NAME

au_preselect, **getauditflagsbin**, **getauditflagschar** - convert between string and numeric values of audit masks

LIBRARY

Basic Security Module Library (libbsm, -lbsm)

SYNOPSIS

#include <bsm/libbsm.h>

int
au_preselect(*au_event_t event, au_mask_t *mask_p, int sorf, int flag*);

int
getauditflagsbin(*char *auditstr, au_mask_t *masks*);

int
getauditflagschar(*char *auditstr, au_mask_t *masks, int verbose*);

DESCRIPTION

These interfaces support processing of an audit mask represented by type *au_mask_t*, including conversion between numeric and text formats, and computing whether or not an event is matched by a mask.

The **au_preselect**() function calculates whether or not the audit event passed via *event* is matched by the audit mask passed via *mask_p*. The *sorf* argument indicates whether or not to consider the event as a success, if the AU_PRS_SUCCESS flag is set, or failure, if the AU_PRS_FAILURE flag is set. The *flag* argument accepts additional arguments influencing the behavior of **au_preselect**(), including AU_PRS_REREAD, which causes the event to be re-looked up rather than read from the cache, or AU_PRS_USECACHE which forces use of the cache.

The **getauditflagsbin**() function converts a string representation of an audit mask passed via a character string pointed to by *auditstr*, returning the resulting mask, if valid, via **masks*.

The **getauditflagschar**() function converts the audit event mask passed via **masks* and converts it to a character string in a buffer pointed to by *auditstr*. See the *BUGS* section for more information on how to provide a buffer of sufficient size. If the *verbose* flag is set, the class description string retrieved from audit_class(5) will be used; otherwise, the two-character class name.

IMPLEMENTATION NOTES

The **au_preselect**() function makes implicit use of various audit database routines, and may influence the behavior of simultaneous or interleaved processing of those databases by other code.

RETURN VALUES

The **au_preselect**() function returns 0 on success, or returns -1 if there is a failure looking up the event type or other database access, in which case *errno* will be set to indicate the error. It returns 1 if the event is matched; 0 if not.

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

SEE ALSO

libbsm(3), audit_class(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

The *errno* variable may not always be properly set in the event of an error.

The **getauditflagschar**() function does not provide a way to indicate how long the character buffer is, in order to detect overflow. As a result, the caller must always provide a buffer of sufficient length for any possible mask, which may be calculated as three times the number of non-zero bits in the mask argument in the event non-verbose class names are used, and is not trivially predictable for verbose class names. This API should be replaced with a more robust one.