Setuid Interface Formalization: POSIX.1-2008 (2013)

Mark S. Dittmer

November 2, 2014

The tables below describe setuid function standards, a representation in quantifier-free first-order logic, and an associated semantics. The standard texts are from POSIX 1003.1 Base Specification Issue 7 [2]. Some input argument names are changed for clarity, and formula numbers are referenced in the standard text (in parentheses) to indicate the passage(s) from which a particular formula is derived. The following assumptions, which may not be self-evident in the text, are made:

1. Function success entails both a return value of 0 and a change in user process user IDs consistent with the "DESCRIPTION" section of the function standard.

2. Function failure entails not only a return value of -1 and appropriate setting of `errno`, but also no change in the state of the process user IDs.

3. The term "any value", when applied to a user ID, is interpreted to mean "any valid user ID value".

Table A.1: Semantic definitions of syntactic elements in logic formulas.

Syntactic Element(s)	**Element Type**	**Semantics**
0	Integer constant	The number 0; either the root/superuser UID or a function return vaule that indicates success (depending on the context).
-1	Integer constant	The number -1; a function return value that indicates failure or a "do not change UID" input argument value.
arg_uid	Integer Variable (Int. Var.)	The only input argument to `setuid()` and `seteuid()`.
arg_ruid	Int. Var.	The first input argument to `setreuid()`.
arg_euid	Int. Var.	The second input argument to `setreuid()`.
old_ruid	Int. Var.	The real UID of the process before function invocation.
old_euid	Int. Var.	The effective UID of the process before function invocation.
old_svuid	Int. Var.	The saved UID of the process before function invocation.

Continued on next page

Table A.1 – continued from previous page

Syntactic Element(s)	**Element Type**	**Semantics**
new_ruid	Int. Var.	The real UID of the process directly following function invocation.
new_euid	Int. Var.	The effective UID of the process directly following function invocation.
new_svuid	Int. Var.	The saved UID of the process directly following function invocation.
rtn	Int. Var.	The return value of the invoked function.
AP	Boolean Variable (Bool. Var.)	An implementation-dependent parameter. True if and only if the process has *appropriate privileges*.
success	Bool. Var.	Indicates whether or not the function succeeded with no errors.
fail	Bool. Var.	Indicates whether or not the function failed due to an error.
einval	Bool. Var.	Indicates that the `EINVAL` error occurred.
eperm	Bool. Var.	Indicates that the `EPERM` error occurred.
ruid_success	Bool. Var.	Indicates correct behavior of the real UID for `setreuid()` function success.
euid_success	Bool. Var.	Indicates correct behavior of the effective UID for `setreuid()` function success.
svuid_success	Bool. Var.	Indicates correct behavior of the saved UID for `setreuid()` function success.
arg_euid_success	Bool. Var.	Relates *arg_euid_is_valid* (see below) with *AP* and the case that the *euid* argument is -1.
arg_euid_is_valid	Bool. Var.	Indicates that the *arg_euid* argument passed to `setreuid()` is permissible for a process without appropriate privileges.
arg_euid_is_invalid	Bool. Var.	Indicates that the *arg_euid* argument passed to `setreuid()` is not permissible for a process without appropriate privileges.
einval	Bool. Var.	Indicates that an `EINVAL` error occurs in a call to `setreuid()`.

Continued on next page

Table A.1 – continued from previous page

Syntactic Element(s)	**Element Type**	**Semantics**
eperm	Bool. Var.	Indicates that an `EPERM` error occurs in a call to `setreuid()`.
$= \neq$	Binary Predicates	True if and only if the value on the left and right are equal, or not equal (respectively).
$RuidIsPermitted(.)$	Unary Predicate	True if and only if its integer argument is a real UID that is "permitted by the implementation".
$IsUID(.)$	Unary Predicate	True if and only if its integer argument is a valid UID in the system.
$\neg \wedge \vee \rightarrow \leftrightarrow$	Boolean Connectives	The Boolean connectives: *negation*, *and*, *or*, *impliciation*, *equivalence* (respectively).

Table A.2: Formulas common among `setuid()`, `seteuid()`,`setreuid()`, and `setresuid()`.

Logic Expressions	
$(success \ \wedge \ \neg fail) \ \vee \ (fail \ \wedge \ \neg success)$	(A.1)
$fail \leftrightarrow \ \ rtn = -1 \ \wedge \ new\_ruid = old\_ruid \wedge$ $new\_euid = old\_euid \ \wedge \ new\_svuid = old\_svuid$	(A.2)
$fail \leftrightarrow \ \ einval \ \vee \ eperm$	(A.3)
$IsUID(old\_ruid) \ \wedge \ IsUID(old\_euid) \ \wedge \ IsUID(old\_svuid)$	(A.4)
$IsUID(new\_ruid) \ \wedge \ IsUID(new\_euid) \ \wedge \ IsUID(new\_svuid)$	(A.5)

Table A.3: Function definition and formulas for `setuid()`.

Annotated Standard Text / Logic Expressions

setuid(arg_uid)

DESCRIPTION

If the process has appropriate privileges, *setuid()* shall set the real user ID, effective user ID, and the saved set-user-ID of the calling process to *arg_uid* (A.7, A.9).

If the process does not have appropriate privileges, but *arg_uid* is equal to the real user ID or the saved set-user-ID, *setuid()* shall set the effective user ID to *arg_uid*; the real user ID and saved set-user-ID shall remain unchanged (A.8, A.10).

RETURN VALUE

Upon successful completion, 0 shall be returned. Otherwise, -1 shall be returned and *errno* set to indicate the error (A.1, A.2, A.6).

ERRORS

The *setuid()* function shall fail if:

[EINVAL] The value of the *arg_uid* argument is invalid and not supported by the implementation (A.7, A.8, A.3, A.12).

[EPERM] The process does not have appropriate privileges and *arg_uid* does not match the real user ID or the saved set-user-ID (A.3, A.7, A.8, A.13).

$$success \leftrightarrow \quad (success\_ap \ \vee \ success\_nap) \tag{A.6}$$

$$AP \rightarrow \quad success\_ap \ \vee \ fail \tag{A.7}$$

$$\neg AP \rightarrow \quad success\_nap \ \vee \ fail \tag{A.8}$$

$$success\_ap \leftrightarrow \quad rtn = 0 \ \wedge \ new\_ruid = arg\_uid \ \wedge \ new\_euid = arg\_uid \ \wedge \ new\_svuid = arg\_uid \tag{A.9}$$

$$success\_nap \leftrightarrow \quad rtn = 0 \ \wedge \ new\_ruid = old\_ruid \ \wedge \ new\_euid = arg\_uid \ \wedge \ new\_svuid = old\_svuid \tag{A.10}$$

$$success\_nap \leftarrow \quad \neg AP \ \wedge \ (old\_ruid = arg\_uid \ \vee \ old\_svuid = arg\_uid) \tag{A.11}$$

Continued on next page

Table A.3 – continued from previous page

Annotated Standard Text / Logic Expressions
$einval \leftrightarrow \ \neg IsUID(arg\_uid)$ (A.12)
$eperm \leftrightarrow \ \neg AP \ \wedge \ arg\_uid \neq old\_ruid \ \wedge \ arg\_uid \neq old\_svuid$ (A.13)

Table A.4: Function definition and formulas for `seteuid()`.

Annotated Standard Text / Logic Expressions
seteuid(arg_uid) **DESCRIPTION** If *arg_uid* is equal to the real user ID or the saved set-user-ID, or if the process has appropriate privileges, seteuid() shall set the effective user ID of the calling process to *arg_uid*; the real user ID and saved set-user-ID shall remain unchanged (A.14, A.15). The seteuid() function shall not affect the supplementary group list in any way. **RETURN VALUE** Upon successful completion, 0 shall be returned; otherwise, -1 shall be returned and errno set to indicate the error (A.1, A.2, A.14). **ERRORS** The *seteuid()* function shall fail if: [EINVAL] The value of the *arg_uid* argument is invalid and not supported by the implementation (A.3, A.16). [EPERM] The process does not have appropriate privileges and *arg_uid* does not match the real user ID or the saved set-user-ID (A.3, A.17).
$success \leftrightarrow \ rtn = 0 \ \wedge \ new\_ruid = old\_ruid \wedge \ new\_euid = arg\_uid \ \wedge \ new\_svuid = old\_svuid$ (A.14)
$success \leftarrow \ arg\_uid = old\_ruid \ \vee \ arg\_uid = old\_svuid \ \vee \ AP$ (A.15)
$einval \leftrightarrow \ \neg IsUID(arg\_uid)$ (A.16)

Continued on next page

Table A.4 – continued from previous page

Annotated Standard Text / Logic Expressions
$eperm \leftrightarrow \neg AP \ \wedge \ arg\_uid \neq old\_ruid \ \wedge \ arg\_uid \neq old\_svuid$ (A.17)

Table A.5: Function definition and formulas for `setreuid()`.

Annotated Standard Text / Logic Expressions
setreuid(arg_ruid, arg_euid) **DESCRIPTION** The *setreuid()* function shall set the real and effective user IDs of the current process to the values specified by the *arg_ruid* and *arg_euid* arguments. If *arg_ruid* or *arg_euid* is -1, the corresponding effective or real user ID of the current process shall be left unchanged (A.19, A.20). A process with appropriate privileges can set either ID to any value (A.27). An unprivileged process can only set the effective user ID if the *arg_euid* argument is equal to either the real, effective, or saved user ID of the process (A.22, A.23). If the real user ID is being set (*arg_ruid* is not -1), or the effective user ID is being set to a value not equal to the real user ID, then the saved set-user-ID of the current process shall be set equal to the new effective user ID (A.21). It is unspecified whether a process without appropriate privileges is permitted to change the real user ID to match the current effective user ID or saved set-user-ID of the process. **RETURN VALUE** Upon successful completion, 0 shall be returned (A.18). Otherwise, -1 shall be returned and errno set to indicate the error (A.1, A.2).
Continued on next page

Table A.5 – continued from previous page

Annotated Standard Text / Logic Expressions

ERRORS

The *setreuid()* function shall fail if:

[EINVAL] The value of the *arg_ruid* or *arg_euid* argument is invalid or out-of-range (A.3, A.25).

[EPERM] The current process does not have appropriate privileges, and either an attempt was made to change the effective user ID to a value other than the real user ID or the saved set-user-ID or an attempt was made to change the real user ID to a value not permitted by the implementation (A.3, A.24, A.26).

$$success \leftrightarrow \; rtn = 0 \; \wedge \; ruid\_success \; \wedge \; euid\_success \; \wedge \; svuid\_success \; \wedge \; arg\_euid\_success \tag{A.18}$$

$$ruid\_success \leftrightarrow \; (arg\_ruid = -1 \; \wedge \; new\_ruid = old\_ruid) \; \vee (arg\_ruid \neq -1 \; \wedge \; new\_ruid = arg\_ruid) \tag{A.19}$$

$$euid\_success \leftrightarrow \; (arg\_euid = -1 \; \wedge \; new\_euid = old\_euid) \; \vee \; (arg\_euid \neq -1 \; \wedge \; new\_euid = arg\_euid) \tag{A.20}$$

$$svuid\_success \leftrightarrow \; (new\_svuid = arg\_euid) \; \leftarrow \; (arg\_ruid \neq -1 \; \vee \; (arg\_euid \neq -1 \; \wedge \; arg\_euid \neq old\_ruid)) \tag{A.21}$$

$$arg\_euid\_success \leftrightarrow \; arg\_euid\_is\_valid \; \leftarrow \; (\neg AP \; \wedge \; arg\_euid \neq -1) \tag{A.22}$$

$$arg\_euid\_is\_valid \leftrightarrow \; arg\_euid = old\_ruid \; \vee \; arg\_euid = old\_euid \; \vee \; arg\_euid = old\_svuid \tag{A.23}$$

$$arg\_euid\_is\_invalid \leftrightarrow \; \neg(arg\_euid = -1 \; \vee \; arg\_euid = old\_ruid \; \vee \; arg\_euid = old\_svuid) \tag{A.24}$$

$$einval \leftrightarrow \; \neg(\; (arg\_ruid = -1 \; \vee \; IsUID(arg\_ruid)) \wedge (arg\_euid = -1 \; \vee \; IsUID(arg\_euid))) \tag{A.25}$$

Continued on next page

Table A.5 – continued from previous page

Annotated Standard Text / Logic Expressions
$eperm \leftrightarrow \neg AP \wedge (arg\_euid\_is\_invalid \vee \neg RuidIsPermitted(arg\_ruid))$ (A.26)
$AP \rightarrow success \vee \neg IsUID(arg\_ruid) \vee \neg IsUID(arg\_euid)$ (A.27)

Table A.6: Function definition and formulas for `setresuid()`. The function definition is the consensus standard, derived from three platform-specific `setresuid()` manual pages [1, 3, 4]. Ellipses indicate areas where platform-specific details appear in manual pages.

Annotated Standard Text / Logic Expressions
setresuid(arg_ruid, arg_euid, arg_svuid)
DESCRIPTION setresuid() sets the real user ID, the effective user ID, and the saved set-user-ID of the calling process. Unprivileged user processes may change the real UID, effective UID, and saved set-user-ID, each to one of: the current real UID, the current effective UID or the current saved set- user-ID (A.33, A.34, A.35, A.36, A.37). Privileged processes [...] may set the real UID, effective UID, and saved set-user-ID to any value (A.32). If one of the arguments equals -1, the corresponding value is not changed (A.29, A.30, A.31). [...]
RETURN VALUE On success, zero is returned (A.28). On error, -1 is returned, and errno is set appropriately (A.38, A.39).
Continued on next page

Table A.6 – continued from previous page

Annotated Standard Text / Logic Expressions
ERRORS [...] [EINVAL] The value of the *arg_ruid*, *arg_euid*, or *arg_svuid* argument is invalid and not supported by the implementation (A.3, A.38). [EPERM] The calling process is not privileged and tried to change the IDs to values that are not permitted.

$$success \leftrightarrow \quad rtn = 0 \ \wedge \ ruid\_success \ \wedge \ euid\_success \ \wedge \ svuid\_success \tag{A.28}$$

$$ruid\_success \leftrightarrow \quad (arg\_ruid = -1 \ \wedge \ new\_ruid = old\_ruid) \ \vee (arg\_ruid \neq -1 \ \wedge \ new\_ruid = arg\_ruid) \tag{A.29}$$

$$euid\_success \leftrightarrow \quad (arg\_euid = -1 \ \wedge \ new\_euid = old\_euid) \ \vee \ (arg\_euid \neq -1 \ \wedge \ new\_euid = arg\_euid) \tag{A.30}$$

$$svuid\_success \leftrightarrow \quad (arg\_svuid = -1 \ \wedge \ new\_svuid = old\_svuid) \ \vee \ (arg\_svuid \neq -1 \ \wedge \ new\_svuid = arg\_svuid) \tag{A.31}$$

$$AP \rightarrow \quad success \ \vee \ (arg\_ruid \neq -1 \ \wedge \ \neg IsUID(arg\_ruid)) \ \vee \ (arg\_euid \neq -1 \ \wedge \ \neg IsUID(arg\_euid)) \ \vee \ (arg\_svuid \neq -1 \ \wedge \ \neg IsUID(arg\_svuid)) \tag{A.32}$$

$$arg\_ruid\_is\_valid \leftrightarrow \quad arg\_ruid = old\_ruid \ \vee \ arg\_ruid = old\_euid \ \vee \ arg\_ruid = old\_svuid \tag{A.33}$$

$$arg\_euid\_is\_valid \leftrightarrow \quad arg\_euid = old\_ruid \ \vee \ arg\_euid = old\_euid \ \vee \ arg\_euid = old\_svuid \tag{A.34}$$

Continued on next page

Table A.6 – continued from previous page

Annotated Standard Text / Logic Expressions	
$arg\_svuid\_is\_valid \leftrightarrow \quad arg\_svuid = old\_ruid \vee$ $arg\_svuid = old\_euid \vee$ $arg\_svuid = old\_svuid$	(A.35)
$new\_uids\_are\_valid \;\leftrightarrow\; arg\_ruid\_is\_valid \wedge$ $arg\_euid\_is\_valid \;\wedge\; arg\_svuid\_is\_valid$	(A.36)
$success \leftarrow \quad \neg AP \;\wedge\; new\_uids\_are\_valid$	(A.37)
$einval \leftrightarrow \neg(\quad (arg\_ruid = -1 \;\vee\; IsUID(arg\_ruid)) \wedge$ $(arg\_euid = -1 \;\vee\; IsUID(arg\_euid)) \wedge$ $(arg\_svuid = -1 \;\vee\; IsUID(arg\_svuid)))$	(A.38)
$eperm \leftrightarrow \quad \neg AP \;\wedge\; \neg new\_uids\_are\_valid$	(A.39)

References

[1] getresgid, getresuid, setresgid, setresuid – get or set real, effective and saved user or group ID. FreeBSD System Calls Manual, April 2001.

[2] IEEE and The Open Group. POSIX.1-2008, 2013. Available from http://pubs.opengroup.org/onlinepubs/9699919799/.

[3] setresuid, setresgid - set real, effective and saved user or group ID. Linux Programmer's Manual, July 2007.

[4] getresgid, getresuid, setresgid, setresuid - get or set real, effective and saved user or group ID. OpenBSD Programmer's Manual, August 2013.