NAME

setaudit, **setaudit_addr** - set audit session state

SYNOPSIS

#include <bsm/audit.h>

int
setaudit(*auditinfo_t *auditinfo*);

int
setaudit_addr(*auditinfo_addr_t *auditinfo_addr, u_int length*);

DESCRIPTION

The **setaudit**() system call sets the active audit session state for the current process via the *auditinfo_t* pointed to by *auditinfo*. The **setaudit_addr**() system call sets extended state via *auditinfo_addr* and *length*.

The *auditinfo_t* data structure is defined as follows:

```
struct auditinfo {
        au_id_t         ai_auid;        /* Audit user ID */
        au_mask_t       ai_mask;        /* Audit masks */
        au_tid_t        ai_termid;      /* Terminal ID */
        au_asid_t       ai_asid;        /* Audit session ID */
};
typedef struct auditinfo        auditinfo_t;
```

The *ai_auid* variable contains the audit identifier which is recorded in the audit log for each event the process caused.

The *au_mask_t* data structure defines the bit mask for auditing successful and failed events out of the predefined list of event classes. It is defined as follows:

```
struct au_mask {
        unsigned int    am_success;     /* success bits */
        unsigned int    am_failure;     /* failure bits */
};
typedef struct au_mask  au_mask_t;
```

The *au_termid_t* data structure defines the Terminal ID recorded with every event caused by the process.

It is defined as follows:

```
struct au_tid {
        dev_t          port;
        u_int32_t      machine;
};
typedef struct au_tid   au_tid_t;
```

The *ai_asid* variable contains the audit session ID which is recorded with every event caused by the process.

The **setaudit_addr**() system call uses the expanded *auditinfo_addr_t* data structure supports Terminal IDs with larger addresses such as those used in IP version 6. It is defined as follows:

```
struct auditinfo_addr {
        au_id_t         ai_auid;        /* Audit user ID. */
        au_mask_t       ai_mask;        /* Audit masks. */
        au_tid_addr_t   ai_termid;      /* Terminal ID. */
        au_asid_t       ai_asid;        /* Audit session ID. */
};
typedef struct auditinfo_addr   auditinfo_addr_t;
```

The *au_tid_addr_t* data structure which includes a larger address storage field and an additional field with the type of address stored:

```
struct au_tid_addr {
        dev_t          at_port;
        u_int32_t      at_type;
        u_int32_t      at_addr[4];
};
typedef struct au_tid_addr      au_tid_addr_t;
```

These system calls require an appropriate privilege to complete.

RETURN VALUES

The **setaudit**() and **setaudit_addr**() functions return the value 0 if successful; otherwise the value -1 is returned and the global variable *errno* is set to indicate the error.

ERRORS

[EFAULT] A failure occurred while data transferred to or from the kernel failed.

[EINVAL] Illegal argument was passed by a system call.

[EPERM] The process does not have sufficient permission to complete the operation.

SEE ALSO

audit(2), auditon(2), getaudit(2), getauid(2), setauid(2), libbsm(3)

HISTORY

The OpenBSM implementation was created by McAfee Research, the security division of McAfee Inc., under contract to Apple Computer Inc. in 2004. It was subsequently adopted by the TrustedBSD Project as the foundation for the OpenBSM distribution.

AUTHORS

This software was created by McAfee Research, the security research division of McAfee, Inc., under contract to Apple Computer Inc. Additional authors include Wayne Salamon, Robert Watson, and SPARTA Inc.

The Basic Security Module (BSM) interface to audit records and audit event stream format were defined by Sun Microsystems.

This manual page was written by Robert Watson <rwatson@FreeBSD.org>.