NAME

libbsm - Basic Security Module (BSM) Audit API

LIBRARY

Basic Security Module Library (libbsm, -lbsm)

SYNOPSIS

#include <bsm/libbsm.h>

DESCRIPTION

The **libbsm** library routines provide an interface to BSM audit record streams, allowing both the parsing of existing audit streams, as well as the creation of new audit records and streams.

INTERFACES

The **libbsm** library provides a large number of Audit programming interfaces in several classes: event stream interfaces, class interfaces, control interfaces, event interfaces, I/O interfaces, mask interfaces, notification interfaces, token interfaces, and user interfaces. These are described respectively in the au_class(3), au_control(3), au_event(3), au_mask(3), au_notify(3), au_stream(3), au_token(3), and au_user(3) manual pages.

Audit Event Stream Interfaces

Audit event stream interfaces support interaction with file-backed audit event streams: au_close(3), au_close_buffer(3), au_free_token(3), au_open(3), au_write(3), audit_submit(3).

Audit Class Interfaces

Audit class interfaces support the look up of information from the audit_class(5) database: endauclass(3), getauclassent(3), getauclassent_r(3), getauclassnam(3), getauclassnam_r(3), setauclass(3).

Audit Control Interfaces

Audit control interfaces support the look up of information from the audit_control(5) database: endac(3), setac(3), getacdir(3), getacfilesz(3), getacflg(3), getacmin(3), getacna(3), getacpol(3), au_poltostr(3), au_strtopol(3).

Audit Event Interfaces

Audit event interfaces support the look up of information from the audit_event(5) database: endauevent(3), setauevent(3), getauevent(3), getauevent_r(3), getauevnam(3), getauevnam_r(3), getauevnonam(3), getauevnonam_r(3), getauevnum(3), getauevnum_r(3).

Audit I/O Interfaces

Audit I/O interfaces support the processing and printing of tokens, as well as the reading of audit records: au_fetch_tok(3), au_print_tok(3), au_read_rec(3).

Audit Mask Interfaces

Audit mask interfaces convert support the conversion between strings and *au_mask_t* values. They may also be used to determine if a particular audit event is matched by a mask: au_preselect(3), getauditflagsbin(3), getauditflagschar(3).

Audit Notification Interfaces

Audit notification routines track audit state in a form permitting efficient update, avoiding frequent system calls to check the kernel audit state: au_get_state(3), au_notify_initialize(3), au_notify_terminate(3). These interfaces are implemented only for Darwin/Mac OS X.

Audit Token Interface

Audit token interfaces permit the creation of tokens for use in creating audit records for submission to event streams. Each interface converts a C type to its *token_t* representation: au_to_arg(3), au_to_arg32(3), au_to_arg64(3), au_to_attr64(3), au_to_data(3), au_to_exec_args(3), au_to_exec_env(3), au_to_exit(3), au_to_file(3), au_to_groups(3), au_to_header32(3), au_to_header64(3), au_to_in_addr(3), au_to_in_addr_ex(3), au_to_ip(3), au_to_ipc(3), au_to_ipc_perm(3), au_to_iport(3), au_to_me(3), au_to_newgroups(3), au_to_opaque(3), au_to_path(3), au_to_process(3), au_to_process32(3), au_to_process64(3), au_to_process_ex(3), au_to_process32_ex(3), au_to_process64_ex(3), au_to_return(3), au_to_return32(3), au_to_return64(3), au_to_seq(3), au_to_sock_inet(3), au_to_sock_inet32(3), au_to_sock_inet128(3), au_to_socket_ex(3), au_to_subject(3), au_to_subject32(3), au_to_subject64(3), au_to_subject_ex(3), au_to_subject32_ex(3), au_to_subject64_ex(3), au_to_text(3), au_to_trailer(3), au_to_zonename(3).

Audit User Interfaces

Audit user interfaces support the look up of information from the audit_user(5) database: au_user_mask(3), endauuser(3), setauuser(3), getauuserent(3), getauuserent_r(3), getauusernam(3), getauusernam_r(3), getfauditflags(3).

Audit Constant Conversion Interfaces

These functions convert between BSM and local constants, including the errno(2) number, socket type, and protocol famil spaces, and must be used to generate and interpret BSM return and extended socket tokens: au_bsm_to_domain(3), au_bsm_to_errno(3), au_bsm_to_fcntl_cmd(3), au_bsm_to_socket_type(3), au_domain_to_bsm(3), au_errno_to_bsm(3), au_fcntl_cmd_to_bsm(3), au_socket_type_to_bsm(3).

SEE ALSO

au_class(3), au_domain(3), au_errno(3), au_mask(3), au_notify(3), au_socket_type(3), au_stream(3),

au_token(3), au_user(3), audit_submit(3), audit_class(5), audit_control(5)

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 Robert Watson, Wayne Salamon, and Suresh Krishnaswamy for McAfee Research, the security research division of McAfee, Inc., under contract to Apple Computer, Inc.

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

BUGS

Bugs would not be unlikely.

The **libbsm** library implementations are generally thread-safe, but not reentrant.

The assignment of routines to classes could use some work, as it is decidely ad hoc. For example, **au_read_rec**() should probably be considered a stream routine.