

## NAME

avc\_open, avc\_destroy, avc\_reset, avc\_cleanup – userspace SELinux AVC setup and teardown

## SYNOPSIS

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

```
int avc_open(struct selinux_opt *options, unsigned nopt);
```

```
void avc_destroy(void);
```

```
int avc_reset(void);
```

```
void avc_cleanup(void);
```

## DESCRIPTION

**avc\_open()** initializes the userspace AVC and must be called before any other AVC operation can be performed.

**avc\_destroy()** destroys the userspace AVC, freeing all internal memory structures. After this call has been made, **avc\_open()** must be called again before any AVC operations can be performed. **avc\_destroy()** also closes the SELinux status page, which might have been opened manually by **selinux\_status\_open(3)**.

**avc\_reset()** flushes the userspace AVC, causing it to forget any cached access decisions. The userspace AVC normally calls this function automatically when needed, see **NETLINK NOTIFICATION** below.

**avc\_cleanup()** attempts to free unused memory within the userspace AVC, but does not flush any cached access decisions. Under normal operation, calling this function should not be necessary.

## OPTIONS

The userspace AVC obeys callbacks set via **selinux\_set\_callback(3)**, in particular the logging and audit callbacks.

The options which may be passed to **avc\_open()** include the following:

### AVC\_OPT\_SETENFORCE

This option forces the userspace AVC into enforcing mode if the option value is non-NULL; permissive mode otherwise. The system enforcing mode will be ignored.

## KERNEL STATUS PAGE

Linux kernel version 2.6.37 supports the SELinux kernel status page, enabling userspace applications to **mmap(2)** SELinux status state in read-only mode to avoid system calls during the cache hit code path.

**avc\_open()** calls **selinux\_status\_open(3)** to initialize the selinux status state.

**avc\_has\_perm(3)** and **selinux\_check\_access(3)** both check for status updates through calls to **selinux\_status\_updated(3)** at the start of each permission query and take the appropriate action.

Two status types are currently implemented. **setenforce** events will change the effective enforcing state used within the AVC, and **policyload** events will result in a cache flush.

## NETLINK NOTIFICATION

In the event that the kernel status page is not successfully **mmap(2)**'ed the AVC will default to the netlink fallback mechanism, which opens a netlink socket for receiving status updates. **setenforce** and **policyload** events will have the same results as for the status page implementation, but all status update checks will now require a system call.

**RETURN VALUE**

Functions with a return value return zero on success. On error, `-1` is returned and *errno* is set appropriately.

**AUTHOR**

Eamon Walsh <ewalsh@tycho.nsa.gov>

**SEE ALSO**

`selinux(8)`, `selinux_check_access(3)`, `avc_has_perm(3)`, `avc_context_to_sid(3)`, `avc_cache_stats(3)`, `avc_add_callback(3)`, `selinux_status_open(3)`, `selinux_status_updated(3)`, `selinux_set_callback(3)`, `security_compute_av(3)`