



Computing Services

SASL

## SELinux security module upgrade for sasl

<i>SELinux security module upgrade for sasl</i> .....	<i>1</i>
SELinux environment.....	1
SELinux Informations.....	1
Gentoo Informations.....	1
General informations about the sasl module.....	1
Main scheme.....	1
Operation before module modifications.....	2
Module modification .....	2
Full sasl module.....	3

Note : this work has been made during an internship (summer 2007) at the ENST Bretagne of Rennes, France, SERES team. Note that these patches are only draft that have not been approved by the hardened-gentoo community.

Bug's reference: #199404 - [http://bugs.gentoo.org/show\\_bug.cgi?id=199404](http://bugs.gentoo.org/show_bug.cgi?id=199404)

### ***SELinux environment***

#### **SELinux Informations**

SELinux status: enabled  
 SELinuxfs mount: /selinux  
 Current mode: permissive  
 Mode from config file: permissive  
 Policy version: 21  
 Policy from config file: strict

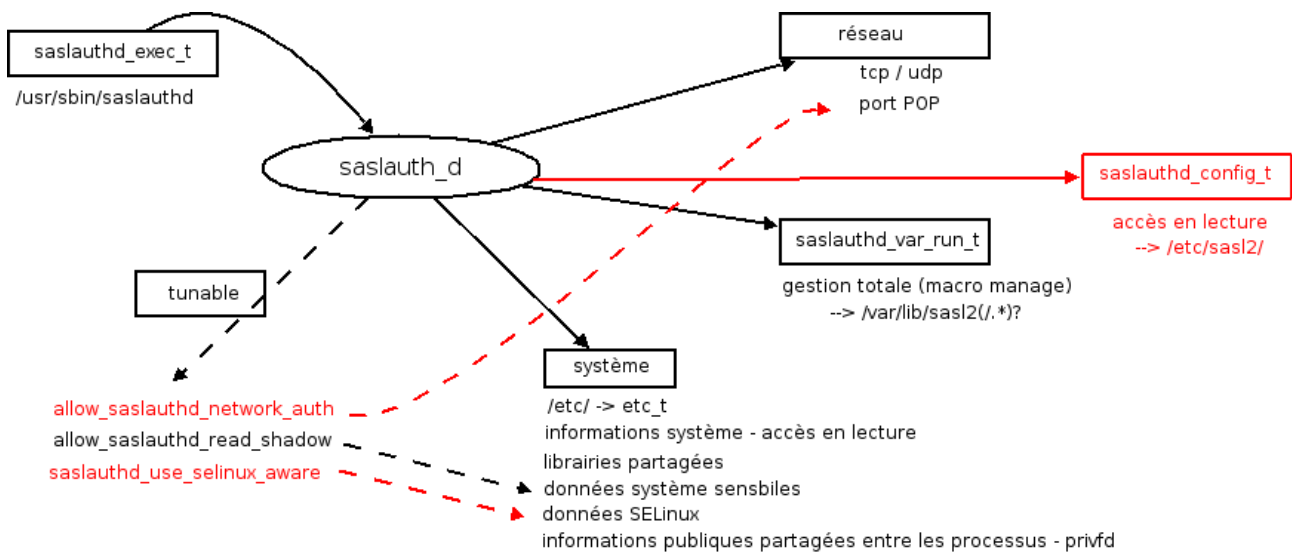
#### **Gentoo Informations**

Linux 2.6.20-hardened-r5  
 dev-libs/cyrus-sasl 2.1.22-r2.

### ***General informations about the sasl module***

#### **Main scheme**

The following figure describes sasl module schema. The modifications defined in this patch are in red.



## Operation before module modifications

The module modifications required new types.

As one of them may be used by other modules, it has been created on a new module, called `netif_support`. It creates the not yet defined `lo_netif_t`, used to label the `lo` interface.

Note that these modifications do not impact previous modules, as `lo_netif_t` is defined as a `netif_t` attribute.

```
module netif_support 1.0 ;
require {
attribute netif_type ;
}

type lo_netif_t ;
typeattribute lo_netif_t netif_type ;
```

The following command as then to be performed, in order to label the `lo` interface : `semanage interface -a -t lo_netif_t lo`.

## Module modification

The following modifications have been applied to the patch.

```

> policy_module(sasl,1.4.3)
+ type saslauthd_config_t;
+ read_files_pattern(saslauthd_t,saslauthd_config_t,saslauthd_config_t)
+
+ gen_tunable(allow_saslauthd_network_auth,true)
+ tunable_policy(`allow_saslauthd_network_auth',`
> corenet_tcp_sendrecv_all_if(saslauthd_t)
> corenet_tcp_sendrecv_all_nodes(saslauthd_t)
+ ',`
+ corenet_tcp_sendrecv_lo_if(saslauthd_t)
+ corenet_tcp_sendrecv_all_nodes(saslauthd_t)
+')
+ tunable_policy(`use_saslauthd_selinux_aware',`
      selinux_compute_access_vector(saslauthd_t)
+ ')

```

```

sasl.fc:
+ /etc/sasl2/*. * -- saslauthd_config_t

```

The modifications are about the network control reinforcement.

A new boolean option have been added to control the fact that the sasl module is able to connect on non localhost interfaces or not. Thus, it is possible to enhanced the security if the module is supposed to be accessed only by local services. Another boolean option is used to control the access to selinux option by saslauthd. This previous allow rule as been deactivated per default but can be reactivated by the boolean option modification.

The third modification defined a new type `saslauthd_config_t`. This type is used to enhanced the security of sasl configuration files. Currently, the files are labeled `etc_t` and thus no deep control can be performed by SELinux AVC daemon. This new type reinforce the acces control.

The `sasl.fc` modification can be immediately applied with `semanage fcontext -a -t saslauthd_config_t 'etc/sasl2/*. *'`

Not that the following command as to be performed in order to apply local restriction, as non-localhost connexion are allowed per default, `setsebool allow_saslauthd_network_auth 0`

## Full sasl module

```

policy_module(sasl,1.4.3)

#####
#
# Declarations
#

## <desc>
## <p>
## Allow sasl to read shadow
## </p>
## </desc>
gen_tunable(allow_saslauthd_read_shadow,false)
gen_tunable(allow_saslauthd_network_auth,true)

```

```
gen_tunable(use_saslauthd_selinux_aware, false)

type saslauthd_t;
type saslauthd_exec_t;
type saslauthd_config_t;

init_daemon_domain(saslauthd_t,saslauthd_exec_t)

type saslauthd_var_run_t;
files_pid_file(saslauthd_var_run_t)

#####
#
# Local policy
#
allow saslauthd_t self:capability setuid;
dontaudit saslauthd_t self:capability sys_tty_config;
allow saslauthd_t self:process signal_perms;
allow saslauthd_t self:fifo_file { read write };
allow saslauthd_t self:unix_dgram_socket create_socket_perms;
allow saslauthd_t self:unix_stream_socket create_stream_socket_perms;
allow saslauthd_t self:tcp_socket create_socket_perms;

manage_files_pattern(saslauthd_t,saslauthd_var_run_t,saslauthd_var_run_t)
manage_sock_files_pattern(saslauthd_t,saslauthd_var_run_t,saslauthd_var_run_t)
files_pid_filetrans(saslauthd_t,saslauthd_var_run_t,file)

kernel_read_kernel_sysctls(saslauthd_t)
kernel_read_system_state(saslauthd_t)

corenet_non_ipsec_sendrecv(saslauthd_t)

tunable_policy(`allow_saslauthd_network_auth',`
    corenet_tcp_sendrecv_all_if(saslauthd_t)
    corenet_tcp_sendrecv_all_nodes(saslauthd_t)
    ',`
    corenet_tcp_sendrecv_lo_if(saslauthd_t)
    corenet_tcp_sendrecv_all_nodes(saslauthd_t)
    ')

corenet_tcp_sendrecv_all_ports(saslauthd_t)
corenet_tcp_connect_pop_port(saslauthd_t)
corenet_sendrecv_pop_client_packets(saslauthd_t)

dev_read_sysfs(saslauthd_t)
dev_read_urand(saslauthd_t)

fs_getattr_all_fs(saslauthd_t)
fs_search_auto_mountpoints(saslauthd_t)

tunable_policy(`use_saslauthd_selinux_aware',`
    selinux_compute_access_vector(saslauthd_t)
    ')

```

```
auth_domtrans_chk_passwd(saslauthd_t)
auth_use_nsswitch(saslauthd_t)

domain_use_interactive_fds(saslauthd_t)

files_read_etc_files(saslauthd_t)
files_dontaudit_read_etc_runtime_files(saslauthd_t)
files_search_var_lib(saslauthd_t)
files_dontaudit_getattr_home_dir(saslauthd_t)
files_dontaudit_getattr_tmp_dirs(saslauthd_t)

init_dontaudit_stream_connect_script(saslauthd_t)

libs_use_ld_so(saslauthd_t)
libs_use_shared_libs(saslauthd_t)

logging_send_syslog_msg(saslauthd_t)

miscfiles_read_localization(saslauthd_t)
miscfiles_read_certs(saslauthd_t)

seutil_dontaudit_read_config(saslauthd_t)

sysnet_read_config(saslauthd_t)

userdom_dontaudit_use_unpriv_user_fds(saslauthd_t)
userdom_dontaudit_search_sysadm_home_dirs(saslauthd_t)

#ifdef `targeted_policy', `
    term_dontaudit_use_unallocated_ttys(saslauthd_t)
    term_dontaudit_use_generic_ptys(saslauthd_t)
    files_dontaudit_read_root_files(saslauthd_t)
`
')

# cjp: typeattribute dont work in conditionals yet
auth_can_read_shadow_passwords(saslauthd_t)
tunable_policy(`allow_saslauthd_read_shadow', `
    auth_tunable_read_shadow(saslauthd_t)
`)

optional_policy(`
    mysql_search_db(saslauthd_t)
    mysql_stream_connect(saslauthd_t)
`)

optional_policy(`
    seutil_sigchld_newrole(saslauthd_t)
`)

optional_policy(`
    udev_read_db(saslauthd_t)
`)
read_files_pattern(saslauthd_t, saslauthd_config_t, saslauthd_config_t)
```