Full credit is given to the above companies including the OS that this PDF file was generated!

Red Hat Enterprise Linux Release 9.2 Manual Pages on 'sepgsql_contexts.5' command

$ man sepgsql_contexts.5

selabel_db(5) SELinux API documentation selabel_db(5)

NAME

selabel_db - userspace SELinux labeling interface and configuration file format for the RDBMS objects context backend

SYNOPSIS

```
#include <selinux/label.h>

int selabel_lookup(struct selabel_handle *hnd,
                   char **context,
                   const char *object_name, int object_type);

int selabel_lookup_raw(struct selabel_handle *hnd,
                   char **context,
                   const char *object_name, int object_type);
```

DESCRIPTION

The DB contexts backend maps from a pair of object name and class into security contexts. It is used to find the appropriate context for data? base objects when relabeling a certain database. The returned context must be freed using freecon(3).

selabel_lookup(3) describes the function with its return and error

codes.

The object_name should be a fully qualified name using the hierarchy of database objects. For example, the pg_class table in the postgres data? base and pg_catalog schema should be qualified as:

Bpostgres.pg_catalog.pg_class

The NOTES section has further information on database support for name? space hierarchies.

The object_type argument should be set to one of the following values:

SELABEL_DB_DATABASE

The object_name argument specifies the name of a database itself, such as "postgres".

SELABEL_DB_SCHEMA

The object_name argument specifies the name of a schema object, such as "postgres.public".

SELABEL_DB_TABLE

The object_name argument specifies the name of a table object, such as "postgres.public.my_table"

SELABEL_DB_COLUMN

The object_name argument specifies the name of a column object, such as "postgres.public.my_table.user_id"

SELABEL_DB_TUPLE

The object_name argument specifies the name of a table object which contains the tuples to be relabeled, such as "postgresql.public.my_table". Note that we have no way to identify individual tuple objects, except for WHERE

clause on DML statements, because it has no name.

SELABEL_DB_PROCEDURE

The object_name argument specifies the name of a proce? dure object, such as "postgres.public.my_func". Note that we don't support lookup of individual security contexts for procedures which have the same name but different ar? guments.

SELABEL_DB_SEQUENCE

The object_name argument specifies the name of a sequence object, such as "postgres.public.my_seq".

SELABEL_DB_BLOB

The object_name argument specifies the name of a large object, such as "postgres.16308". Note that a large ob? ject does not have a name, so it is identified by its identifier value.

SELABEL_DB_VIEW

The object_name argument specifies the name of a view ob? ject, such as "postgres.public.my_view".

SELABEL_DB_LANGUAGE

The object_name argument specifies the name of a language object, such as "postgres.public.tcl".

SELABEL_DB_EXCEPTION

The object_name argument specifies the name of a excep? tion object.

SELABEL_DB_DATATYPE

The object_name argument specifies the name of a type or

domain object, such as postgres.public.my_type.

Any messages generated by selabel_lookup(3) are sent to stderr by de? fault, although this can be changed by selinux_set_callback(3).

selabel_lookup_raw(3) behaves identically to selabel_lookup(3) but does not perform context translation.

The FILES section details the configuration files used to determine the database object context.

OPTIONS

In addition to the global options described in selabel_open(3), this backend recognizes the following options:

SELABEL_OPT_PATH

A non-null value for this option specifies a path to a file that will be opened in lieu of the standard DB con? texts file. It tries to open the specfile designed for SE-PostgreSQL as default, so if another RDBMS uses this interface, it needs to give an explicit specfile designed for that RDBMS (see the FILES section for details).

FILES

The database context file used to retrieve a context depends on the SE? LABEL_OPT_PATH parameter passed to selabel_open(3). If NULL, then the SELABEL_OPT_PATH value will default to the active policy database con? texts location (as returned by selinux_sepgsql_context_path(3)), other? wise the actual SELABEL_OPT_PATH value specified is used (this option must be used to support databases other than SE-PostgreSQL).

The default database object contexts file is:

/etc/selinux/{SELINUXTYPE}/contexts/sepgsql_context

Where {SELINUXTYPE} is the entry from the selinux configuration file config (see selinux_config(5)).

The entries within the database contexts file are shown in the Object Name String Values and FILE FORMAT sections.

Object Name String Values

The string name assigned to each object_type argument that can be present in the database contexts file are:

```
??????????????????????????????????????
?object_type          ? Text Name    ?
??????????????????????????????????????
?SELABEL_DB_DATABASE  ? db_database  ?
??????????????????????????????????????
?SELABEL_DB_SCHEMA    ? db_schema    ?
??????????????????????????????????????
?SELABEL_DB_VIEW      ? db_view      ?
??????????????????????????????????????
?SELABEL_DB_LANGUAGE  ? db_language  ?
??????????????????????????????????????
?SELABEL_DB_TABLE     ? db_table     ?
??????????????????????????????????????
?SELABEL_DB_COLUMN    ? db_column    ?
??????????????????????????????????????
?SELABEL_DB_TUPLE     ? db_tuple     ?
??????????????????????????????????????
?SELABEL_DB_PROCEDURE ? db_procedure ?
??????????????????????????????????????
?SELABEL_DB_SEQUENCE  ? db_sequence  ?
??????????????????????????????????????
?SELABEL_DB_BLOB      ? db_blob      ?
```

??????????????????????????????????????

?SELABEL_DB_EXCEPTION ? db_exception ?

??????????????????????????????????????

?SELABEL_DB_DATATYPE ? db_datatype ?

??????????????????????????????????????

FILE FORMAT

Each line within the database contexts file is as follows:

object_type object_name context

Where:

object_type

This is the string representation of the object type shown in the Object Name String Values section.

object_name

The key used to obtain the context based on the ob?ject_type.

The entry can contain '*' for wildcard matching or '?' for substitution.

Note that if the '*' is used, then be aware that the or?der of entries in the file is important. The '*' on its own is used to ensure a default fallback context is as?signed and should be the last entry in the object_type block.

context

The security context that will be applied to the object.

The following example is for SE-PostgreSQL:

```
# ./contexts/sepgsql_contexts file
# object_type  object_name   context
db_database    my_database   system_u:object_r:sepgsql_db_t:s0
```

```
db_database    *            system_u:object_r:sepgsql_db_t:s0
db_schema      *.*          system_u:object_r:sepgsql_schema_t:s0
db_tuple       row_low      system_u:object_r:sepgsql_table_t:s0
db_tuple       row_high     system_u:object_r:sepgsql_table_t:s0:c1023
db_tuple       *.*.*        system_u:object_r:sepgsql_table_t:s0
```

NOTES

1. A suitable database contexts file needs to be written for the tar?
   get RDBMS and the SELABEL_OPT_PATH option must be used in sela?
   bel_open(3) to load it.

2. The hierarchy of the namespace for database objects depends on the
   RDBMS, however the selabel* interfaces do not have any specific
   support for a namespace hierarchy.

   SE-PostgreSQL has a namespace hierarchy where a database is the top
   level object with the schema being the next level. Under the schema
   object there can be other types of objects such as tables and pro?
   cedures. This hierarchy is supported as follows:

   If a security context is required for "my_table" table in
   the "public" schema within the "postgres" database, then the
   selabel_lookup(3) parameters for object_type would be SELA?
   BEL_DB_TABLE and the object_name would be "postgres.pub?
   lic.my_table", the security context (if available), would be
   returned in context.

3. If contexts are to be validated, then the global option SELA?
   BEL_OPT_VALIDATE must be set before calling selabel_open(3). If
   this is not set, then it is possible for an invalid context to be
   returned.

SEE ALSO

selinux(8), selabel_open(3), selabel_lookup(3), selabel_stats(3),

selabel_close(3), selinux_set_callback(3),

selinux_sepgsql_context_path(3), freecon(3), selinux_config(5)