should use `OMIT` to omit groups from the `/etc/group` file only if your user community is very large and either of the following conditions occur:

(i) The group file is taking up too much space.

(ii) Group-ID-to-name mapping is too slow (during `ls -lg`, for example).

Examples:

(a) To override the group ID of group kmem to be 3 use the entry:

```
kmem::3:
```

(b) To override the group password and membership for group system to the single account root, use the entry:

```
system:*::root
```

AUTHOR'S ADDRESS

Joe Comuzzi
Digital Equipment Corporation
550 King Street, LKG2-2/Z7
Littleton, MA 01460
USA

Internet email: Comuzzi@tuxedo.enet.dec.com
Telephone: +1-508-486-7695