NAME

ausearch_add_item − build up search rule

SYNOPSIS

#include <auparse.h>

int ausearch_add_item(auparse_state_t ****au*, const char ****field***, const char ****op***, const char ****value***, ausear ch_rule_t** *how*)

DESCRIPTION

ausearch_add_item adds one search condition to the current audit search expression. The search conditions can then be used to scan logs, files, or buffers for something of interest. The *field* value is the field name that the *value* will be checked for. The *op* variable describes what kind of check is to be done. Legal *op* values are:

exists
: just check that a field name exists

=
: locate the field name and check that the value associated with it is equal to the value given in this rule.

!=
: locate the field name and check that the value associated with it is NOT equal to the value given in this rule.

The value parameter is compared to the uninterpreted field value. If you are trying to match against a field who's type is AUPARSE_TYPE_ESCAPED, you will want to use the ausearch_add_interpreted_item() function instead.

The *how* value determines how this search condition will affect the existing search expression if one is already defined. The possible values are:

AUSEARCH_RULE_CLEAR
: Clear the current search expression, if any, and use only this search condition.

AUSEARCH_RULE_OR
: If a search expression *E* is already configured, replace it by **(***E* **||** *this_search_condition***)**.

AUSEARCH_RULE_AND
: If a search expression *E* is already configured, replace it by **(***E* **&&** *this_search_condition***)**.

RETURN VALUE

Returns −1 if an error occurs; otherwise, 0 for success.

SEE ALSO

ausearch_add_expression(3), **ausearch_add_interpreted_item**(3), **ausearch_add_timestamp_item**(3), **ausearch_add_regex**(3), **ausearch_set_stop**(3), **ausearch_clear**(3), **ausearch_next_event**(3), **ausearch_cur_event**(3), **ausearch−expression**(5).

AUTHOR

Steve Grubb